Capturing the Temporal Properties of Complex Systems: an Evaluation of the Timebands Approach

Roderick Grenville White

Submitted for the degree of
Doctor of Philosophy

University of York
Department of Computer Science

September 2009

Abstract

The management and use of time is important to some extent in all systems. Be it simply the time between failures, the time to repair, the responsiveness to an operator's demand, or the need for regular and frequent updates to the flight control surfaces of a fly-by-wire aircraft, time is a key parameter. Even though the times related to these characteristics span many orders of magnitude, from years at one extreme to milliseconds at the other, a range of 10^{12}, the engineering techniques used to specify and capture the designs of these systems use one simple concrete notion of time — the second. In a discipline where abstraction is seen as being a key approach to the successful management of complexity, it appears paradoxical that there are no abstractions for time.

This thesis argues that the provision of a banded abstraction of time supports better management, treatment and use of time. The application of such an abstraction as the primary focus for the architectural organisation of systems will ultimately result in systems of higher quality, ones which are more readily adapted to new roles, and can be more easily and predictably integrated into larger collaborations — systems-of-systems. Such improvements can reduce development timescales, resulting in the earlier availability of the final system; in time sensitive domains this early availability can ultimately lead to both commercial and strategic advantages.

An existing timebands framework is the primary focus for this thesis. The major contributions of this thesis are to:

- Extend the timebands framework in two ways: the addition of the concept of system modes; and the development of a process to identify and populate the bands of a system;
- Provide an in-depth evaluation of the extended timebands framework through both an analysis of its definition, and its application in an industrial scale case study, demonstrating that its principles are sound, and that it has a clear role in the development of systems.

Contents

Abstract 3

Contents 5

List of figures 11

List of tables 13

Acknowledgements 15

Declaration 17

1 Introduction 19

1.1 Systems and the importance of time 20

1.1.1 The importance of time 20

1.1.2 The perception and measurement of time 21

1.1.3 The use and management of time in a system context 21

1.2 Central proposition 22

1.2.1 Research approach/method 23

1.2.2 Contribution 24

1.3 The structure of the dissertation 24

2 A Review and Assessment of the Timebands Framework 27

2.1 Motivation and background 28

2.2 A review of the associated literature 29

2.3 A description of the framework 34

2.3.1 The bands 34

2.3.2 Component elements — activities and events 36

2.3.3 Temporal relationships 37

2.3.4 Clocks . . . 39
2.3.5 Behaviours . . . 39
2.4 Timebands applied . . . 39
2.4.1 Description and scoping . . . 40
2.4.2 Constructing the model . . . 41
2.5 An initial assessment of the timebands framework . . . 50
2.5.1 Strengths . . . 50
2.5.2 Weaknesses . . . 51
2.5.3 Unclear aspects . . . 52
2.6 Recent developments . . . 52
2.7 Summary . . . 53

3 System Modes: their Relationship to the Timebands Framework 55
3.1 The concept and nature of mode and state . . . 55
3.1.1 A survey of the terms mode and state . . . 56
3.1.2 An introduction to Architecture Description Languages . . . 58
3.1.3 The use of Mode and State in Architecture Description Languages . . . 59
3.2 Clarification of the differences and relationships between the concepts . . . 61
3.2.1 System description . . . 61
3.2.2 A simple mode model . . . 62
3.2.3 An alternative model . . . 64
3.2.4 A comparison of the two models . . . 65
3.3 The relationships between modes and the artefacts of the timebands framework . . . 66
3.3.1 Characteristics of mode models of systems . . . 66
3.3.2 Mapping modes to timebands — an informal example system . . . 67
3.3.3 Timebands selection and mapping . . . 70
3.3.4 Mapping inter-mode transitions to the timebands . . . 72
3.3.5 The relationship between modes and time bands . . . 77
3.3.6 The interactions between modes and inter-mode transitions . . . 82
3.4 Summary . . . 88

4 System Modes: their Inclusion in the Timebands Framework 89
4.1 Extension to the formal framework . . . 90

4.1.1 Modes . . . 90
4.1.2 Transitions . . . 91
4.1.3 The overall mode model . . . 95
4.2 Using modes as constraints — invariants, and pre- and post-conditions . . . 97
4.2.1 Informal description . . . 97
4.2.2 Formal description . . . 100
4.3 Application of the formal approach to an example system . . . 102
4.3.1 Formal description of the mode model . . . 103
4.3.2 Using the mode model to define constraints . . . 105
4.4 Hierarchical mode models . . . 106
4.4.1 Band allocation . . . 107
4.4.2 The init "transition" . . . 108
4.4.3 Mode model termination — Powered to Unpowered . . . 109
4.4.4 Extensions to the formal model to accommodate hierarchy . . . 110
4.5 Concurrent modes . . . 112
4.6 Summary . . . 114

5 Timebands: Creating a Model — Band Selection and Artefact Allocation 117
5.1 Background . . . 118
5.2 The nature of the bands . . . 119
5.3 Band selection . . . 123
5.3.1 Bands for a new system . . . 123
5.3.2 Bands for an existing system . . . 127
5.3.3 Systems-of-systems . . . 130
5.4 Band selection and model creation: a unified approach . . . 132
5.4.1 Individual monolithic systems . . . 133
5.4.2 Systems-of-systems . . . 138
5.4.3 Band selection process summary . . . 139
5.5 Allocation of artefacts to the bands . . . 141
5.5.1 Impact of the partial ordering of bands . . . 141
5.5.2 Artefact identification . . . 142
5.5.3 Allocation to bands . . . 145
5.6 Artefact interactions and relationships . . . 145

5.7 Summary 146

6 Application of the Timebands Framework: A Case Study 149

6.1 System description 150

6.1.1 General background and overall system 151

6.1.2 The component systems 152

6.1.3 Major stakeholders and the environment 158

6.2 Constructing the case study — the research method 160

6.3 Initial overall system-of-systems model 162

6.3.1 Developing the model 162

6.3.2 The initial model 172

6.3.3 Populating the model — artefact identification, placement and relationships 173

6.3.4 Relationships between bands 180

6.3.5 Observations on the initial system-of-systems model 182

6.4 Component models 183

6.5 Payload delivery vehicle model 184

6.5.1 Identifying the bands 184

6.5.2 Populating the model — artefact identification, placement and relationships 198

6.5.3 Identifying Inter-band relationships 218

6.6 Local command post model 220

6.6.1 Identifying the bands 221

6.6.2 Populating the model — artefact identification, placement and relationships 228

6.6.3 Identifying the inter-band relationships 240

6.7 Final overall system-of-systems model 243

6.7.1 Inter-model band relationships 243

6.7.2 Inter-model activity and event relationships 248

6.8 Summary 252

7 Evaluation, Assessment and Conclusions 255

7.1 Specific issues 257

7.1.1 The bands and their characteristics 257

7.1.2 Activities and events . . . 261
7.1.3 Precedence and inter-band mappings . . . 267
7.1.4 System modes . . . 271
7.1.5 Model creation process . . . 275
7.1.6 Tool support . . . 279
7.2 General issues . . . 283
7.2.1 ...with the framework . . . 283
7.2.2 ...with the modelling process . . . 285
7.3 Evaluation and assessment summary . . . 288
7.3.1 Open questions and issues . . . 289
7.4 Overall summary and final conclusions . . . 290
7.4.1 Thesis summary . . . 290
7.4.2 Final conclusions . . . 291
7.4.3 Suggestions for further research . . . 292
7.4.4 Contributions and closing remarks . . . 294

A A Review of Architecture Description Languages 295

A.1 Origins and history . . . 296
A.1.1 Origins . . . 297
A.1.2 Standardisation . . . 299
A.1.3 ADLs, the UML and more recent developments . . . 299
A.1.4 Modelling non-functional properties . . . 300
A.2 What is (and what is not) an ADL? . . . 301
A.2.1 The relationship between software architecture and ADLs . . . 301
A.2.2 What is an ADL — its essence and characteristics . . . 304
A.3 Related areas of research and similar languages . . . 311
A.3.1 The UML . . . 311
A.3.2 System and Architecture Design Languages . . . 311
A.4 Application to real-time and embedded systems . . . 312
A.5 A closer look at a number of languages . . . 313
A.5.1 Selection and comparison approach . . . 313
A.5.2 Description and appraisal of the languages . . . 315
A.6 Comparison summary . . . 323

A.7 Summary 323

Abbreviations **327**

Bibliography **329**

List of Figures

2.1 Time bands example 36
2.2 Timeline for the ATM cash withdrawal transaction 41
2.3 Event-activity relationship for a button press 46
2.4 Event-activity relationships for the PIN verification 47
2.5 Event-activity relationships for the authorisation and dispensing of money . . 48

3.1 Mode model for a simple system 63
3.2 The alternative mode model for a simple system 65
3.3 The orthogonality of mode, state and function 66
3.4 Mode model for a simple system 68
3.5 Chain of actions resulting from a mode transition initiation 69
3.6 Mapping of basic activities to the time bands 73
3.7 Mapping of events and activities across the time bands 75
3.8 Track — Flight mode change 84
3.9 Mode change options 86

4.1 Simple inter-mode transitions 92
4.2 Example of cross-band constraint mapping 99
4.3 Mode diagram for example system 103
4.4 Example system using concurrent modes 113

5.1 Band selection process 140

6.1 Schematic for overall system 152
6.2 Modes for a deployment 167
6.3 Modes for the mission under **Active** 168
6.4 Deployment band — activities and events 176
6.5 Mission band — activities and events 178

6.6 Operator-Interaction band — activities and events . . . 180
6.7 System-of-systems inter-band relationships . . . 181
6.8 Context of the central processing element of the vehicle . . . 185
6.9 **Active**^{V} mode model — the vehicle context of **Active** . . . 188
6.10 **Pre-launch** mode model . . . 189
6.11 **Transit** mode model . . . 189
6.12 **Engaging** mode model . . . 191
6.13 Overall real-time network design . . . 193
6.14 Mission control . . . 193
6.15 Guidance . . . 194
6.16 Navigation . . . 194
6.17 Airframe control . . . 195
6.18 Video processing . . . 195
6.19 VEHICLE-LIFE band — activities and events . . . 200
6.20 MISSION^{V} band — activities and events . . . 203
6.21 FLIGHT-PHASES band — activities and events . . . 207
6.22 CAPABILITY band — activities and events . . . 211
6.23 CONTROL band — activities and events . . . 214
6.24 BASIC-BLOCK band — activities and events . . . 217
6.25 Local command post context . . . 221
6.26 TACTICAL band — activities and events . . . 231
6.27 RATIONAL band — activities and events . . . 234
6.28 COGNITIVE band — activities and events . . . 238
6.29 LCP-BLOCKS band — activities and events . . . 240
6.30 A comparison of the bands across the three models . . . 244

7.1 Granularity relationships between Years, Months, Weeks and Days . . . 260
7.2 Possible schemes for inter-band mappings for strict precedence . . . 269
7.3 Mappings between modes and mode models . . . 272

A.1 Acme solution to the producer–consumer architecture example . . . 318
A.2 MetaH solution to the producer–consumer architecture example . . . 320

List of Tables

2.1 Time scale of human action 31

2.2 ATM band characteristics 43

2.3 Event-band mapping 44

3.1 Use of the terms mode and state in a selection of ADL-like languages 59

3.2 Band characteristics for the autonomous vehicle example 71

4.1 Precisions and granularities for the autonomous vehicle example 103

4.2 Persistence bounds for modes 104

6.1 Band Characteristics 172

6.2 SYSTEM-LIFE band definition 174

6.3 DEPLOYMENT band definition 175

6.4 MISSION band definition 176

6.5 OPERATOR-INTERACTION band definition 179

6.6 Vehicle model band characteristics 197

6.7 Vehicle inter-band granularity mappings 197

6.8 VEHICLE-LIFE band definition 199

6.9 $\text{MISSION}^{\text{V}}$ band definition 201

6.10 FLIGHT-PHASES band definition 206

6.11 CAPABILITIES band definition 211

6.12 CONTROL band definition 213

6.13 BASIC-BLOCK band definition 216

6.14 Local Control Post model band characteristics 228

6.15 Local command Post inter-band granularity mappings 228

6.16 TACTICAL band definition 229

6.17 RATIONAL band definition 232

6.18 COGNITIVE band definition 236
6.19 LCP-BLOCKS band definition 239
6.20 Revised band characteristics for the system-of-systems model 248
6.21 Overall system inter-band granularity mappings 248
6.22 Origins of artefacts in the overall model 250
6.23 Simple metrics for the models 253

A.1 Examples of ADLs 310
A.2 Comparison summary 323

Acknowledgements

Without the help and support of a number of people, it is doubtful that this thesis would have been completed. First and foremost, thanks go to my supervisor Professor Alan Burns who, despite my rather irregular progress, provided perfectly balanced support and guidance over the protracted course of my studies. Thanks are also due to my external examiner Professor Ian Sommerville and internal assessor Dr Neil Audsley for their helpful suggestions and comments, and to the other members of the Real-Time Systems group for making my visits to York enjoyable and productive.

My family has put up with much over the course of these studies, without their forbearance and support this thesis would not have been completed.

Finally, I acknowledge the support given by my employer MBDA for allowing me to undertake the PhD and providing the necessary funding, and my colleagues who have encouraged me throughout the process.

Declaration

The contents of this thesis, and the research it represents, are my own original work and have not been published elsewhere — all sources of other material are acknowledged with explicit references.

Chapter 1

Introduction

Time is an often marginalised aspect of a system's character. Whilst there is some recognition of the importance of time for those systems described as "real-time", it is a facet that is given little attention across systems more generally; nevertheless, in the widest sense it would be fair to say that systems exhibit, or rely on, some notion of time at all levels of abstraction.

As systems become more complex, and the deployment of those based on computers becomes more pervasive, the issues relating to time in all its guises tend to become more significant; this tendency is further exacerbated by the trend of constructing new, large systems from sets of existing component systems — the development of the so-called systems-of-systems [50]. However, whist this significance increases, and despite significant research and development, the notations used in the description of systems at their various levels of abstraction do not adequately capture the essential temporal information. Actually, in many examples of these notations there is neither a capability, nor any attempt to provide a capability that allows the temporal aspects to be addressed, let alone one that truly recognises their importance. Even in the case of notations intended for use with real-time systems, a field where the importance of time is recognised and widely understood, time is often treated as a "second class citizen" when compared to its functional counterpart, an example of such a notation being the Architecture Analysis & Design Language (AADL) [91].

Given the relevance of time in a systems context, the question this poses is: how may time be effectively and beneficially included in system descriptions?

1.1 Systems and the importance of time

In order to gain an acceptance, or even the appropriate recognition, of the importance of time to systems in a general sense, a number of questions have to be addressed:

- Why is time important?
- What is time — how is it perceived and measured?
- How can time be used and managed in a systems context?

These questions, together with their answers given in the sections below, provide a path towards the comprehensive integration of time with system descriptions.

1.1.1 The importance of time

Time is important throughout the life of the system, to some degree almost every aspect of a system has some temporal context, some examples are: the time to develop or manufacture; the expected deployed, or in-service life; maintenance periods; time to train users; its reliability (or mean time between failures) and availability; and so on — there are many more. The degree to which the temporal context is important varies from aspect to aspect, and from system to system, often it is difficult to identify the true importance of time in any particular case.

It is notable that amongst the above examples the durations considered vary considerably; from an in-service life of decades, to times relating to the responsiveness of a control system, perhaps a few milliseconds or less, giving a range of values somewhere in the order of 10^{12}. Equally the temporal context for a given aspect will vary from system to system. For example, the disposal of a nuclear power station might itself run to a number of years, whereas the time taken for disposal of a simple consumer device may well be negligible by comparison — but in each case the time taken will be something to be accounted for as it will affect the “price” of the item.

In the case of real-time systems, time is acknowledged as having a heightened importance; in addition to the examples identified above, which may be considered as being in some way peripheral, in these systems time has a central role, with their performance, in the widest sense, being fundamentally dependent on time. Allied with this central role is the recognition that time can be a fundamental parameter to be used in trade-offs and optimisations for these systems. In addition, knowledge of the temporal properties that emerge from the implementations of these systems are of particular importance for the effective management of “component” obsolescence and replacement.

Whist, as noted above, it might not always be immediately apparent that time is a particularly important "issue" for a particular system there are often underlying issues that can makes its proper consideration crucial. Following on from the obsolescence issue above, it is becoming common practice to extend the life of a particular system by coupling it to other systems, with the result of an improvement in capability — for this coupling to be successful an accurate knowledge of the timing properties of the interfaces of both systems is essential.

1.1.2 The perception and measurement of time

Technically time is measured in seconds — all quantities are either multiples or fractions of seconds. Humankind has created a number of abstractions of time that allow larger concepts to be discussed in isolation from these low-level units. These abstractions do not, necessarily, map precisely on to absolute multiples of seconds; the year, for example, is a variable quantity of 365 or 366 days, or in the base units, 31,536,000 or 31,622,400 seconds — not to mention the occasional "leap-second". This abstraction could best be described as a technical one, in that there is a fundamental basis for the values used; more generally however, humankind uses terms such as "year" far more loosely, and these uses are perhaps best described as social abstractions. Unlike the technical abstractions, which are invariant, these social abstractions vary from one society to another, with Western and Japanese culture adopting the strictest view, whilst at the other extreme, there are societies that do not really "measure" time; for example, ones that rely on seasonal variations to indicate the passing of years. Levine [54] provided an extensive discussion regarding the perception of time across different cultures.

Given the above abstract views of time it is perhaps strange therefore that there are no generally accepted abstractions over the concept of time when it is used "constructively" in a technical situation, it appears that invariably the multiples of seconds are used in an absolute form. Clearly in some circumstances, for example long-lived space missions, temporal precision is important and the use of seconds is not inappropriate. However, in the case of a system that has a nominal life of ten years, dictated by some fixed-life component, it is clear that this value is not precise, and a margin of a few days or weeks around the ten years would be quite acceptable. This simple example indicates that, even in technical situations, the perception and measurement of time does not have to, and probably should not, follow a single, high precision model.

1.1.3 The use and management of time in a system context

It is clear from the above that time is an important quantity across many aspects of a system, thus it follows that its "correct" (effective) use and management should lead to

better system solutions. A significant part of this correctness is the ability to create temporal abstractions allowing time to be "discussed" in a manner appropriate to the aspect of the system under consideration. The normal approach of a single measure of time has tended to result in the over-specification of timing issues and perhaps an unfounded and misguided belief that greater precision will deliver a better system. A number of authors have noted the difficulties and impact of not having the "real" temporal requirements for system, these include Hutchesson and Hayes [44], Norström *et al.* [72] and Bate *et al.* [5]. Their observations are reinforced by personal experience where it has been observed that the true timing requirements usually only emerge once the development of a system is well advanced, and at a point where it is often too late to make significant changes; the consequence of this being an implementation that is overly complex and inherently inflexible.

Whilst it is desirable to use and manage time effectively this is not supported by any mainstream "design" notations; however there is an emerging notation, that of "timebands" defined by Burns *et al.* [11].[1] In outline, the timebands framework is comprised of a small number of concepts and components. Fundamentally it promotes a banded notion of time, one where each band is defined in terms of two characteristics: it granularity (or unit of time) and its precision. The framework allows for a flexible relationship between the bands, with no enforced band-to-band hierarchy.

The bands, in turn, provide the temporal context for two distinct components: events and activities. Events are defined as being "instantaneous" within the band in which they are defined whilst activities, conversely, have a duration measured in the granularity of their defining band. Within a band the events may be related to one another through precedence relationships, whilst events in a higher (coarser) band may be related to activities in a lower one through mapping relationships. The framework defines a number of other concepts, these are described later in Chapter 2 where timebands are considered in more detail.

1.2 Central proposition

The fundamental goal of the research is to evaluate the timebands framework and identify how (or whether) the concept of timebands may be used to provide a "better" architectural description for embedded real-time systems. The focus here is predominately on the technical aspects of the systems, and does not really consider the wider societal in-

[1] Burns *at al.* use the phrase "time bands" in the original work, more recently the single word "timebands" has become the accepted form.

teractions, there is however some consideration of the more detailed human-machine interactions.

In this context of this thesis the term “better” refers primarily to the time (and hence cost) required to develop and deploy the system (or capability) — in the military context the earlier the system is deployed the more effective it is likely to be. Equally the sooner a set of existing systems can be integrated into a system-of-systems to provide a novel capability the more effective this too will be.

This leads to the central proposition:

> *Current approaches to the development of real-time embedded systems fail to capture adequately their non-functional properties, especially time, in a consistent and coherent manner. Using a banded notion of time as the primary consideration in their architectural organisation addresses this failing. This results in systems that have well defined temporal properties that do not unnecessarily constrain their implementation, and provide a common framework for the combination of these systems.*

This represents a radical change from the usual approach to defining architectures where, as noted earlier, the focus is on the functional aspects. It seeks to address the problem of how to make non-functional properties, time in this case, a central consideration in the architecture — something that current architectural notations do not achieve or support.

1.2.1 Research approach/method

This research is based on the existence of the original timebands framework defined by Burns *et al.* [11]; the dual goals being to identify its utility and limitations, and develop corresponding improvements and extensions. This is approached in three ways:

- Consideration of the capabilities of the original timebands framework, both in isolation and through it application to an example real-time system, identifying any critical limitations and omissions;
- Provision of extensions to address the identified deficiencies;
- The application of the extended framework to an industrial scale case study focused on a significant system consisting of a number of components which are treated as systems in their own right.

The results of the case study are used to help focus a detailed evaluation of the (extended) framework, from which further suggestions for its improvement emerge.

1.2.2 Contribution

The thesis provides a wide-ranging critical assessment of the timebands framework, both in its original form and once extended to address key deficiencies. From this the key contributions of this research can be seen in two areas:

- Consideration of the issues to be addressed in the creation of timebands models for both existing and new systems — defining a process to support such modelling;
- Application of the extended framework and process to an industrial scale case study — creating both component and system-of-system models;

1.3 The structure of the dissertation

The remainder of this thesis is divided in to the following chapters.

Chapter 2 A review and assessment of the timebands framework, its origins, status, capabilities and shortcomings.

Chapter 3 An examination of the nature of system modes, contrasting "mode" with "state", and the relationship of modes and inter-mode transitions with the timebands framework.

Chapter 4 Continuing from Chapter 3 developing the formal extension of the timebands framework to include support for the concept of system modes and modal hierarchies.

Chapter 5 The development of a process for creating timebands models, addressing band selection and definition, and artefact allocation, and considering the issues relating the modelling of both component systems and systems-of-systems.

Chapter 6 The application of the extended timebands framework and model creation process to a large-scale, wide-ranging industrial case study, creating two contrasting component models and using these to develop another model for the overall system-of-systems;

Chapter 7 An evaluation and assessment of the timebands framework using the models produced by the case study in Chapter 6; drawing detailed conclusions

regarding the capabilities of the framework and the feasibility of its application to architectural description, identifying both the strengths and weaknesses, and general utility of the approach. The thesis concludes with an overall summary, and draws some final conclusions with respect to the central proposition; some suggested topics for future research are also included.

A description and survey of Architecture Description Languages (ADL) is given as an Appendix, this provides some background on how time, and non-functional properties more generally, are captured in the current functional/architectural design notations used for real-time systems.

Chapter 2

A Review and Assessment of the Timebands Framework

This chapter considers the background to, and the principles of the timebands framework; providing a critical review of this approach to the management of temporal issues in complex systems.

The baseline definition of the timebands framework is that given by Burns *et al.* [11]; fundamentally the timebands framework has been developed to:

> *... make better use of 'time', with the aim of producing more dependable computer-based systems.* [11]

Whilst in itself it is not intended as an approach for describing the architectures of systems, it allows the temporal aspects of such architectures, which as noted in the previous chapter are currently poorly served across architectural description and design notations in general, to be specified or described, and reasoned about.

The remainder of this chapter is divided to address the following topics:

- The general motivation for, and the background to the timebands framework;
- A review of the associated and background literature — the origins and inspirations of the framework;
- A description of the framework — the principal elements and their relationships;
- The application of the framework to a simple example — a limited banking Automated Teller Machine (ATM);

- An initial outline assessment of the framework — noting its apparent strengths and weaknesses, and identifying areas where there is currently a lack of clarity;
- A brief summary of recent developments — publications since the original definition of the framework [11].

The assessment of the framework is based on two sources of information: the original definition [11]; and the experience of the application to the example system presented in this chapter. Together these allow both the theoretical and practical aspects of the framework to be subjected to an initial assessment. Whilst the depth of this initial assessment is limited, this shortcoming will be addressed later in this thesis in the context of a larger case study.

Before considering the features and application of the framework the first step is to understand its motivation and origins.

2.1 Motivation and background

The timebands concept is presented by Burns *et al.* [11]. The motivation can be seen in the multi-faceted challenge of constructing large, complex socio-technical systems — they note that *"the complexity of such systems makes all stages of development (...) subject to failure, resulting in costly re-work"*; they also observe that *"the production of an unambiguous behavioural description of an existing system is far from straightforward"*.

Observing that such systems are required to function across many different timescales, from microseconds or less, to hours or more, they assert that time is a crucial notion in the specification (or behavioural description) of such systems. Whilst the functional structure of the systems is generally possessed of a number of layers of abstraction, they note, however, that time is usually represented as a single flat physical phenomenon and suggest that this abstraction is deficient in three ways; it:

- Fails to support the structural properties of the system;
- Forces different temporal notations onto the same flat description;
- Fails to support the separation of concerns that the different timescales of the system facilitate.

They go on to assert that:

> *Just as the functional properties of a system can be modelled at different levels of abstraction or detail, so too should its temporal properties be representable in different, but provably consistent timescales.*

Following on from these observations and the assertion they propose a framework that explicitly identifies a number of distinct time bands in which the system being studied is situated; the aim being to produce more dependable computer-based systems through a better use of "time". Their view is that such a framework enables both the temporal properties of exiting systems to be defined, and those of new or modified systems to be specified.

The concept of a time band is attributed to Newell in his work [70] on human cognition. Considering the hierarchical structures of the brain Newell notes that different timescales are relevant to different levels of the hierarchy. By way of contrast the Burns *et al.* approach places the time band at the centre of their framework so that it can be used with any organisational scheme or architecture. They note that the number of bands and their specific granularities will be system-specific, but they contend that the relationships between the bands exhibit important invariant properties.

2.2 A review of the associated literature

As noted above Newell [70] is cited by Burns *et al.* as being origin of the concept of the time band, thus this is perhaps the best place to start any analysis and assessment of the concept of a banded notion of time.

Newell starts from the position that intelligent systems, by necessity, are built up from multiple levels of systems — a system level being a collection of components that are linked together and interact, thus producing a behaviour at that system level. In a hierarchical systems the components at one level are realised by subsystems at the next lower level, and so on; he identifies two "pillars" that support this view. First there is the empirical evidence regarding the engineering of systems to obtain intelligence, which indicates that a building up of these systems through multiple layers is required. The second comes from the analysis of hierarchy by Simon [98]. Simon states that stability dictates that systems need to be hierarchical, and that attempting to build hierarchical systems without first building sub-assemblies will ultimately fail — this is what is generally observed across all fields of engineering.

As an example of such a hierarchy, Newell suggest the structure of matter: from nucleons, to atoms, molecules, macromolecules, cells and so on. He distinguishes between strong and weak hierarchies: the former being where the is little or no interaction between the levels; whilst in the latter there is significant interaction between levels. He notes that, in engineered systems, care is taken to create strong hierarchies — ones where coupling between the layers is limited, and cohesion within each layers is strong.

Simon suggests that the linkages between the levels of the hierarchy are more complex than their obvious decompositions, and that other links also exist; he uses the example of the hierarchy of a human organisation where there are many more inter-relationships than are defined formally. In engineered systems, the hierarchy is usually largely functional or physical, it is interesting to debate if this view could be extended to consider the issues of non-functional interaction between the levels. He argues that *"hierarchy is one of the central structural schemes that the architect of complexity uses"*. Thus, as temporal issues are one of the prime causes of complexity in real-time embedded systems, the belief that providing a temporal hierarchy for such complex systems appears entirely justified — the only question might be: why had this not been previously recognised?

In considering hierarchical decomposition Simon notes that the hierarchy decomposes down to some lowest level of elementary subsystems and that, for most systems, this lowest level is somewhat arbitrary — there is no fundamental principle guiding the depth of decomposition. This really indicates that it is necessary to identify the bounds of the system from the outset, and define the point below which no further decomposition is necessary. These bounds will be dependent not only on the particular system, but also the reasons for capturing the hierarchy. In association with hierarchy he also considers the issue of *span* — the number of subsystems into which a system is partitioned at the next lower level in the hierarchy. The higher the *span* the flatter the hierarchy and, potentially, the bigger the gaps between the levels; it seems probably that high levels of *span* will lead to high levels of coupling between the levels, and poor cohesion within the levels — though this is not supported by any explicit evidence.

Newell notes that as one moves up through a hierarchy the size of the components increases and their "speed" decreases — as the size of the component increases so too does the time it takes to produce a response to some input. Using the example of the increasing response times of atoms, molecules, macromolecules, cells and so on, he suggests that the scale is geometric with the appropriate scale unit being the logarithm of time. From this general observation of temporal expansion he considers the temporal scale of human action. He defines a series of levels, each separated by a factor of ten, and groups these levels into bands, where each band contains three unique adjacent levels.

The table of Newell's levels and bands [70, p. 122] is reproduced in Table 2.1; in addition to the bands shown in the table he suggest that there are both other higher and lower bands. The higher bands are those above the Social band; these are a Historical band considering times up to millennia, and an Evolutionary band containing times of up to a million of years. At the other end of the spectrum there are lower level bands below the Biological

Scale (sec)	Time units	System	World (theory)
10^7	months		
10^6	weeks		**Social**
10^5	days		
10^4	hours	Task	
10^3	10 mins	Task	**Rational**
10^2	mins	Task	
10^1	10 sec	Unit task	
10^0	1 sec	Operation	**Cognitive**
10^{-1}	100 ms	Deliberate act	
10^{-2}	10 ms	Neural circuit	
10^{-3}	1 ms	Neuron	**Biological**
10^{-4}	100 μs	Organelle	

Table 2.1: Time scale of human action (from [70])

band, a possible example being that of the Macromolecule with a temporal range of 10^{-5} to 10^{-7} seconds. It is worth noting Newell's view that, in order for the system levels to exist they must be relatively strong, and that if the explanation of phenomena in one level requires detail from a lower level decomposition then it is doubtful that the higher level should indeed be considered a level. When considering the higher bands, to some extent the Social, and certainly the Historical and Environmental bands, he does not believe that there is adequate empirical evidence to support their existence.

This is perhaps a little harsh — as the levels expand the complexity also increases, the logical decomposition of the Evolutionary band is not simply human-centric, and the layers below it may not be the same for all different entities. One would suspect, for example, that the cognitive or rational dynamics of the fly would be different from those of a human. The lack of evidence does not prove that the higher bands do not exist, it is simply that at these higher levels temporal issues become very complex, and thus the evidence to support the existence of the bands may be difficult to obtain.

Newell uses empirical evidence to support the definition and existence of the Biological band with levels that have accepted names and definitions; he uses this and available empirical evidence to underpin his prediction for the levels of the Cognitive band, which is his main focus. Newell notes the neural system does not have sufficient time to produce a fully cognitive response, which begins to become apparent in the order of seconds; he

attributes this to there being a real-time constraint on cognition, which was previously noted by others:

> ***Real-time constraint on cognition:*** *The principle is that there are available only ≈100 operation times (two minimum system levels) to attain cognitive behavior out of neural circuit technology.* (Newell [70, p. 130])

Whilst Newell makes a strong case for the levels in his Cognitive bands in the higher bands the distinctions between the levels is less developed. The Rational band relates tasks performed using some "knowledge" or skill, the time taken for such tasks is more determined by the size of the task and the required outcome, rather than simply by the time to perform some basic function. It appears at these higher levels that the case for the levels, and hence the bands, is perhaps less justifiable.

As the basis for the higher bands is less than rigorous, then it is perhaps also the case that there is a reduced justification for the theory that these levels and bands are (totally) ordered and distinct. It does not seem unreasonable to assume that there could be some overlap between the temporal range of tasks performed in the upper reaches of the Rational band, and activities that might occur at the lower end of the Social band, assuming that the Social band extends down to tasks undertaken by groups (e.g. a football team). If the higher bands are considered more broadly across societies and cultures the strengths of the levels and bands are clearly weakened; Levine in [54] provides a number of examples of how problems can arise where there is a "temporal mismatch" between cultures.

For all of the possible limitations in the upper bands, the model presented by Newell strongly suggests a hierarchical approach to treatment of time in complex systems. Perhaps what is clear is that the definitions of the bands need to relate to the capabilities of the technologies present in the band — thus whilst the Biological and Cognitive "fixed" by human "technology", other bands, both higher and lower, will be dependent on the system context.

Returning to consider the differences between Newell's approach and the timebands framework the major points are that:

- Newell strongly relates time to the size of the structure being considered, whereas the timebands framework has no *a priori* view of size relative to time, or belief that the two are inherently linked;
- The fixed and strictly ordered levels of the bands of Newell's model are replaced by

a more flexible approach allowing for an arbitrary set of partially ordered bands, the characteristics of which are dictated by the dynamics of the system being described;

- Newell uses the terms time scale and band, the framework uses the single notion of the band to represent a distinct temporal level in any system;
- Whilst Newell uses a single physical measure (seconds) to define his bands the framework places no such requirement.

Moving on from Newell to consider other aspects to the background of the timebands framework, Burns *et al.* [11] note that large computer systems exhibit dynamic behaviour on many different levels. They observe that, whilst computational components operate at speeds where the durations their operations are measured in nanoseconds, the environmental damage that might result from the failure of some systems could persist for centuries. They suggest that, being able to accommodate such a wide temporal range requires a framework of considerable descriptive and analytical power.

In most cases where there is an attempt to identify time granularity, they note that the formulations map all activities to the finest granularity present in the system (or some fine grained universal reference such as a Cesium atomic clock). They assert that this approach results in cumbersome formulas, and fails to recognise the distinct role that time plays in the structuring of the system. They identify the work by Corsetti *et al.* [20, 14] which identify *"a finite set of disjoint and differently grained temporal domains"*, as being an exception. They consider Corsetti's framework to be less extensive than their own, but note that it does show how the notion of temporal domains may be embedded into a logical specification language.

Bettini *et al.* [7] note that the first formalisation of a time granularity system is probably that described by Clifford and Rao [17]. The approach in this paper differs from that of both Burns and Corsetti in that there is a total ordering over the granularities — an approach closer to that of Newell. Total ordering is common to other early approaches to granularity such as those described by Dean [25], and by Montanari *et al.* [67].

Whilst there considerable literature on the subject of temporal granularity, it is apparent that there is very little evidence of the multiple granularity models being applied to anything other than the simplest or most specialised of systems, and certainly not ones where the temporal domains are imposed by the technologies used to construct the systems. Other temporal logics have been developed but these are not seen as being relevant to the timebands framework, as, generally, they do not promote or support the concept of related temporal granularities that are the apparent strength of the framework.

2.3 A description of the framework

In describing the framework there are several key aspects to be considered; these are:

- Bands
- Entities
- Relationships

Whilst these are all described in the original report [11], for completeness each of these is also described in the sections below. A formal definition of the framework is also provided in the original report but this is not reproduced here.

2.3.1 The bands

Fundamental to the framework are the bands themselves and there are three aspects to the band that need to be understood; these are:

- What is a band?
- What happens within a band?
- What are the interactions or relationships between the bands?

Ultimately the bands only have a meaning when considered in the context of things occurring within them, these occurrences are described using the concepts of the event and the activity. For now the events can be considered to be things that have no discernible duration within the band, whilst the activities have some duration — these entities will be considered in more detail later.

What is a band?

The band is defined in terms of two fundamental concepts:

- A granularity (G), expressed as a unit of time that has some meaning within the band; it is the rate at which a nominal clock for the band ticks;
- A precision (ρ), a measure of accuracy of the timescale defined by the band, it defines the maximum separation between two events that can be considered to occur nominally simultaneously within that band; the precision of the band is the precision of the clock used to measure time in that band.

Within the framework there is no concept of the upper bound of a band, unlike in Newell's model where there is the concept of the scale of a band.

Behaviours within a band

Activities are placed in a band if they engage in significant behaviours within the timescale of that band — they have dynamics that give rise to changes that are "observable" within the granularity of that band.

The bands can accommodate both discrete and continuous views of time, both of which are needed to capture the temporal properties of complex systems — nature, after all, is not a discrete system. The band defines a temporal context (its granularity) into which the discrete actions may be placed, whilst the band can also accommodate continuous processes if they produce, or respond to, important observable events within its temporal context.

As noted above, within a band there are activities and events; activities are performed by some form of "agent", which might be human or technical, the kind of agent will be dependent on the parameters of the band. In very fine grained bands the agents will be exclusively technical, in some very coarse grained bands they will be exclusively human or societal; there will also be bands in which a mixture of agents are present. Clearly for any band where human agents are involved its definition, granularity and precision, relies on studies from the areas of the psychology and sociology of time, examples being Seow [94] and Levine [54] respectively.

Relationships between bands

Figure 2.1 shows a simple banded model and illustrates how the relationships between the bands may be formed. Events, which are instantaneous, in a given band (B) may either originate in the environment (i.e. their origins are outside the scope of the model of the system), or they map to activities in a lower, finer grained band (C). Taking the latter case, for an activity in band C to map to an event in the band B its duration (or clock ticks in the lower band) must be less than the precision defined for the higher band (B). As an example of this event-activity mapping, in Figure 2.1 event E in band B maps to activity Z in band C.

Considering a yet higher band (A), it can be considered to be constant for issues of concern in band B; any activity in band A will only exhibit an observable "event" for any activity in the finer band B. Likewise, behaviours in the finest band C are considered to be instantaneous in band B. This simple mapping assumes that the bands are totally or-

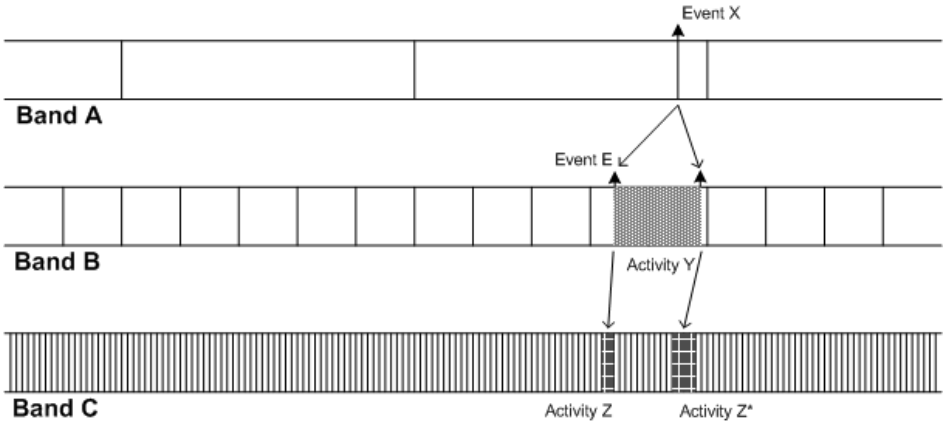


Figure 2.1: Time bands example

dered, and the durations of the activities in the lower bands are less than the precision of the immediately higher band.

Unlike Newell's model where the bands are defined to be totally ordered and have regular fixed scales (granularities), the timebands model imposes no such constraints. The relationships between the granularities of each band depend on the characteristics of the bands themselves, with typical ratios being between 10:1 and 100:1, thus the distinction as to in which band an activity should be placed is not necessarily clear cut. It is not necessary for an activity in one band to map to an event in an adjacent higher band — the precisions of each band play a key role here in determining the possible mappings.

2.3.2 Component elements — activities and events

As noted earlier the timebands framework defines two closely related component elements: the activity and the event. These, together with the bands themselves, are the fundamental elements of the framework.

The activities denotes items of "work" to be undertaken by some agent, the duration of the activity is the number of clock ticks of the band in which it is defined; the activity is bound to that one band. The event is defined formally as being a special case of the activity, one with a duration of zero, when measured in the granularity of the band in which it appears. However, the concepts associated with the event and the activity are in some ways really quite different. Rather than being some item of work the event can be view as the observable effect of some process. Thus events, other than those originating in the environment, can be thought of as being the result of an activity, or its observable state change; the point at which the change becomes observable is dependent on the detailed behaviour of the low-level activity. In fact activities are themselves "defined" by two events (in the same band as the activity), a start event and an end event; the event

corresponding the start of activity Y being denoted by $\uparrow Y$ and that corresponding to its end by $\downarrow Y$.

Perhaps what is less clear is whether the duration of an event should be less than the precision of the band rather than simply the granularity — the constraint on the duration being less than one, i.e. zero. If the duration has to be less than the precision then it raises the question: how is "something" with a duration of between the precision and the granularity represented within the band? The answer is quite simple: the entity is in the wrong band. The framework demands that the duration of the event is indeed less than the precision of its band.

2.3.3 Temporal relationships

There are two types of relationships to consider: those within a band, and those between bands. The first of these is concerned with the issues of precedence, equivalence and causality, whilst the second relates to the mapping between elements located in different bands — each is considered below.

Precedence, equivalence and causality

The framework defines the related concepts of precedence, equivalence and causality. Within a band, two events can be said to have a precedence relationship if one is defined to occur before the other. This in itself, it should be noted, does not imply the earlier event was the cause of the latter; equally temporal separation between events does not imply precedence.

Two forms of precedence are defined: strict and weak. For strict precedence it is required that there is separation between the events, but not one that is necessarily measurable; for the weak form the only requirement is that the second event does not occur before the first. The notations for strict and weak precedence for the events A and B where A precedes B are given in the equations below:

$$A \prec B \qquad (strong) \tag{2.1}$$

$$A \preceq B \qquad (weak) \tag{2.2}$$

Closely related to precedence is the concept of equivalence, in this case the pair of events exhibit mutual weak precedence; thus:

$$A \preceq B \wedge B \preceq A \Leftrightarrow A \equiv B \tag{2.3}$$

It is important to note that precedence implies nothing about temporal separation in any band. It is possible for events between which there are precedence relationships to be

separated by less that the precision of their band, i.e. temporally they are simultaneous in that band.

Precedence can give rise to causality, but does not necessarily imply it. If event A precedes event B then information may flow from A to B, and B may be caused by A, but the simple notion of precedence is insufficient on its own to infer causality.

Precedence is only defined to be between events — there is no direct concept of activity precedence; it is however possible to create such relationships by reference the start and end events of the activities. Thus for activity A to precede activity B totally it is necessary for the end event of A to preceded the start event of B, i.e. $\downarrow A \preceq \uparrow B$; through reference to these events it is possible to specify any form of relationship between the activities.

Inter-band mappings

The concept of event precedence is bound to a single band, it is not possible to directly specify the precedence between an event in band A and one in band B; in order to form such relationships inter-band mappings are required.

The fundamental approach to relating the behaviour in one band to that in another is the *Mapping*. A member of the *Mapping* associates an event in one band with an activity (or event) in a lower band; the mapping of the start event of a clock tick in a higher band, to an activity in a lower band leads to the precision of the higher band being defined by the duration of this activity — hence the precision is a property of the relationship between two bands; however this is not captured in the formal definition.

Another relationship between bands is that of a granularity mapping — this defines how the clock ticks in different bands are related. It is not necessary for this to be a simple scaled mapping; Burns *et al.* use the example of the mapping between a day band and a (calendar) month band where there is a set of possible mappings, i.e. $\{28, 29, 30, 31\}$. These allow the times in different bands to be related, though the issue of precision also has to be considered.

When considering the precedence relationships between events in a higher band these events often map onto activities in lower bands; where the bands are ordered (at least partially) any precedence or causality present in the lower bands is reflected into the higher bands. Where events in the higher band occur at the same time but they are not related through a precedence relationship no causality may be inferred; in this circumstance, the corresponding activities in the lower band will be logically concurrent, and may occur in any sequence or be overlapped.

It appears inevitable that the bands for the model of any complex system, which is built from a variety of technologies, will, unlike Newell's, have undefined temporal regions (gaps between the bands), or bounds that overlap to some degree. Such overlaps can lead to ambiguities as to in which band the artefact should be placed, and complicate the issues of inter-band mapping, especially between adjacent bands — thus it seems inevitable that excessive overlaps and the resultant potential for ambiguities will weaken the model.

2.3.4 Clocks

Each band defines at least one abstract clock; this defines the temporal frame of reference within that band; each clock ticks and records the passage of time within its band. Multiple clocks are allowed in a band, one possible use of this, suggested by Burns *et al.*, is to allow the mapping to different time zones. Where multiple clocks are defined for a band they must all progress at the same rate.

The clocks can be used to identify the time of each occurrence of all of the events within the band, the precision of the clock defines the precision of the band; thus allowing a time ordering of events to be defined — however it should be remembered that this time ordering does not imply precedence.

2.3.5 Behaviours

Behaviours are not a fundamental part of the framework, instead they are collections of activities and events within a single band. These collections may be partially ordered through precedence giving rise to sequential and parallel compositions. Behaviours may also contain behaviours thus providing some hierarchical structuring within a band; and two behaviours have a precedence relationship if they contain events that are ordered by precedence.

The behaviour is only briefly mentioned in the original report, and no examples of their use are provided.

2.4 Timebands applied

In order to give a more concrete understanding of, and further insight into the utility of the framework in this section it is applied to produce a timebands model of a simple example system that spans a number of temporal scales — the system is that of a bank cash dispensing machine, or automated teller machine (ATM). Before creating the actual model the system is scoped and described.

2.4.1 Description and scoping

Whilst in reality a bank ATM is quite complicated the basic functions of which most users are aware appear quite straightforward — the user enter a card and a PIN, and requests an amount of money, which hopefully the machine dispenses. Clearly the ATM interacts with a large number of other systems, for example the central accounting computer system of the bank(s). Including these systems would result in a very complex model, thus the first step is to identify the boundary of the system.

Physically the system comprises the "hole-in-the-wall" machine present in many locations, and with which most people are familiar; thus is consists of a store of money, a user interface, a local embedded computer system, a smart card reader, and an electro-mechanical money dispenser. Of course this is a fairly boring and inactive system if there are no customers, so these too are considered to be part of the system. For simplicity the maintenance operations (servicing and money loading) are not in the scope of the system and can be considered part of the environment. Whilst not a hard real-time system in the strictest sense, there are real-time expectations from the customer, and for any communications between the ATM and the wider ATM and banking networks.

A fundamental part of the defining the scope of the system is the identification of its interactions with its environment (the "use" cases), when the time bands are defined each of these interactions will be mapped to its appropriate band. These basic interactions, or "use cases", are:

- Account authorisation request;
- Account authorisation response;
- A request to withdraw money;
- A withdrawal confirmation.

In order to keep the model of the system relatively simple the possibility of using other common services such as PIN-change, phone top-up and balance requests are not considered, as too is the option of having "offline" verification of the request at the ATM. A simple timeline of a customer's interactions with the ATM is given in Figure 2.2 — this too represents a simplification of the system in that ignores issues such as: the inconvenience of the request being rejected; the timeout of user input; and failed communications with the bank.

The figure identifies a number of basic "tasks" for the customer and the ATM and a number of interactions between them, it also identifies some ordering, and precedence

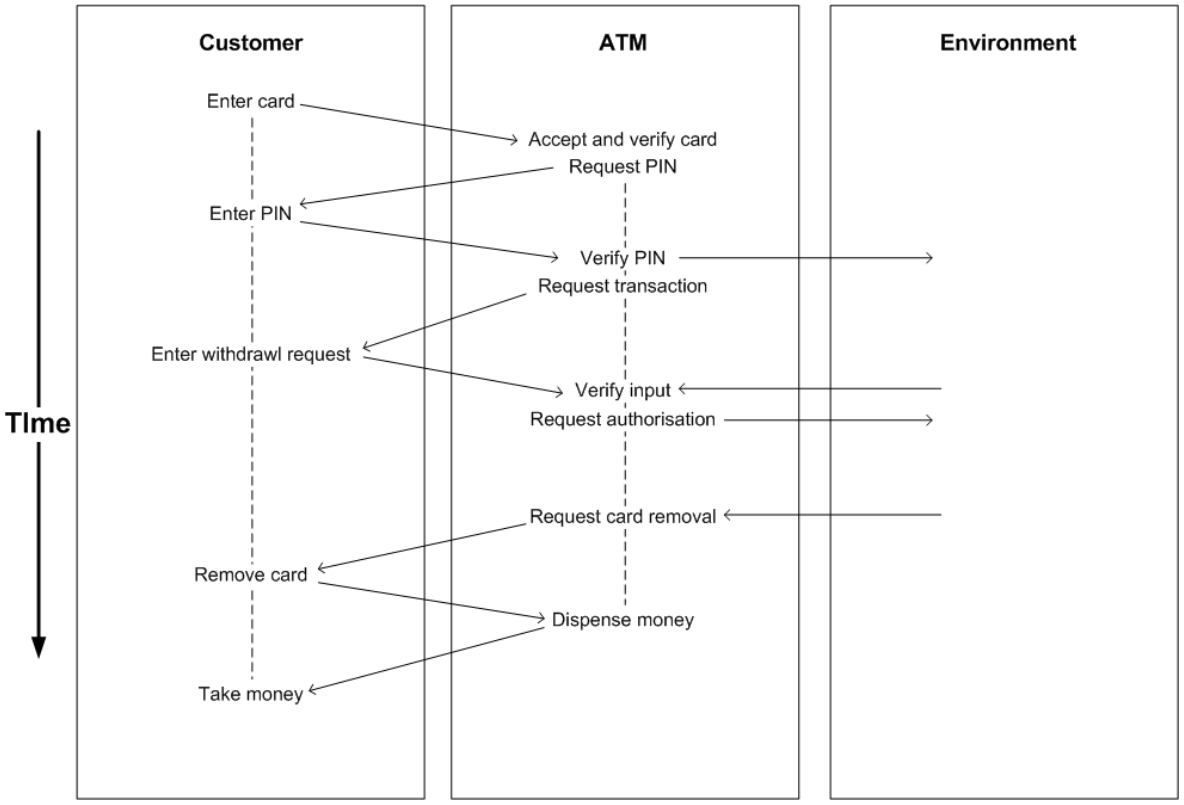


Figure 2.2: Timeline for the ATM cash withdrawal transaction

can be inferred from this. What it does not do, clearly, is identify the timescales of any of the tasks to be performed, or how long the interactions take, or are allowed to take.

2.4.2 Constructing the model

The are several steps in the construction of the model, these are:

- Identifying and characterising the bands;
- Populating the bands with the fundamental events — generally these are from the environment;
- Identifying the relationships between the fundamental events;
- Defining the activities and the inter-band event-activity mappings, ensuring precedence relationships across the bands are maintained.

The execution of each of these steps is described in the sections below.

Identifying and characterising the bands

Identifying the number of bands and the characteristics of each are the fundamental step in constructing a timebands model; however in order to identify the bands it is first necessary to define the bounds, or bounding bands, of the model. In this simple case the upper

band (the ATM System band) is one in which the overall withdrawal transaction appears as an event, i.e. the numerical value of the precision of the band has to be greater than the duration of the transaction, thus a value of two minutes does not seem unreasonable. The granularity in this band is of little consequence as we are only concerned about events and not attempting to define the duration of activities, thus a somewhat arbitrary value of one hour will be used. Of course it should be noted that if the context of the system were to be extended, to include for example periodic maintenance, then activities might be defined for this band and hence the granularity becomes significant.

At the other bound of the model the lowest band has to be defined. In this case it is assumed that the fastest component in the system is the local embedded processing system; however, for this model it is not necessary to consider the capability of the hardware, instead the characteristics of the software can be used to set the band parameters. The value of the granularity is somewhat arbitrary but a value of 10 microseconds is not unreasonable; as there are no lower bands with which to define mapping relationships, the precision of this band has no importance in the case of this model and is thus left undefined.

The other bands in the system are largely defined by the capabilities and behaviours of the other element of the system. There is clearly considerable interaction between the customer and the ATM, this probably warrants at least one band. Whilst there is the overall transaction with the machine, there are also lower level operations such as the inserting of the card or the typing of the PIN; the question is can both the overall and the "atomic" operations reside in the same band? Following from Newell's definition bands for the timescale of human actions (given in Table 2.1), a reasonable definition of a band to encompassing all of the human activity would comprise the *operation, unit task*, and lowest of the *task* levels, i.e. taking part of the Cognitive band together with the lower part of the Rational band. This would give the band the characteristics of a precision of 100 milliseconds, and a granularity of one second. An alternative approach would be to define this as two distinct bands using Newell's definitions for the Cognitive and Rational bands directly, this however seems unnecessary for this simple example.

The next question is: how many more bands are required? So far the definitions of the bands has considered both the embedded processing capability and the human interactions, thus two elements of the system remain: the electro-mechanical elements (card reader and money dispenser) of the ATM, and the communications interactions with the environment (the central computing system). Taking the first of these, the electro-mechanical band, the operations performed are: read card, eject card, count money, and

eject money; these are all relatively short lived with each operation taking at most a few seconds. A band with a precision of one millisecond and a granularity of 100 milliseconds seems appropriate. As the communications interactions with the central computer are strictly in the environment the durations of these are not of particular concern in this model, and hence it is appropriate for all of the interactions to be mapped as events in the appropriate bands. To some degree this indicates that the system should "protect" itself against unexpected (untimely) behaviours in the environment.

Finally there is the overall transaction to consider: does this require its own band? The transaction is a higher level (slower) process than the individual human tasks captured in the customer band, and are thus not in the scope of that band. Equally the highest band will only capture the overall transaction as a single event, it is not sensible (or possible given the precision) to provide more detail at that level. In the light of this an additional band for the overall transaction is defined; its precision is five seconds, and its granularity one minute. In the context of the limited environment of this model the need for this band might be questioned; however if there was a desire to expand the model to address a wider system context including, for example, maintenance activities then this band, or a similar one, would certainly be required.

The five bands and their characteristics are summarised in Table 2.2.

Band	Granularity	Precision
ATM System	1 hour	2 mins
Transaction	1 min	5 secs
Customer	1 sec	100 ms
Electro-mechanical	100 ms	1 ms
Embedded computer	10 μs	–

Table 2.2: ATM band characteristics

Populating the bands with the fundamental events

In order to populate the bands the first step is to identify the events. In this case the fundamental events can be derived from the timeline diagram given earlier in Figure 2.2; these reside in the Transaction band where they represent an interaction is with customer, and in the Embedded Computer band where there is some "low-level" interaction with the environment. The other "obvious" event is that of the overall transaction, which resides in the ATM System band.

Additional events can be postulated for the other bands: the customer pressing a key as part of the PIN entry, and the low-level stimuli from the embedded computer that cause the card or the money to be ejected by the various electro-mechanical devices. These are not considered for now.

The locations of the events in the various bands are given in the Table 2.3 below, the identifiers are provided for convenience and are used in later descriptions of parts of the model.

Event	Band	Identifier
ATM transaction	ATM System	E0
Insert card	Transaction	E1
Request PIN	Transaction	E3
Enter PIN	Transaction	E4
Offer withdrawal	Transaction	E6
Select withdrawal	Transaction	E7
Eject card	Transaction	E8
Remove card	Transaction	E9
Dispense Money	Transaction	E10
Take Money	Transaction	E11
Verify account	Embedded computer	E12
Request authorisation	Embedded computer	E13
Account verified	Embedded computer	E14
Authorisation provided	Embedded computer	E15

Table 2.3: Event-band mapping

Identifying the relationships between the fundamental events

Taking the events defined in Table 2.3 their inter-relationships can now be defined. On the whole these follow from the timeline provided in Figure 2.2, and are thus relatively straightforward. In general there is a strict ordering of the events in the Transaction band, these are defined in the precedence relationships given below; as there is only one event in the ATM System band no precedence relationships can be defined for this band.

$$E1 \prec E3 \prec E4 \prec E6 \prec E7 \prec E8 \prec E9 \prec E10 \prec E11$$

At the precision of the Transaction band (five seconds) the ordering of the events is clear in an overall sense, i.e. there is a distinct separation between the insertion of the card (E1)

and the receipt of the money (E11), but there is a lack of discernible temporal ordering with regard to certain parts of the sequence. The time between, for example, entering the PIN (E4) and the offering the withdrawal service (E6) will probably not be indiscernible in this band.

The events that denote the interaction between the embedded computer system and the environment (E12 to E15) also exhibit precedence relationships; these are:

$$E12 \prec E14 \tag{2.4}$$

$$E13 \prec E15 \tag{2.5}$$

The precedence relationships defined for the transaction band above do not capture the issues relating to the interaction with the central computer system with regard to verifying the card and authorising the transaction; these are internal events and become apparent as the model is developed.

Defining the activities and activity-event the inter-band mappings

The first activity to be defined is the one in Transaction band that maps to the overall event ($E0$) in the ATM system band, this will be named "Cash transaction" or A0. The bounds of this activity are the events $E1$ and $E11$; there is an equivalence, $\equiv$, between the start and end events of A0 ($\uparrow A0$ and $\downarrow A0$ respectively) and the these two events, thus:

$$\uparrow A0 \equiv E1 \tag{2.6}$$

$$\downarrow A0 \equiv E11 \tag{2.7}$$

Defining the duration of this activity is somewhat problematic, the question is: what does the duration mean — is it some nominal value, the minimum or maximum, or something else? In the case of A0 the duration is going to be in the range $1 \ldots 2$ minutes — the meaning of the duration of an activity will be returned to later in this thesis.

Moving on to consider some of the events in the Transaction band and how they are mapped to activities in the lower bands. The Enter PIN event (E4) clearly maps onto some activity (A4) in the Customer band; entering the PIN requires the customer to enter four digits on the keypad — these correspond to four events in the Customer band: BP1, BP2, BP3 and BP4 . The overall duration of this activity will be a small number of seconds. The start of the activity being the first key press (BP1), and with the end being the last key press (BP4).

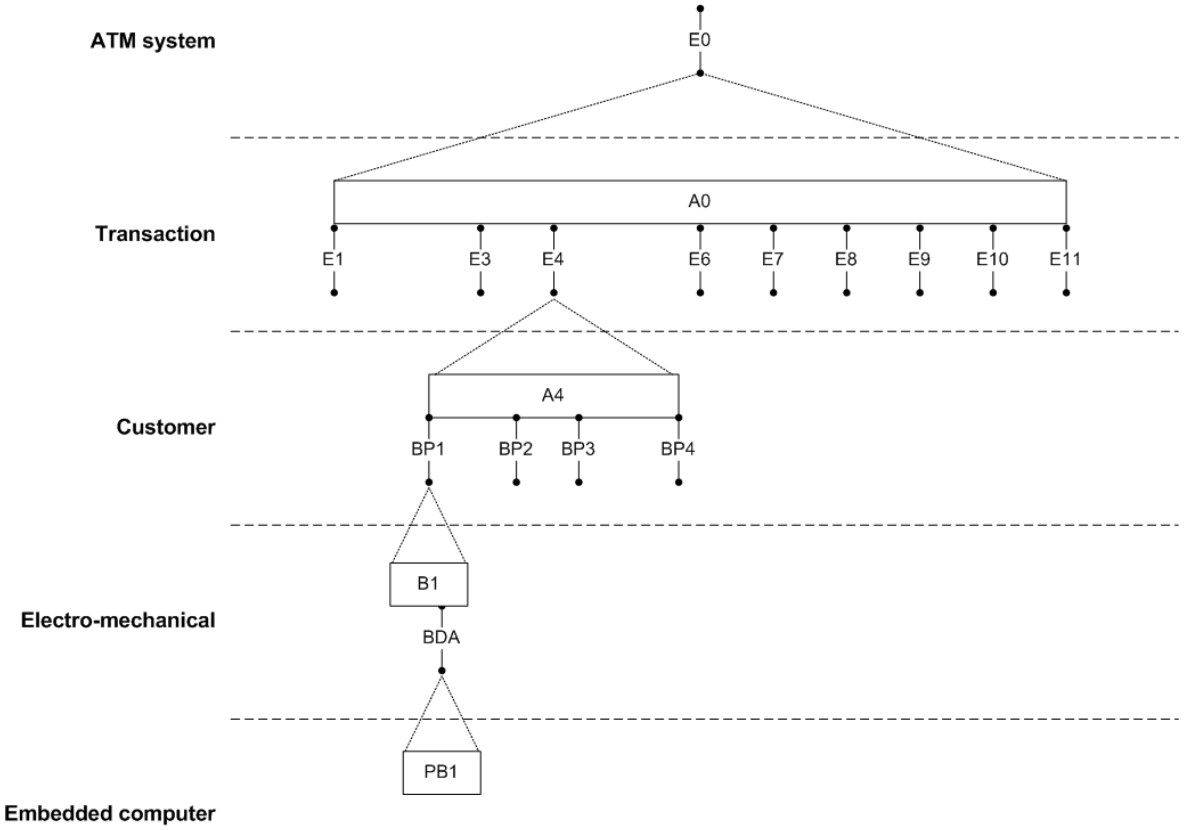


Figure 2.3: Event-activity relationship for a button press

Each of the button press events maps to a corresponding activity at a lower level, if the desire was to consider the Electro-mechanical behaviour then the button press could map to that band, otherwise the effect could be directly mapped to the Embedded computer band. "Skipping" the Electro-mechanical band indicates that the issues addressed in that band relating to the keys, such as key bounce, are of no concern in this particular model of the system.

Taking the exhaustive approach and mapping the key press first to an activity in the electro-mechanical band a short duration activity (B1) is created; in order for it to map to the event in the Customer band its duration has to be less than 100 ms, the precision of the Customer band. At some point within this activity there exists an event (BDA) that relates to the embedded computer using this data. This event is the mapped to an activity (PB1) in the lowest band, that of the Embedded computer. This activity could be the response to an interrupt and the validation of the data; as before the duration of this activity has to be less that the precision of the higher band, in this case less than one millisecond. The inter-band relationships described above are shown in Figure 2.3.

In the Embedded computer band there is an activity to verify the PIN input by the customer against that stored on the card; thus preceding the start of this verification both the card

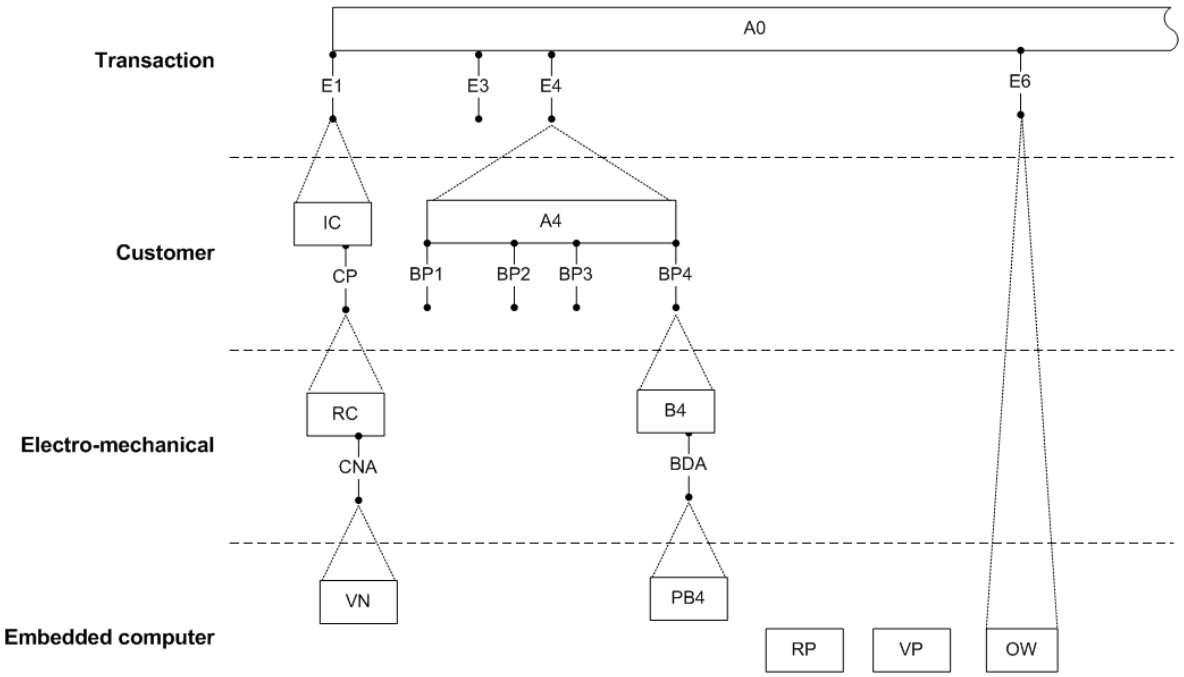


Figure 2.4: Event-activity relationships for the PIN verification

must have been read, and the PIN entered. The reading of the card is an activity in the Electro-mechanical band, the critical point of which (the event CNA) is mapped to an activity in the Embedded computer band that validates the number. Using RP, VN and VP to represent the activities Read PIN, Validate Number, and Validate PIN respectively, the precedence relationships for the PIN verification in the Embedded computer band are:

$$\downarrow VN \prec \uparrow VP \wedge \tag{2.8}$$

$$\downarrow RP \prec \uparrow VP \wedge \tag{2.9}$$

$$\downarrow PB4 \prec \uparrow RP \tag{2.10}$$

There is no precedence relationship between the Read PIN and the Validate Number activities however they both have a common origin in that the card has to inserted and be "valid". Subsequent to the PIN being verified the offer for withdrawal is presented, and is visible in the Transaction band as event E6. The relationships between the card being inserted and read, and the PIN being verified are shown in Figure 2.4.

Continuing from the verification of the PIN, there is a request to the central computer to confirm the account and the presentation of the withdrawal option to the customer. The processing of the withdrawal request will only proceed once the account has been confirmed; subsequent to this, the withdrawal amount has to be verified before any money is dispensed (E10). The relationships between these activities are shown in Figure 2.5.

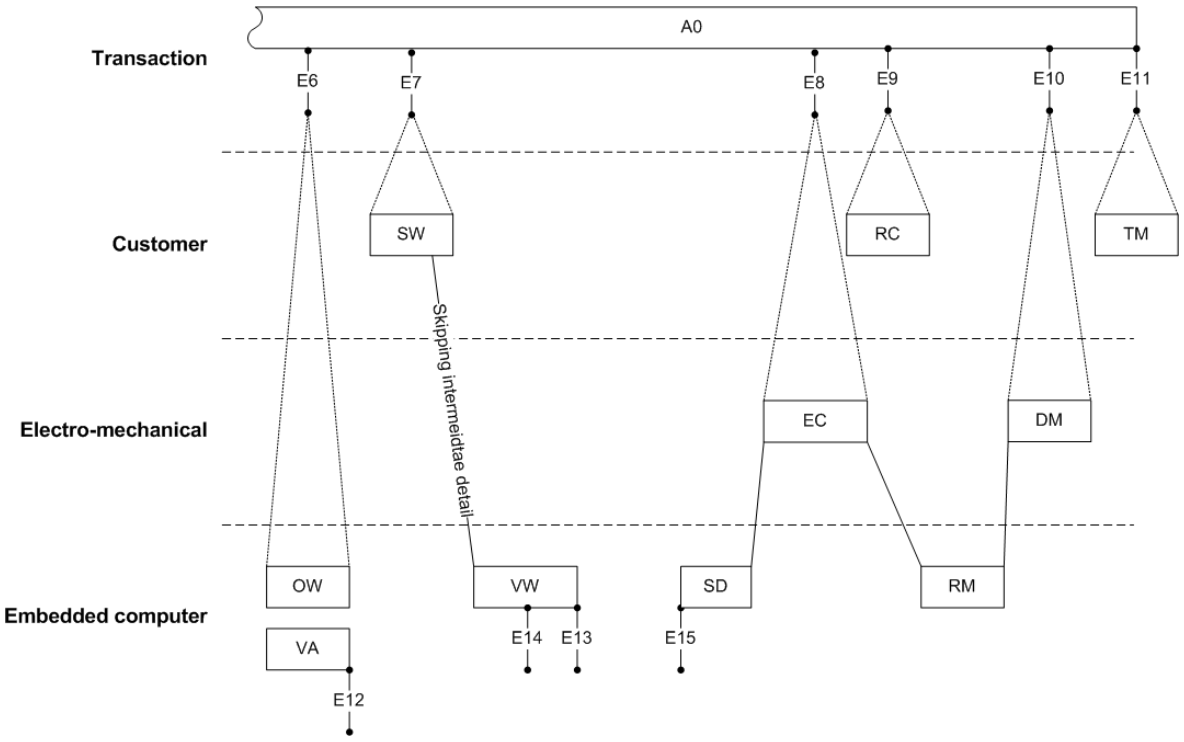


Figure 2.5: Event-activity relationships for the authorisation and dispensing of money

Preservation of precedence

Another aspect to be considered in the mapping of events in one band to activities in the lower bands is that of the preservation of precedence and causality. In this example, where the system is largely reactive, there are some obvious causal chains, e.g. entering the card ⟶ requesting a PIN etc, that largely follow the precedence relationships defined in the Transaction band.

If the events in the Transaction band are considered to be the "requirements" of the system then there may be strict relationships between them. At the lower levels there is no need to preserve the precedence to the degree such that in the case where E1 precedes E2, and these are mapped to activities A1 and A2, for the event denoting the end of activity A1 to precede the event denoting the start of A2. In this case all that is required is that the notional point in each activity that gives rise to the events in the higher band are correctly ordered — the limiting case for this to be possible is that the start event of A1 preceded the end event of A2, i.e. there is some degree of overlap between the activities.

Considering an alternative approach where the system is defined by the activities in the lower bands. Here, if the activities exhibit a strict ordering, such that the end of one precedes the start of the next, it is clear that any events due to these activities that become apparent in the higher bands will also be strictly ordered. In essence, anything stated in a lower band must be preserved in all higher bands.

Comparing these approaches it is clear that the former approach, of stating the requirement as a set of events in a higher band, provides significantly more scope for implementation flexibility. It thus appears to be crucial that these important events be placed in the highest band possible, allowing the maximum scope for flexibility in all of the lower bands — the issues surrounding the placement of external events are discussed in more detail in a later chapter. In the case of the simple model presented here, as shown in Table 2.3, a number of the events concerned with the interaction between the ATM and the central computer were placed in the Embedded computer band (the lowest band), the questions are: could these be placed in a higher band, and if so what is the effect on the model?

It is possible to place these events into a higher band, moving them to the Transaction band changes the precedence relationships in this band to be:

$$E1 \prec E3 \prec E4 \prec E6 \prec E7 \prec E8 \prec E9 \prec E10 \prec E11$$

$$E4 \prec E12$$

$$E12 \prec E14$$

$$E14 \prec E7$$

$$E7 \prec E13$$

$$E13 \prec E15$$

$$E15 \prec E10$$

These relationships clearly show the opportunities for parallel operations in the system — this appears to be an improvement in that such opportunities for the implementation are not immediately apparent in the earlier model.

Summary of the example system

The example system was chosen to be relatively simple, but equally to be one that would map to a number of bands; as such the example is limited in two respects. The first of these is that in this case the system already exists; whilst capturing the temporal dynamics is clearly a valid and useful application of the framework, it gives little indication as to how the framework might be applied during the development of a new system.

The second limitation relates to the style of the example system. In this case the system is a reactive one and as such it does not indicate how the approach would work with a more continuous control system, or one with persistent data.

It is apparent that it would be difficult to construct the model without a reasonably clear view of the functionality of the system, or indeed identifying the boundary of the system. Having a clear view of the sequences (in a reactive system), and being able to identify the interactions with the environment are a good starting point from which to create a timebands model.

The model created for this example system strongly suggests that, for a reactive system at least, the approach has considerable merit. The ability to simplify the temporal requirements to a set of "time-free" sequences of events, ordered by precedence instantly removes a significant source of over specification — the need to detail each step explicitly. The bands, in effect, define the temporal context for each of the sequences at the "correct" level of abstraction.

There is clearly the question as to whether or not the bands are the "correct" ones; as noted earlier, when the bands where being defined, other possible bands could be envisaged. Ultimately however the "correctness" of the bands will be dependent on the use to which the model is to be put, e.g. the form of analysis to be undertaken. In this example the bands chosen achieve their goal of showing how a model may be constructed. Whilst on the subject of the bands, the ones in this example are totally ordered and there is no overlap, thus the issue of the impact of overlap and partial ordering has not been considered.

2.5 An initial assessment of the timebands framework

A thorough assessment of the framework and its practical worth will be covered in a later chapter in the context of its application to a large case study; however there are some initial observations as to its perceived strengths and weaknesses that have been identified through an analysis of the literature, and its application to the example above. Given the limitations of the example there remain some areas of the framework that are unclear or unexplored. A discussion regarding these various aspects is contained in the sections below.

2.5.1 Strengths

The two main virtues of the framework are its simplicity and its flexibility. It is simple in that there are very few elements, and the relationships between the elements are generally quite clear. Its flexibility stems from the lack of constraints imposed on, and between, entities.

The provision of a formally defined framework generally helps to ensure that the terms are clearly defined, and in this instance the definition is small enough to be readily understood by non-formalists!

Fundamentally the framework appears to succeed in its goal of providing an approach to the partitioning of the dynamic behaviour of the system into bands of similar (temporal) dynamics, thus supporting the separation of concerns that is widely recognised as being key to good system structuring. As systems evolve, e.g. through obsolescence, elements might move from one band to another thus changing the fundamental dynamic relationships between components; a suitable timebands model should help to provide an early warning of such dynamic issues.

2.5.2 Weaknesses

Fundamentally the weaknesses in the framework can be seen across four areas:

- Guidance on how to select the bands and create models;
- Examples of use;
- Potential omissions and “ambiguities”; and
- Tool support.

The lack of guidance is of fundamental importance, the timebands framework is unlikely to be widely exploited if the is no guidance on its application. Whilst the framework has been defined for a number of years, documented examples of its use is not widespread, and these do not address the issues of band selection. The example in the original report [11] is the only significant published application of the framework and, whilst this is good so far as it goes, it does have some limitations. Perhaps the most obvious being that it provides the bands as a given rather than reasoning about their characteristics.

As noted earlier, the formal definition of the framework is quite simple, unfortunately however it does not appear to be complete. The most obvious omissions are:

- That of the concept of precision from the formal definition of the band; and
- The associated constraint that the duration of an event must be less than that of the precision of the band.

The range, or scale, of a band is left rather open-ended, thus it is not clear as to what the upper bound on the duration of any activity in any given band should be, possibly running the risk that activities will be defined in an inappropriately low band. Perhaps

as part of any model the bands should have upper bounds placed on them — this would highlight issues regarding the degree of overlap of, and gaps between, the bands. For the remainder of this thesis a “scale” property will be applied to all bands when models are created.

As noted in the creation of the model for the simple example system there is an issue with the meaning of the duration of an activity. The framework defines this has being a single value, but gives no guidance as to what this value should relate, e.g. maximum, minimum, nominal etc. Perhaps changing this to be a range encompassing all “normal” (i.e. non-failure) values would be an appropriate clarification.

The final area of weakness is the lack of tool support — as noted in Appendix A a lack of tool support was an impediment to the adoption of the early ADLs, and without such support it is difficult to envisage how complex models might be created and, importantly, verified and validated.

2.5.3 Unclear aspects

There are four areas where the framework is, to some degree, either unclear, or has not been investigated in any depth; these are:

- The issues regarding the definition of the granularity and precision of the bands and the relationship between these properties;
- The utility of the concept of the behaviour;
- The issues associated with creating models in which the bands are only partially ordered, or overlap in some way;
- The ability of the framework to support all of the notions for behaviour that might be expressed in complex systems. An example here would be the temporal aspects relating to data, e.g. the concept of freshness.

Some of these topics will be returned to in later chapters for further consideration.

2.6 Recent developments

More recently there have been a number of publications related to the timebands framework that follow on from the original technical report. These are (chronologically): Burns and Baxter [10]; Baxter *et al.* [6]; and Burns and Wellings [12];

The first two provide little beyond the original report. The last one alludes to the concept of state being included in the model, but does not elaborate on the concept, provide examples of its use, or formalise its definition.

2.7 Summary

The aim of this chapter was to explore the issues associated with the timebands framework through a review of the associated literature and its application to a simple example. The literature relating to timebands themselves is actually quite limited, however there is significant material on the subjects of hierarchy and the banded model of time, as well as other insights from psychology and sociology on the perception of "time".

From the literature and the example model the framework appears, in general, to be capable of capturing the temporal properties of a system. A number of weaknesses have been observed, none of which appear to be fundamental, or present a blocking issue with respect to the application of the framework to model a large system.

By its very nature the modelling of the simple example system presented above only explores certain aspects of the framework and its application; different types of system might exposes additional or different issues. Additionally no consideration has been given here to the application of the framework to a system at different points in its life-cycle, e.g. during its definition, through its development, or once in service. It seems probable that, unlike the example here whose bands are based largely on the technologies used, the bands for the model of a system constructed during its definition would be more closely related to the required behaviours. The problem of creating a model for a new system is addressed in a later chapter as part of a industrial scale case study.

The event and activity can be seen to map simply to functional elements common in traditional architectural and design notations, e.g. the interrupt or thread. There are however other concepts that are commonly used in functional design notations, such as mode and state, where their mappings to timebands concepts is less obvious. In the next chapter the relationship between mode and state is considered, and an approach to the incorporation of modes into the timebands framework is developed. Modes are chosen in preference to states as it appears, from comments by Burns and Wellings [12] noted earlier, that there is already some consideration of the inclusion of states within the timebands framework.

Chapter 3

System Modes: their Relationship to the Timebands Framework

In the previous chapter it was observed that system modes and states are concepts that do not immediately appear to map onto the original timebands framework. In this chapter the issues relating to system modes and their relationship to the timebands framework are considered in detail, and two approaches to the extension of the framework to enable their accommodation are proposed; in the following chapter one of these approaches is developed further and formalised. However, in order to introduce the extension clear understandings of the concept and nature of modes, and how they are distinguished from states are first required; thus the remainder of this chapter is divided into the following three parts:

1. A survey of the use of the terms mode and state, both generally within computer science and more specifically across a selection of ADLs;
2. A discussion on the nature of modes, and their relationship to, and differences with the concept of state;
3. An examination of the relationships between modes and the existing artefacts of the timebands framework, and the proposal of two contrasting approaches to the extension of the framework to accommodate system modes.

Parts 2. and 3. are supported by the use of simple examples.

3.1 The concept and nature of mode and state

The terms mode and state have been, and continue to be, used widely across computer science to describe some aspects of systems' organisation and behaviour. However, even

given this usage[1] there appears to be little clarity as to their respective meanings; consequently, the differences and relationships between the terms, and the particular utility of either term are not readily apparent.

Considering and understanding (or defining) the differences between modes and states is important if they are to be used appropriately in the context of architectural descriptions constructed taking a timebands centric perspective. Understanding how these terms are are used in ADLs is of interest in that it can provide insights as to the necessity, or otherwise, of accommodating modes in the timebands framework.

This section is divided into three parts:

- A general survey of the origin and use of the terms mode and state;
- A brief introduction to ADLs; and
- A discussion the use of the terms mode and state in the field of ADLs.

3.1.1 A survey of the terms mode and state

This survey is necessarily brief due to the limited and disjoint nature of the information available. The important area to be considered is the use of the terms mode and state in architectural description, this will provide an understanding as to how the terms are used in "functional" design and thus give some insight as to how they might be modelled in a timebands context. However, before considering the use of the terms with regard to architectural or systems description, it is first useful to understand the origins and use of the terms more generally across the field of computer science.

State

The term state has been in use since the early days of computer science, though its meaning and use have been far from consistent; equally, it is difficult to identify the first instance of any particular usage — though the early independent work by Moore [68] and Mealy [58] on finite-state machines, appears to be the fundamental origins of the majority of other work. So from the very beginning two fundamentally different views are present, depending on whether actions and outputs occur within states (Moore), or on the transitions between states (Mealy).

A major change in the way in which state was viewed came with the introduction of hierarchical state machines, this is generally attributed to work by Harel on StateCharts [38].

[1] Or perhaps because of the breadth of this usage.

This allowed more complex systems to be modelled using degrees of hierarchical abstraction; it also introduced the notions of concurrent (or conjunctive) states, and hence the ability of the system to be in a number of different states simultaneously. However Harel notes that his approach only defines certain aspects of a system, and that the approach was targeted at defining a particular class of systems — reactive ones. Interestingly this targeting at a particular class of system is one reason why Medvidovic and Taylor concluded that Harel's StateCharts were not complete enough to be described as an ADL in their work on classifying and comparing ADLs [62].

In general the term 'state' has been used in three basic contexts; these are to describe:

- A particular condition of the system, or a particular element of the system at an instant (finite states), and the organisation of such states and their interactions — finite state machines or automata;
- The state of a notionally continuously varying element of the system, or the environment with which the system interacts — continuous state;
- The totality of the conditions in which the system can exist — often described as the state space.

These all point to the notion of the *condition* of the system rather than that of its *behaviour* or *function*.

Clearly there are overlaps between the first two of the above points as the distinction between them is largely an issue of granularity and sampling fidelity. The concept of state space is related to that of finite states, again requiring the concept of granularity. The state space can be used either to show how the states of an individual machine are related, or to describe the totality of the possible states of a system built from a number of components — each with its own set of finite states.

Mode

The term mode came into use somewhat later than that of state, however in common with state its usage has been far from consistent. The initial use of the term seems to be generally acknowledged as being the work on the specification of software requirements for the A7e aircraft by Parnas *et al.* [40]. Their definition stated that a "mode class" was a partition of the state space of the system; it allowed systems to exist in (certain combinations of) multiple modes simultaneously. Later Jahanian and Mok attempted to clarify, or perhaps reinforce, this distinction between modes and states in their definition of Modecharts [49]; they suggested that states relate to the data in the system, whereas modes

relate to some aspect of the control of the system, i.e. some aspects of its behaviour. They also introduced a significant semantic difference between mode and state in that modes may change instantaneously, whereas states must exist in any condition for a finite period.

In more recent times things have not become much clearer, with different authors taking different approaches there is still no clear single view as the meaning of the term mode; in some cases the term is used to describe something that could equally have been defined as a state. Some authors (e.g. those of the SAE standard for AADL [90]) take the view that modes differ from states in that the change from one mode to another may be gradual (proceed over a period of time), whereas state changes are instantaneous. Here again this view is not universally held, and is clearly directly opposed to the view of Jahanian and Mok who, for Modechart, define mode transitions as being instantaneous — this lack of consensus indicates the importance of being clear about the these issue if modes are to be used in a timebands context.

3.1.2 An introduction to Architecture Description Languages

In order to be able to usefully incorporate modes (or states) into the timebands framework it is necessary to understand how the concepts are used and supported in the development of large-scale systems; to a large extent such an understanding may be drawn from an assessment of the support they receive in current "design" notations. In order to aid this understanding the following section provides a review of the support for the concepts across the field of Architecture Description Languages (ADL); however, before considering the specific support they provide in this area, it is useful to outline the basic characteristics of these languages.

ADLs are a class of languages developed to support the description of the architecture or high-level design of systems, and the combination of systems into larger systems-of-systems; as such they focus primarily on structural and "functional" issues, and are often domain specific. It is their focus on the large-scale systems and systems-of-systems that makes these languages of some interest in this thesis.

A wide range of ADLs have been developed over a number of years, these include: Wright [1], UniCon [97], Rapide [56], MetaH [103], Darwin [57], Modechart [49], and the UML [78]. The general styles the these languages cover quite a broad spectrum, ranging from ones that are implementation-independent, to ones that assume a specific model (or models) of implementation — and, hence, can be quite constraining. In some cases the languages address only the higher level structural issues, however, in others some of the fine grained

detail is also captured — in an extreme case, even down to the level of the software subprogram. A common trait across ADLs is the lack of support for capturing and modelling non-functional properties; considering specifically real-time systems, there is generally little support for temporal issues in anything other than a peripheral role; where acknowledged temporal information is simply added as independent properties to the fundamental elements, there being no overall notion of “time”.

An in-depth review of ADLs covering there origins, capabilities and current status is given in Appendix A.

3.1.3 The use of Mode and State in Architecture Description Languages

In the field of ADLs there appears to be a general trend of not considering the modes or states of a system as being of any particular importance or interest — the reasons for this are not immediately apparent. Considering a broad range of ADLs it is clear that there is little consideration of state in most of them, and the explicit use of the term mode is even more restricted. Table 3.1 identifies the use of the terms mode and state across a selection of ADLs and related languages.

ADL	State	Mode	Type	Reference
Wright	✗	✗	Implementation-independent	[1]
xADL2	✗	✗	Implementation-independent	[22]
C2	✗	✗	Implementation-independent	[101]
xACME/ACME	✗	✗	Implementation-independent	[93, 33]
Unicon	✗	✗	Implementation-independent	[96]
Rapide	✗	✗	Implementation-independent	[55]
UML	✔	✗	Implementation-constraining	[74]
MetaH	✔	✔	Implementation-constraining	[103]
AADL	✔	✔	Implementation-constraining	[90]
Ptolemy	✔	✔	Implementation-constraining	[47]

Table 3.1: Use of the terms mode and state in a selection of ADL-like languages

As can be seen from Table 3.1 there are only four ADLs that explicitly model state — MetaH, UML,[2] AADL and Ptolemy. Additionally there are only three that directly considers mode — MetaH, AADL[3] and Ptolemy; though it could be argued that some of the modal behaviour issues are addressed through the use of "Use Cases" in the UML. However this could only be considered to be a relatively weak approach as “Use Cases” do

[2]The UML encompasses derivatives such as SysML and Marte.

[3]This is not surprising as AADL has its origins in MetaH.

nothing to address the issues around inter-mode transitions (except perhaps in terms of pre-conditions over the use cases). As noted in the table, in general the languages which identify modes and state as important are those generally classified as being of the *implementation-constraining* group, possibly indicating that these concepts are of more interest at the more concrete end of the architectural spectrum. At the other end of the spectrum, the *implementation-independent* ADLs appear to universally fail in acknowledging the concept.

A possible rationale for this lack of explicit importance comes from the notion that the higher-level ADLs, those classified as being *implementation-independent*, are attempting to provide a language that is best suited for describing the *components*, their *connections* and their *configurations*. Often this seems to result in a lesser concern being given to the conditions, functions, behaviours and operation of the system — the focus being predominately concerned with its overall structure.

A question, which follows from this, is: if modes are not seen as being directly important is it possible that the behaviours of the system could be modelled using other elements of the language? One option would be to use the generally available (in ADLs) concept of the *configuration* to address the modal behaviours of the system. Whilst it is possible to conceive of ways in which this could be done it does appear to be stretching the abstraction a little too far — it would also lead to the overloading of the use of *configurations* in that they are also used to define structure through component topologies resulting in a significant potential for confusion, negatively impacting on the goal of separation of concerns. An additional issue inherent in such an approach would be the lack of consideration of the transition between the configurations, in particular their dynamics.

It is clear that, in one of the cases where mode is given some status, that of MetaH, there is significant effort to separate the concepts of configuration and mode. In MetaH there are two constructs: macros and modes; the basic distinction between these is that modes are defined as being mutually exclusive, whilst macros can exist/operate in parallel. The latter of these seems to be more akin to the normal notion of a configuration (a topology). In AADL (closely related to MetaH) there is no such distinction and only the term mode is used — here it is defined as being a "*possibly different configuration of components and their connectivity contained in the system*". This seems to be nearer to the general use of configurations in other ADLs, though here they remain strongly linked to the behaviour of the system, and unusually the semantics of the transitions between modes are defined quite precisely.

It is an interesting (but unanswered) question as to why, when modes have been considered an important (if ill-defined) aspect of the system description (for example by Parnas

et al. [40], and Jahanian and Mok [49]), do they not appear as specific elements in the majority of the ADLs? This potential omission seems to indicate that ADLs are generally weak in that they do not convey any significant information regarding the behaviour of the system; they appear to be mainly focused on its structure, and perhaps rely on this structure to convey aspects of behaviour. This lack of focus on behaviour contrasts somewhat with that of the timebands approach where behaviour is a central concern — this reinforces the view that modes should be accommodated in the timebands framework.

3.2 Clarification of the differences and relationships between the concepts

In order that both modes and states can be used clearly and correctly within the architectural descriptions of systems it is necessary to have a clear and comprehensible distinction between the two concepts — the purpose of this section is to attempt to provide that distinction. This distinction is developed through the use of an example system; whilst the example is small and relatively simple it is sufficiently rich to allow the fundamental issues relating to, and distinctions between, mode and state to be explored.

3.2.1 System description

The example is based around a model of a very simple lighting system. In essence there are a number of elements to the system:

- A light bulb;
- An associated switch;
- A power supply; and
- A maintenance policy.

Each of the first three elements has a set of states associated with it: switch may be *open*, *closed* or *us* (unserviceable) — the actual state of the switch, *open* or *closed*, in this failure case is of no concern for this example; the power supply may be *on* or *off* (here again, the reason for it being in the *on* or *off* states is of no concern for this example); and the light bulb may be either *ok* or *us*. Given it is known that the light bulb will eventually fail the maintenance policy requires that the bulb be replaced periodically, rather than waiting for it to fail — the aim being to reduce both the cost of running the system,[4] and

[4]Having an element of planned maintenance being a more economic approach than having a purely reactive, repair-on-fail policy.

the general disruption to users.[5] Given this policy, the light bulb has an additional state of *ex* (expired), which it enters when its "normal" life expectancy has been exceeded. The maintenance policy is a key element of the system, albeit one whose importance might not at first be recognised.

3.2.2 A simple mode model

Taking the elements of the system as described above a simple mode model can be proposed. In this case there are three modes; these are:

Operational The system operates on demand — the light bulb responds to the switch being operated;

Expired The bulb has reached its expected life and needs to be replaced (but remains serviceable);

Failed The system does not operate, there is a failure of one or more of the components.

The modes of the system, and the associated inter-mode transitions are shown in the diagram given in Figure 3.1. The operates-on-demand "transitions" are shown for completeness (to show the light is operational), though in reality these are not an inter-mode transitions as the mode does not change.

It can be seen clearly from the figure that there is only a limited relationship between the modes of the system, and the individual states of each component. Considering the light bulb, in the *Operational* mode it must be in the *ok* states and in the *Expired* mode it must be in the *ex* state, however in the *Failed* mode its state is indeterminable as any of the light bulb, switch or power supply may be the cause of the failure. It should be noted that the states of the components are assumed to only change, or become observable, due to some event in the system, thus it is only when the switch is used that the state of the components is considered — the possibilities of power supply interruption or bulb failure whilst the switch is *off* are not considered. Whilst this is clearly a "slight" over-simplification it does not affect the consideration of modes and states presented here.

Given this model it is interesting to consider how it maps to the view taken by Parnas *et al.* [40] that modes are in some way partitions of the state space of the system. In making this comparison it is important to be very clear as to what is meant by the state space

[5] Clearly a maintenance policy such as this does not make a lot of sense for a single light system — but where many lights are used the approach is beneficial.

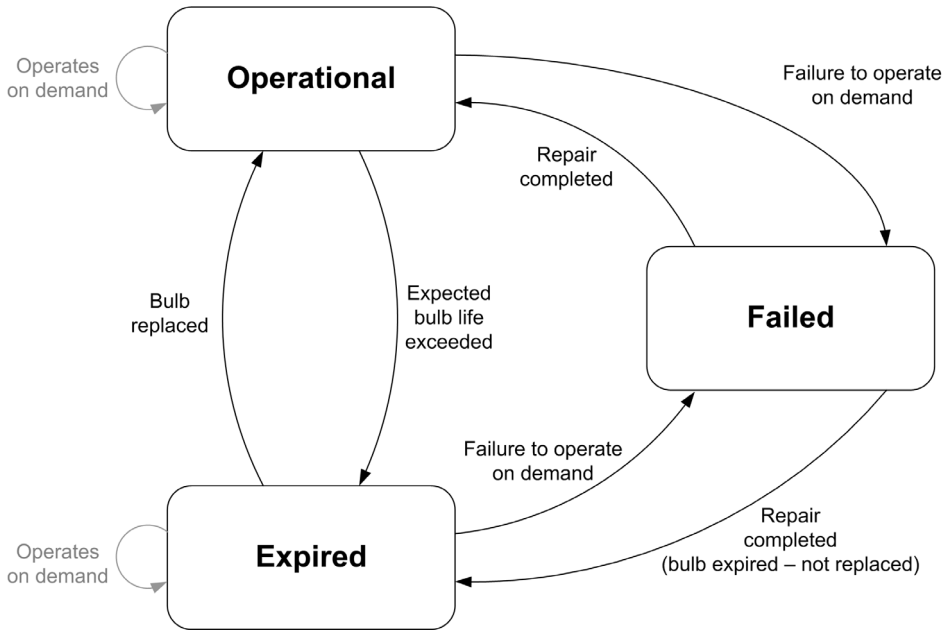


Figure 3.1: Mode model for a simple system

of the system, in this case each point in the state space is defined by a tuple comprising the states of each of the three components (*bulb*, *switch*, *power*) — the affect of the maintenance policy is factored into the state of the light bulb. The state space is the set of all tuples, for which in this simple case, there are 18 elements ($3 \times 3 \times 2$) — this view of the state space simply considers the states of the components in isolation ignoring the relationships between the states of an individual component. The state-space partitions of each of the modes are defined in the following equations.

$$
\begin{aligned}
\mathit{Operational} &\mathrel{\widehat{=}} \{(\mathit{ok}, \mathit{closed}, \mathit{on}), (\mathit{ok}, \mathit{open}, \mathit{on})\} \\
\mathit{Failed} &\mathrel{\widehat{=}} \{(\mathit{ok}, \mathit{closed}, \mathit{off}), (\mathit{ok}, \mathit{open}, \mathit{off}), (\mathit{ok}, \mathit{us}, \mathit{on}), (\mathit{ok}, \mathit{us}, \mathit{off}), \\
&\quad (\mathit{us}, \mathit{on}, \mathit{on}), (\mathit{us}, \mathit{open}, \mathit{on}), (\mathit{us}, \mathit{us}, \mathit{on}), (\mathit{us}, \mathit{closed}, \mathit{off}), \\
&\quad (\mathit{us}, \mathit{open}, \mathit{off}), (\mathit{us}, \mathit{us}, \mathit{off}), (\mathit{ex}, \mathit{closed}, \mathit{off}), (\mathit{ex}, \mathit{open}, \mathit{off}), \\
&\quad (\mathit{ex}, \mathit{us}, \mathit{on}), (\mathit{ex}, \mathit{us}, \mathit{off})\} \\
\mathit{Expired} &\mathrel{\widehat{=}} \{(\mathit{ex}, \mathit{open}, \mathit{on}), (\mathit{ex}, \mathit{closed}, \mathit{on})\}
\end{aligned}
$$

Clearly in this case the modes are indeed partitions of the state space of the system — no state tuple exists in more than one mode; and for that matter it is also complete in that all of the possible state tuples are included in the partitions.[6] However this is a very simple system, and determining the partition that corresponds to the *Failed* mode is not, even in this simple example, entirely straightforward. For instance, does the fact that the power supply is *on* or *off* really mean that the system has failed? The answer depends to

[6]Though this completeness might not always be the case for a more extensive system — some potential state combinations might be discounted as they are not feasible.

some extent as to whether or not the power supply is considered to be part of the system. An extra mode could have been introduced to handle the availability of power explicitly, which may be an appropriate approach if it is external to the rest of the system. This highlights the importance of clearly defining the system boundaries, providing a clear distinction between elements which are part of the system, and those which provide the environment.

The system in this example has been kept relatively simple, and it is viewed at a relatively high level of abstraction — for instance, there has been no consideration of the time it takes from activating the switch to the light becoming lit, clearly there will be a finite delay but the bounds of this have not been specified. Equally the power supply is considered to be *on* or *off*, the issues of fluctuation in supply and its momentary interruption have not been addressed; these temporally related issues appear to be well suited to being addressed in a timebands model.

3.2.3 An alternative model

The approach taken thus far with this example has been to emphasise the differences between the modes of the system, and the states of each individual component; taking this approach there is no complete mapping which requires that: given a component is in a state *A*, then the system must be in mode *B*. Clearly it is possible to take an alternative approach, one where there is a closer relationship between the modes and the states. An example here would be to consider the components independent of their use and define the system in more detail. Hence the components could simply have the states *ok* and *us*; and the system modes would be: *On*, *Off*, *On — Requires-maintenance*, *Off — Requires-maintenance*, and *Failed*. However this results in a significantly more detailed (concrete) model of the system with the need to model combinations such as *Requires – maintenance* and *On/Off* etc. The model is given in Figure 3.2 — the labelling of the transitions has been omitted for clarity.

In this case, the model looks quite different from that previously presented in Figure 3.1 and necessarily exposes the operation of the system, with the transitions enumerating the reasons for the successful operation or failure of the system. Fundamentally, the orthogonality between the modes of the system and states of the individual component parts has been greatly diminished; thus as the system is scaled-up to include a greater number of bulbs and switches so the size of the state-space also increases. This lack of distinction between states and modes results in the mode model becoming inevitably more complex, and progressively less clearly defined.

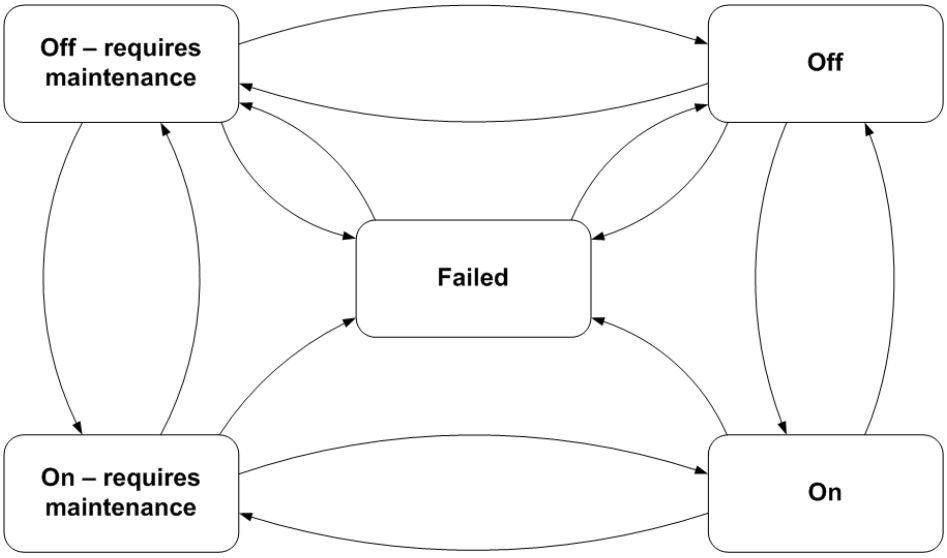


Figure 3.2: The alternative mode model for a simple system

It should be noted that the structure of this model is highly dependent on whether it is constructed top-down or bottom-up. If constructed taking the former approach it may be possible to limit the size of the state space, and hence complexity, at any level; the bottom-up approach however appears to lead to an inevitable explosion in the state space with a resultant increased complexity in defining the modal partitions.

3.2.4 A comparison of the two models

Given such simple models it is unlikely that a simple comparison will yield any great insights; it is however worth considering the scalability of the two approaches. The structure of the original model is unaffected by the number of light bulbs or switches in the system, largely due to the orthogonality between the modes of the system and the states of the individual components. Clearly as the number of components in the system increases the definition of the partitions of the state space for each mode becomes an increasingly significant problem, but the modes remain unchanged. In the case of the state-centric approach adopted for the alternative model there is a very large impact to the model due to the introduction of additional light bulbs and switches; the lack of abstraction here results in the state of each individual component being visible in the mode model — a clearly undesirable outcome in systems with many components.

The orthogonality between modes and states is indicative of the key discriminators between the two concepts — behaviour on the one hand, and condition on the other. As a side issue it is interesting to consider if there is another dimension to this orthogonality and, if so, what that might that be. Given modes are defined as behaviours and states as conditions, perhaps a candidate for this third dimension is that of function. It is possible

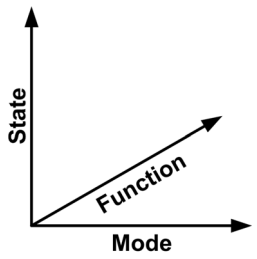


Figure 3.3: The orthogonality of mode, state and function

to consider the behaviour of the system in a particular mode as the action of a set of functions over some conditions, or state, of the system. The conclusion to this is that the behaviours of systems with weak (little) state will be dominated by function, and those with weak (simple) functions by the state. Strong mode models can balance the influences of functions and state. So rather than considering these as simply being orthogonal it might perhaps be better to see the modes as the resultant of the other two. Figure 3.3 depicts the orthogonal nature of these three attributes.

3.3 The relationships between modes and the artefacts of the timebands framework

Having investigated the nature of mode, and how it differs from that of state, the relationship between modes and the artefacts of the timebands framework now needs to be considered. First the characteristics of modes will be further defined, then subsequently the relationships between these characteristics and the timebands artefacts will be considered in the context of another simple system. Having a better understanding of the relationships and interactions between modes and the timebands concepts will help focus the approach to be taken for the addition of modal elements to the timebands framework.

3.3.1 Characteristics of mode models of systems

Modes, and the associated mode models, are generally characterised by a number of attributes, amongst the significant ones being:

- There are usually a relatively small number of discrete values that define the modes of the system;
- A mode is generally long-lived — the occurrences of inter-mode transitions are relatively (*cf.* the life of the mode) infrequent;
- They are viewed as being mutually exclusive — it is not possible to be in two modes (at the same level within any hierarchy) at the same time;

- They are global to the system — they have an impact that is system-wide;
- The transitions from one mode to another are not necessarily instantaneous — there can be an appreciable delay from the initiating event to the point where the system has stabilised in the new mode, however at some level of abstraction the transitions may be considered to be instantaneous (the transition takes no perceivable time). Irrespective of the level of abstraction, the duration of any transition is always small when compared to the time spent in a related mode;
- They are technology independent — they can affect both the physical elements of the system, the hardware and software, and its users (in the broadest sense);
- The initiator for an inter-mode transition may be external to the system being considered — hence the initiator can be considered to be asynchronous to the operation of the system;
- Mode models may be hierarchical;
- Mode models may describe concurrent modes — thus different parts of the system may be in different stable modes at the same time. Clearly this is potentially in conflict to the third point above, which stated that modes are mutually exclusive.

Clearly from the perspective of the mode model of a system there are two fundamental aspects that need to be addressed: the modes themselves; and the transitions between modes. The former defining the steady-state behaviour of the system, which persist for an indefinite duration, whilst the latter represent potentially transient behaviours of well-bounded finite duration. The reasons for mode changes are diverse: they can result from internal, and (largely) synchronous circumstances, or from external, and hence asynchronous, phenomena; they can be part of the 'normal' behaviour of the system or reflect constrained behaviour following the failure of some component.

3.3.2 Mapping modes to timebands — an informal example system

The diagram in Figure 3.4 shows a simple mode model for a system — this will be used to explore the issue of mapping of modes and transitions to the individual time bands later in this chapter. The system is a simple semi-autonomous tracking/interception vehicle. The operator of the system has some limited controls, including the ability to control the power, select and reject the item tracked by the system and "launch" the vehicle; there is also the possibility of reverting the system to a built-in test operation at any time prior to flight. The vehicle can also autonomously select a target to track if launched without a

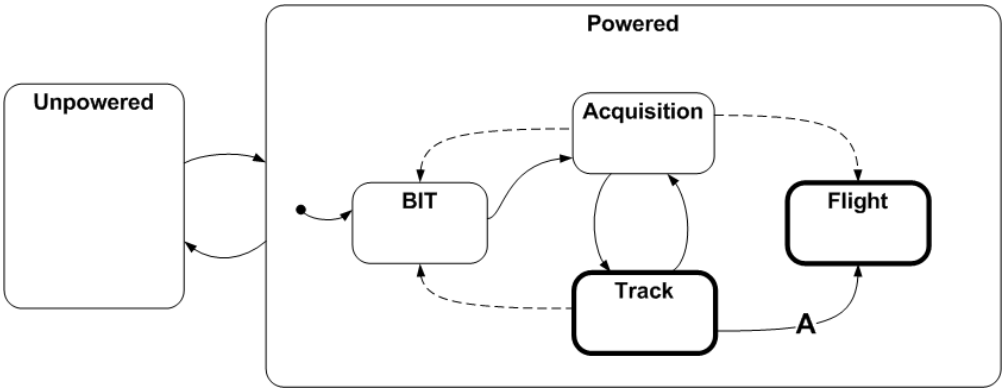


Figure 3.4: Mode model for a simple system

target. The mode model displays limited hierarchical structure, and a number of simple transitions. There are four low-level modes: *BIT, Acquisition, Track* and *Flight*. The *BIT* mode is the "default" mode of the system, the one that is entered "immediately" at power-up. Transitions out of the *Flight* mode are not possible — this is the terminal mode of the system. The dashed transitions are special cases that will not be considered further here.

The transitions between the modes in this model are invoked by events, some of which are internal to this system, e.g. *BIT* to *Acquisition* resulting from the passage of time, and some of which are external, e.g. *Track* to *Flight* — which results from some human input. The externally initiated transitions are perhaps the more interesting in that they result in asynchronous behaviours and are often the more difficult to accommodate in an idealised solution.

In order to understand some of the complexities associated with modes, and particularly the transitions, it is interesting to take one of the mode transitions shown in the figure above and consider its defining chain of events. The transition "A" from *Track* to *Flight* is initiated by the operator pressing the launch button, this triggers a chain of actions that have a range of durations; these actions are shown in the time-line diagram in Figure 3.5. As shown in the figure, it is often difficult to determine precisely at what point in one process its successor is initiated; this non-determinism is again an issue addressed and accommodated by the timebands approach.

Clearly there is a significant delay from the initiating event, the pressing of the button, to the point where the software recognises that a transition from *Track* mode to *Flight* mode is required. The mode change is not, however, complete at this point as the mode still has to be propagated through the software; this takes further time, the length of which depends on the execution profile of the software. Defining when this transition

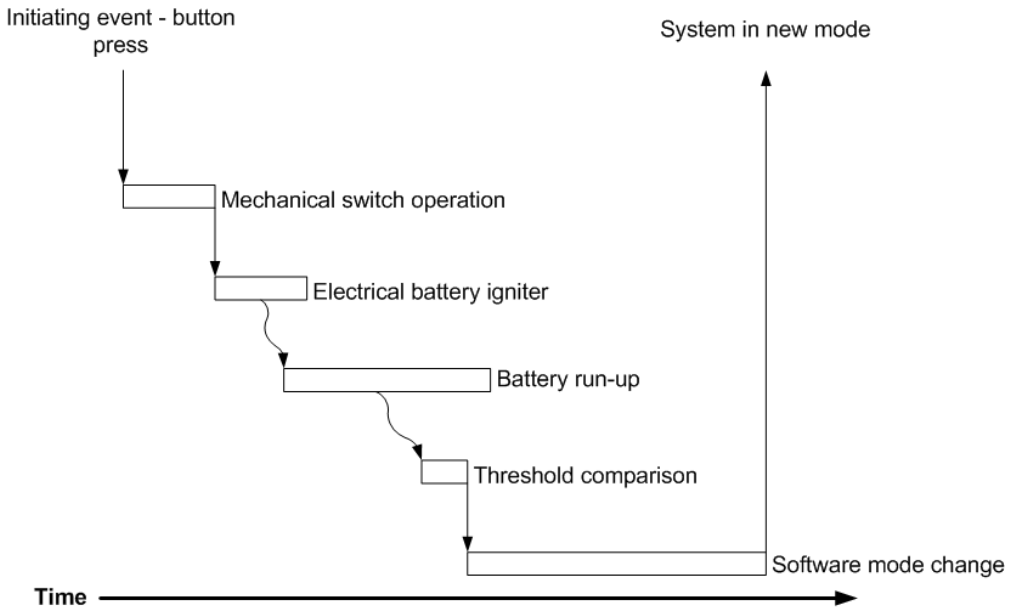


Figure 3.5: Chain of actions resulting from a mode transition initiation

has occurred is non-trivial and there are a number of views that can be taken as to when the mode change had completed. For example, AADL [91] defines the point at which the mode has changed (or "switched" in their terminology) to be that of *"the start of the next hyper-period of all the synchronised tasks involved in the mode change"*; in the case of a system composed entirely from sporadic tasks the mode switch is deemed to be instantaneous. Whilst this clearly addresses some of the temporal issues present in mode changes it is still a rather simple view in that it assumes all task iterations or execution invocations are independent of one another — implying that there are, in effect, no causal chains in the architecture. Addressing this limitation would cause the delay to be until the start of the next initiation of the causal chain, or the hyper-period of the initiation of all of the (synchronised) chains in the architecture. Thus the chain of tasks (transaction) become the subject of interest rather than the underlying tasks themselves — however, this is also somewhat weak. The position is yet more complex in the case where the chain is a pipeline whose overall execution time (for propagation through the entire chain) is greater that its iteration period; here there is the possibility for different elements of the chain can be correctly operating in different modes at the same time. This implies that there is a rather weak view of the system, and that there is too great a focus on the implementation of the mode change — perhaps because this is, in reality, a state based view of the system.

This leads to the question: what mode is the system in during this period of modal discontinuity? Clearly this is not a problem so long as the observable behaviour of the

system is either that of mode A or mode B, the problem only becomes apparent when, at an appropriate level of abstraction, the observable behaviour is that of neither mode A, nor mode B. This observability issue is an aspect that an appropriate timebands model can help to address; by viewing the modes in an appropriately constrained band the behaviour will always be mode A or mode B.

It is clear from the above that the treatment of modes, and mode changes in particular, is non-trivial, and the use of timebands to describe mode dependent behaviours has to address a number of different cases.

3.3.3 Timebands selection and mapping

Before attempting to map the modes and transitions to elements of a timebands model it is perhaps best to define a set of bands appropriate to the simple mode model presented earlier in Figure 3.4. With the bands defined the issues of mapping can then be developed against a somewhat more concrete basis.

The selection of the time bands for a system would, if taking a top-down approach, set the context for what happened in each band — in this case however the approach is somewhat more bottom-up. Here the elements of the system already have behaviours, and bands have to be selected to allow these to be mapped according to their fundamental temporal properties; however there is no real reason to assume that either a top-down or bottom-up approach are inherently better than one another.

The bands used to describe this system, progressing from top to bottom, are:

Sortie This band defines the upper boundary of the system, and relates to a deployment of the system — a mission. This band is largely defined by the "organisational environment" of the system, and as such is characterised by activities such as planning, loading the equipment, undertaking the mission, unloading the equipment, periodic maintenance etc. Durations of activities are up to 10 hours, with a granularity of half an hour and a precision of 3 minutes.

Engagement This band allows the details of the tasks identified in the **Sortie** band to be defined; for the purposes of this thesis it relates to the operation of the system to engage a target during a sortie, other possible tasks are ignored for now. It is characterised by activities of the system that are generally initiated by the operator, examples include designation, tracking and launching the vehicle. Durations of activities in this band are up to 60 minutes, with a granularity of one minute and a precision of one second.

Detection and monitoring This band relates to the operations of the acquisition, tracking and homing aspect of the vehicle. The activities in this band include sensor data

correlation and filtering, health monitoring. Durations of activities in this band are up to 10 seconds, with a granularity of 50 milliseconds and a precision of five milliseconds.

Stabilisation This band relates to the self-stabilisation functions of the vehicle. It is characterised by activities such as sensor tracking and flight control. Durations of activities in this band are up to 500ms, with a granularity of five milliseconds and a precision of 250 microseconds.

Thread This band relates to short duration activities and is characterised by activities such as software tasks and filtering of electrical signals. Durations of activities in this band are up to 500 microseconds, with a granularity of 10 microseconds and a precision of one microsecond

Platform This band defines the lower boundary of the model, and relates to very short duration activities and is characterised by activities such as the operation of electrical and electronic devices, and the firing of pyrotechnic initiators. Durations of activities in this band are up to five microseconds, with a granularity of 100 nanoseconds — as this is the lowest band the precision is left undefined.

In each of the above bands the concept of scale discussed earlier has been introduced to place an upper bound on the duration of activities in that band, the lower bound being defined by its granularity. The characteristics of the bands are summarised in Table 3.2.

Band	Scale	Granularity	Precision
Sortie	10 hr	1/2 hr	3 min
Engagement	60 min	1 min	1 sec
Detection and monitoring	10 sec	50 ms	5 ms
Stabilisation	500 ms	5 ms	250 μs
Thread	500 μs	10 μs	1 μs
Platform	5 μs	100 ns	—

Table 3.2: Band characteristics for the autonomous vehicle example

Clearly the **Sortie** and **Engagement** bands have a degree of direct human interaction, the remaining bands below these two do not. Whilst there is an attempt to keep the "temporal meanings" of the bands distinct there is some overlap, for example between the **Sortie** and **Engagement** bands, issues arising from this overlap, and the resultant partial ordering of the bands will be discussed later in this chapter.

All activities whose duration is less than that of the granularity of the lowest band are of little interest in this model. Equally those beyond the scale the upper band relating to the lifetime management of the vehicle, i.e. its procurement, operation and disposal, are also of little interest in this study of modes and mode transition. Whilst these bands provide bounds or context for the model they do not define the environment; the environment can give rise to events in any band in the model, the band being that appropriate to the temporal dynamics of the event.

3.3.4 Mapping inter-mode transitions to the timebands

The bands can now be populated with the activities for the system — for this example this is simplified to consider only the major activities for two of the modes (*Track* and *Flight*) and those involved in the transition between these modes. Before considering the steady state of each of these modes, the transition between *Track* and *Flight* modes, labelled "A" in figure 3.4, is addressed.

Taking a top-down view, the initiating event is that of the button press — clearly this is an event in the one of the bands associated with human interaction. At the level of the **Sortie** band there may be no interest in the launch event, the only link of interest to the **Engagement** band might be the commencement of an engagement. Probably the important question here is: how much, if any, of the modal behaviour of the system should be visible in the **Sortie** band? If for now it is assumed that the launch event is visible in this band then the next thing to consider is how does this event map to some activity or event in the next lower band, i.e. the **Engagement** band. Clearly, in the **Engagement** band the act of pressing the button is also a key event — whether or not the button has to be pressed and released to initiate the launch adds complexity and can be ignored here, as too can the consideration of whether, in this band, the button press is actually an activity (commencing with the press event, and concluding with the release event).

From the event sequence shown in Figure 3.5 it is clear that the button causes a mechanical switch to provide electrical power to an igniter. The first item to consider here is the mechanical switch — how long does it take to operate (largely due to human interaction); a figure of something less than 50 ms seems appropriate (in reality this would be a filtered signal with a defined response time); this places the switch activity in the **Stabilisation** band. The initiation of the battery igniter is a much faster process taking in the order of one microsecond, placing this in the **Platform** band.

The run-up of the battery is ambient temperature dependent — but overall a time not exceeding 500 ms is common, thus this can be defined as an activity in the **Detection**

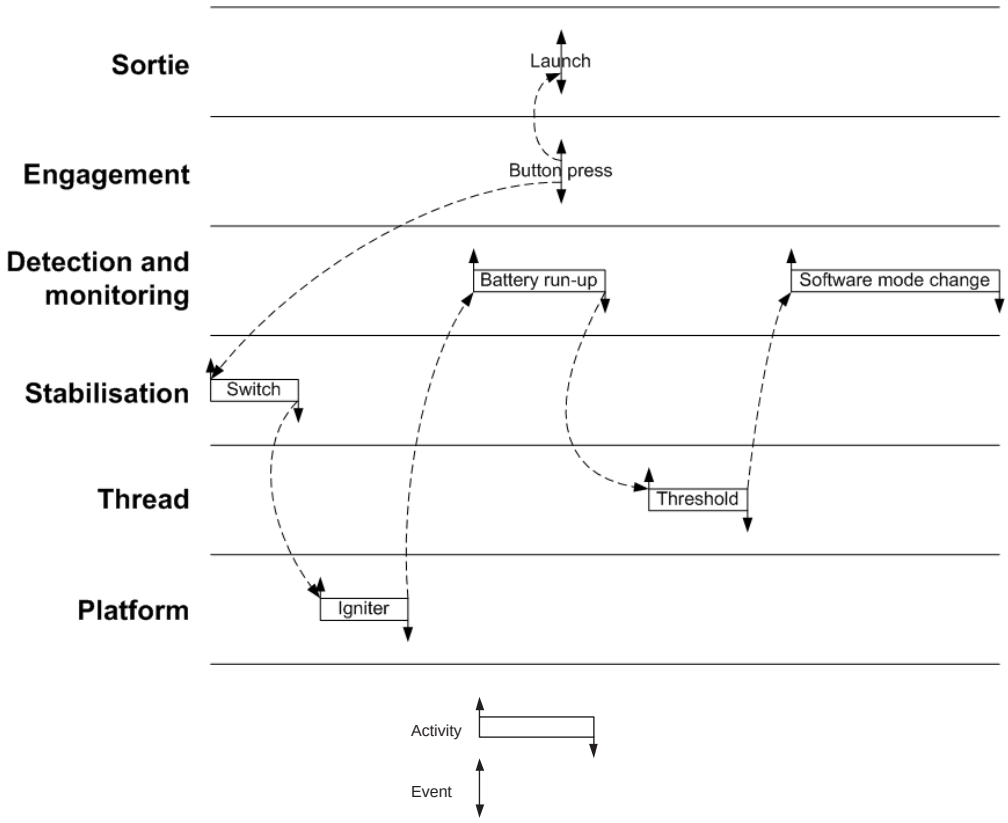


Figure 3.6: Mapping of basic activities to the time bands

and monitoring band. The threshold comparator, which detects that the battery has reached a suitable voltage, is a filtered electronic circuit; thus the logically instantaneous activity of detecting a voltage level above some threshold becomes an activity of finite duration, still this is of relatively short duration, less than 500 microseconds, placing this in the **Thread** band. Continuous processes are difficult to place in bands as they have, by definition, no duration — however, as can be seen in this example, it is the dynamics, the natural frequencies of their observable behaviours, of these processes that are important. In these continuous cases Nyquist sampling theory [73] can be used to determine the appropriate band of the activity monitoring the process.

The software executed mode change can, as discussed above, take a significant time — and can certainly be subdivided into a number of smaller elements in lower bands (for now ignored). In this case we can say that the time for this mode-change activity is in the order of 100 ms, placement it in the **Detection and monitoring** band.

The mappings of these events and activities to bands is summarised in Figure 3.6. The overall transition time is slightly in excess of 600 ms, allowing it to be modelled as an activity in **Detection and monitoring** band; this is well within with the precision of **Engagement** band (1 second).

Having mapped the basic activities to the bands there are two further aspects to consider:

- The relationships between the events and activities in the different bands;
- The precedence relationships between events within a single band.

It is difficult to fully separate these issues, but the relationships between the events and activities in different bands are perhaps the better place to start as this will lead to a consistent view across the bands; this will then allow the precedence relationships within each band to be established with a view of this full context.

Mapping relationships between the bands

In defining the mapping relationships between the bands, there are two basic approaches that may be taken. In the first there are only mappings between each pair of adjacent bands; in the second there is a more flexible approach where mappings between bands are defined as needed, and can include mappings between non-adjacent bands; the former appears quite restrictive in comparison to the second. The strict mapping approach would require the introduction of artificial events in the bands between the one in which the transitions is defined, and the one where it is realised; this would also require that the bands are totally ordered, an undesirable restriction to place on the use of the framework, and one which would probably result in poorer models. In the light of this it is clear that the second approach of allowing mappings directly between the bands as necessary is the better.

Precedence and equivalence relationships

Initially a very limited view of the system is taken in order to determine these relationships. Taking a software design viewpoint, all of the precursor activities relating to the initiation and rise time of the battery are subsumed into an overall “delay”, an activity which precedes the start of the software mode change. The overall duration of this delay is in the order of 550 ms, placing it in the **Detection and monitoring** band. This is the same band as that of the software mode change activity, hence a strict precedence relationship can be seen between the events defining these two activities — the “delay” must be complete before the software mode change can start, i.e. the end event of the former displays precedence over the start event of the latter. At this simplified/abstracted level the precedence between the events is clear, though the mapping of the completion of the software mode change to the higher levels is, at this point, unclear.

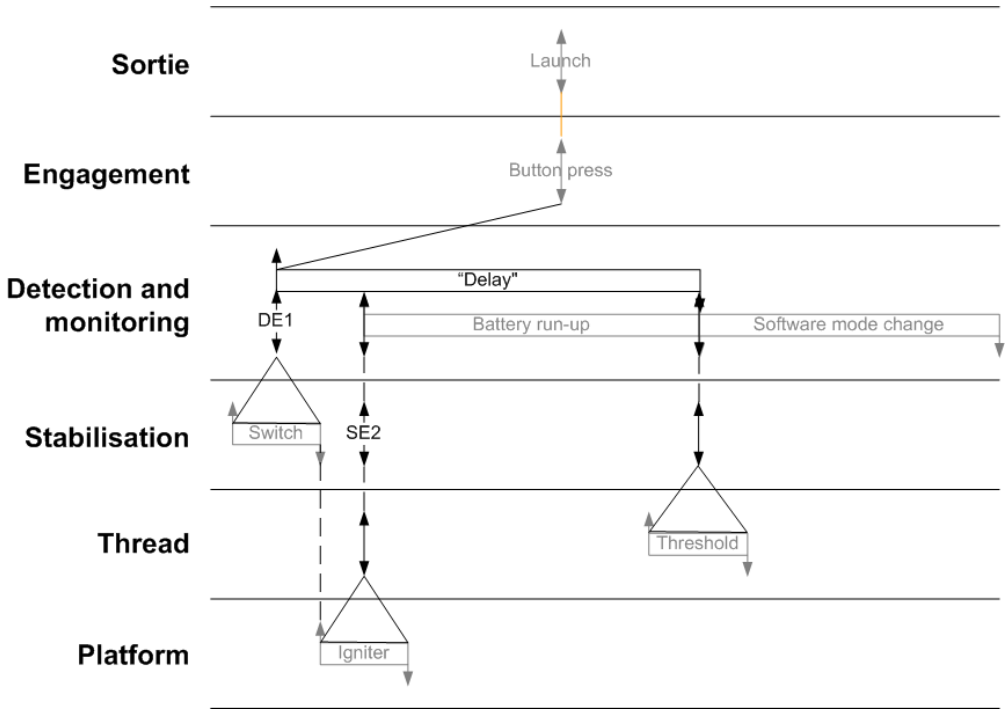


Figure 3.7: Mapping of events and activities across the time bands

Having developed an abstract view of the initiation of the mode change in one band it is now possible to expand the delay in the same band to identify the finer grained activities and events. At the level of the **Detection and monitoring** band the delay comprises:

- Initiation event
- Battery ignition event
- Battery run-up activity
- Battery threshold detected event

Precedence here is again quite clear: the "initiation event" precedes the battery initiation, which in turn precedes the start of the battery run-up. In effect the "initiation event" is the start event of the "delay" with its completion being the "threshold detected event". The events at this level, now have to be mapped to their corresponding activities or events in lower bands, e.g. **Stabilisation**;[7] for now the detail regarding the events associated with the start and end of the battery run-up activity will be ignored. The initiation event maps directly to the button start event in the **Detection and monitoring** band (DE1). The battery ignition event maps to another event in the **Stabilisation** band (SE2) — the precision of this band being inadequate to describe the ignition activity. The threshold detection event also maps to another event at this level — again the precision at this level

[7]Though there is no requirement for this to be the adjacent lower band.

being inadequate to describe the duration of the threshold detection activity. Clearly in terms of precedence the button start event must precede the ignition event, but there is no need for the button action to complete before the ignition of the battery. The firing of the battery igniter clearly needs to occur to allow the start of battery run-up however the two events are, within the precision of the **Detection and monitoring** band, simultaneous, posing the question: does their coincidence allow them to be considered as equivalent (simultaneous) events in that band?

The end event of the battery run-up is of little interest here — the detection of the battery voltage crossing a threshold is of more interest. This event will, given the precision of this band, be effectively coincident with the start of the software mode change activity. Again: can these be considered to be equivalent events at this level? The end of battery run-up may not be coincident with the other events as the battery voltage is likely to continue to rise after the threshold has been passed, and fluctuate thereafter.

The answer to both questions regarding equivalence is based on the definition of equivalence from the formalism — for equivalence to hold each has weak precedence over the other. When considering a single band, i.e. ignoring the detail from the lower bands, it is possible to say in this case that these are the same event; however, this may not always be possible. As an example, in the case where a number of activities in the lower bands have a total duration greater than the precision of the higher band discrete, time separated events may be visible in the higher band, then one event will show precedence over another; where the separation is less than the precision the events become indistinguishable in time and may be viewed as being equivalent or simultaneous. This is allowing the time bands to achieve the goal of abstracting away the detailed issues that have no impact on the observed behaviour of the system in a particular band. The mappings of these additional activities and events to the bands is summarised in Figure 3.7.

The equations below define the relationships between the events in the **Detection and monitoring** bands.

Let:

$$
\begin{aligned}
D^{dm} &= \text{Delay (in } \textbf{Detection and monitoring}\text{)} \\
B^{dm} &= \text{Battery run-up (in } \textbf{Detection and monitoring}\text{)} \\
Sw^{dm} &= \text{Fire switch (in } \textbf{Detection and monitoring}\text{)} \\
S^{dm} &= \text{Software mode change (in } \textbf{Detection and monitoring}\text{)} \\
T^{dm} &= \text{Threshold detected (in } \textbf{Detection and monitoring}\text{)} \\
I^{dm} &= \text{Battery ignition (in } \textbf{Detection and monitoring}\text{)}
\end{aligned}
$$

The equivalence and precedence relationships are:

$$Sw^{dm} \equiv \uparrow D^{dm}$$

$$I^{dm} \equiv \uparrow B^{dm}$$

$$T^{dm} \equiv \uparrow S^{dm} \equiv \downarrow D^{dm}$$

$$Sw^{dm} \prec S^{dm}$$

$$Sw^{dm} \prec T^{dm}$$

As demonstrated in the section above there is a relatively simple relationship between the timebands artefacts and the elements of an inter-mode transition. The mapping of the transitions to a single event in a high band, and its realisation as a sequence of activities and events in the lower bands is a straightforward approach that does not appear to have any real weaknesses; though the introduction of additional events into the intermediate bands does appear to be to a large degree superfluous and can be avoided through the definition of explicit granularity mappings between non-adjacent, ordered bands. It is clear that the best approach is to define the actual relationships directly even when this requires the forming of relationships between non-adjacent bands.

3.3.5 The relationship between modes and time bands

The mapping of the modes themselves to the time bands structure is perhaps a less clear and more problematic issue to address than that of the inter-mode transitions. As shown in the previous section the transitions can be seen to map quite naturally on to events at a high level, and on to activities and events at the lower levels. The modes themselves are more abstract and do not appear to so easily fit with the time bands concepts; whereas transitions are routed in low-level events, it seems inappropriate to consider the abstract concepts captured by modes in the same fine detail. So to start, the nature of modes is considered further.

As noted earlier, whilst the transitions are clearly processes of finite (and bounded) duration, this is not a characteristic of modes; in the general case modes describe behaviours of the system, their persistence is only bounded by the requirement to change mode. In some cases this persistence can be fixed, e.g. a built-in test mode may last for a (relatively) fixed period; however, the persistence of other modes may be controlled by asynchronous events, such as operator actions, and may, hence, be relatively unbounded. The question this variability poses is: (how) can this diverse nature of mode be accommodated in the timebands framework in its current form? Additionally there is also clearly the need to ensure that the representation of modes and the inter-mode transitions are compatible.

Thus there are two issues to be addressed here:

- The mapping of modes to time bands — in which band(s) should the mode reside;
- The association of modes with artefacts of the bands — how is the mode composed from, or related to, the activities and events of the band(s).

Taking each of these in turn, the first issue to be considered is that of mapping the modes to the bands.

Mapping modes to time bands

Here two questions immediately emerge:

- In how many bands should the modes be apparent?
- How should the refinement (mapping) of the modes from a coarse band to finer grained bands be addressed?

As shown in the previous section the transitions naturally map to an event in one band; this would seem to indicate that the modes involved in a transition should also map to a single band, and that the band should be same as that for the transition event. Whilst simple this does not address one of the characteristics of mode models, that of modal hierarchy. This issue was not considered with respect to the transitions largely because the transitions are generally not hierarchical,[8] but it does need to be addressed with respect to the modes themselves.

An apparently straightforward approach to modal hierarchy would be to map each layer in the hierarchy to a different band. In the example system presented earlier (and putting to one side for now the question of the visibility of inter-mode transitions) it is possible to envisage the *Powered* and *Unpowered* modes being mapped to the **Sortie** band, with the sub-modes of *Powered* (*BIT, Acquisition, Track* and *Flight*) being mapped to the **Engagement** band — a straightforward allocation. The limitation of this approach would become apparent for a highly hierarchical model, i.e. one with many levels. If applied literally this would result in a large number of bands with, in all probability, too little distinction and too much overlap between bands — thus making the task of band allocation more difficult and more arbitrary, inevitably resulting in a weaker model. However it could be argued

[8] Transitions are always between modes at the same level within the hierarchy, therefore there is no hierarchy of transitions, though in a hierarchical model it will be necessary to define "compatible" transitions at several levels.

that a model with such a hierarchy is not a good mode model at all, but rather, perhaps, one based on state rather than behaviour.

Given that the bands are chosen to discriminate the key natural temporal dynamics of the system then it is probably necessary that at most a single level of the modal hierarchy is permitted to be mapped to a single band, and that the hierarchical level is complete within that band. It appears that the fundamental issue of mapping is to ensure that all of the set of related modes (i.e. ones with inter-mode transitions) are mapped to the same time band; if this cannot be achieved due to the diversity of the timing characteristics, then it possibly implies that the mode model is inappropriate. As to whether layers in the hierarchy should be mapped to adjacent bands the position is unclear — it does not appear necessary, or indeed desirable, to impose this as a constraint especially when the partial ordering of the bands is considered. In order for layers of the hierarchy to map to different bands it is necessary that there be an ordering between the bands. The issues around the mapping of modal hierarchies to the time bands are explored further in the case study presented in Chapter 6.

A clear outcome from the above discussions is that a set of modes and their associated transitions, a mode model, will be defined in a single band.

Relationships between modes and timebands artefacts

Moving on from how modes are mapped to the band structure, the issue of how a mode may be related to the other artefacts in its and other bands can be considered. Essential to this is the basis of the concept of modes — the fundamental behaviours of the system in the context of some overall condition. There are perhaps two fundamentally different, contrasting, approaches that may be adopted here:

- Compose the modes from the existing artefacts, i.e. the activities, events and behaviours; or
- Create a new artefact, which is independent of the activities and events, i.e. first-class in it own right and use these to constrain the other artefacts.

Both of these approaches are considered, in turn, in the sections below.

Constructing modes from the existing artefacts — the compositional approach

As noted earlier a mode denotes a particular behaviour; its persistence can be variable, but is generally significant (certainly not transient); and it may be entered and exited a number of times through well defined entry and exit transitions. There are three elements

of the timebands framework that can be used to describe a mode within a band: activities, events, and behaviours. Of these the events appear to be the least likely candidates, unlike modes they have no "persistence" — thus events will not be considered further.

Activities have a clear beginning and end, and have a duration — thus, superficially at least, they appear to be able to be mapped to modes. However the definition of an activity in the timebands framework, given in the original report [11], is: *an item of work undertaken by some agent*; whereas modes relate more to the behaviour of the system in the context of some condition — this leads to the question: are these two concepts compatible?

Possibly the key here is level of abstraction, and two issues need to be considered:

- From the modal viewpoint: in a high band is it possible to view a mode as something performing an "element of work"?
- From the transition viewpoint: can an activity be described as something defining the behaviour of the system?

Taking, as an example, the **BIT** mode, this is an element of work (a clearly defined task) and could thus be described as an activity in the timebands framework. This is also helped by the mode having a relatively fixed duration — another key characteristic of the activity.[9] If however a mode such as **Acquisition** is considered things are considerably less clear.

The purpose of the **Acquisition** mode is to use sensors to scan the environment and present elements of interest to the operator, this involves the filtering and correlation of data from the sensors, the mode is left when the operator selects an item of interest to track; therefore, by its very nature it does not have a fixed (or even relatively bounded) persistence. However, at an appropriate level of abstraction the task undertaken can be considered an element of work (albeit a rather complex one when viewed in any detail). Therefore the only issue here is that of the lack of certainty in the persistence — this is a weakness of the timebands approach that has already been noted. If the **Track** mode is considered the same problem with persistence arises — the operator can leave the system tracking the object of interest for a largely indefinite period; but here again the mode can still be said to be closely related to an element of work (directing sensors, correlating sensor data and eliminating "noise"). The **Flight** mode again can be considered to be

[9] Though the appropriateness of having a fixed duration for an activity has already been questioned.

closely related to an element of work, however here the variability in persistence issue is less pronounced — the **Flight** mode persistence is well bounded, but again not fixed.

Continuing to accept, for now, that activities can be used to model a mode in one band the next question is: How should the mode be exposed in the next lower band? If the band in which the mode (as an activity) was defined is treated as being one defining requirements, then the adjacent lower band has to provide a corresponding response. This implies that there should be a strong mapping between the artefacts in the two bands — the activity in band A has to be mapped to a set of activities and events in band B. This results in the mode being clearly visible in both bands. But this leads to the questions:

- In which band does the mode truly reside?
- Is this not just a form of functional decomposition?

The answers to these will be considered later when the two alternative approaches to the definition of mode are compared.

In the lower band, the set of activities and events appears, superficially at least, to map well to the timebands concept of the behaviour. Each mode in this lower band would map to a particular behaviour artefact encompassing all of the functionality "operating" in that mode, giving rise to the system behaviour particular to that mode. As not all functions are exclusive to a given mode there would be common elements in different behaviours, e.g. some form of continuous monitoring or built-in test.

There is, however, a problem with behaviours relating to precedence: behaviours do not themselves describe precedence, this is only defined between events within the band as a whole, and thus they have to apply across all modes. As different modes can exhibit significantly different behaviours it seems unlikely that having one set of precedence relationships defined at the band level would provide the flexibility required to model the different modes. Is this a limitation of the timebands model? This needs to be considered further, but for now behaviours will be used to describe modes (when viewed in a compositional manner), as these allow the interactions between modes and inter-mode transitions to be considered.

Modes as first-class artefacts — the constraining approach

Introducing the mode as a first-class artefact into the framework appears at first sight to be relatively straightforward, and, when viewed in isolation, this is indeed the case. However, complexity begins to arise when the mode is considered in terms of its relationships with the other artefacts — the events and activities.

A naive approach would be to follow something similar to that described above for the composition of modes, but this would inevitably detract from the first-class status, and probably not address the emerging complexity apparent above. Clearly a different approach is required if modes defined in this way are to enhance the framework. The introduction of mode as a first-class entity must not invalidate any existing timebands model, i.e. the modal artefacts are optional. Thus it should also be the case that any valid timebands model that included mode-related artefacts should remain a valid model with the modal elements removed.

An alternative approach is to use the modes to refine the definition of the other artefacts — they provide some constraints as to the occurrence of the activities and events; modes can be used as pre- and post-conditions over activities, and invariants over activities and events. Whilst this is first sight a very simple approach, it is none the less quite powerful. It allows, for example, for the model to constrain an activity to only start, end, or execute in a given mode providing a much refined context for its execution; but it does not demand completeness of mapping that would be the case in previously described approach.

Perhaps the greatest strength of the approach is that it allows for a very flexible relationship between the bands. By simply using the modes as constraints it is possible to apply the constraints across the bands — thus an activity in a lower band may be constrained to only execute in a mode defined in a higher band. There are two conditions that must be satisfied for these constraints to be defined; these are:

- The constrained activity or event must be defined in either the same band as the mode or one lower in the hierarchy; and
- There has to be an ordering between the band in which the mode is defined, and that of the constrained activity or event.

This naturally complements the notion of partial ordering of the bands — constraints are applied as and when necessary, no total mapping is required.

3.3.6 The interactions between modes and inter-mode transitions

Having considered the representations of both the transitions and the modes the next step is to combine the two and consider how they interact. The interactions between modes and transitions effectively complete the model, however, depending on the approach to the definition of the mode within the framework, these interactions are quite different with the “composed” approach being quite complex, and the “constraining” approach

being somewhat less so. The interactions in each of these cases are considered in the sections that follow.

The composed approach

In the case of the composed approach to mode definition there appears to be at least two levels to be considered — the band in which the modes are modelled as activities, and the one in which they are modelled as behaviours. Taking these in turn.

In the band where the modes are modelled as activities it appears that the transitions should be modelled as events. This is largely based on the view that modes have significant duration whilst transitions do not — thus it would seem inappropriate to model a transition as an activity at the level where modes where also modelled as activities. Clearly if the bands are defined as being too broad then it would be possible to have relative short duration activities modelling transitions existing alongside other, longer duration, ones modelling the modes — but having bands this broad would perhaps suggest that the selection of the bands had resulted in a weak model of the system.

The position is significantly more complicated for the band in which the mode is modelled as a behaviour. Here the way in which the transitions are described is largely dependent on the duration of each transition and the granularity defined for the band. In the case of transitions with very short durations, less than the precision, these could continue to be described as events; however for longer durations transitions they could be described as activities, or in extreme cases as behaviours — though this latter case might again imply that the bands had been badly selected, resulting in a weak model. The mode change between *Track* and *Flight* modes for both the **Engagement** and the **Detection and monitoring** bands is shown in Figure 3.8 — here the Track-mode activity is mapped the Track-mode behaviour, the Flight-mode activity to the Flight-mode behaviour, and the Button-press event to the Button-press activity.

In addition to considering how the transitions should be modelled, the issue of precedence between the activities and behaviours of the mode, and the events and activities of the transitions also need to be addressed. A simple approach in the band where modes are represented as activities would be to have a relationship that states the “old mode” (or activity in this mapping) must have started before the transition event can occur, and that the transition event must occur before the start event of the “new mode” (activity). In addition there needs to be a constraint that ensures the “new mode” start event does not occur before the “old mode” end event. Taking the case of the *Track* to *Flight* transition in the **Engagement** band these relationships are given in the equations below.

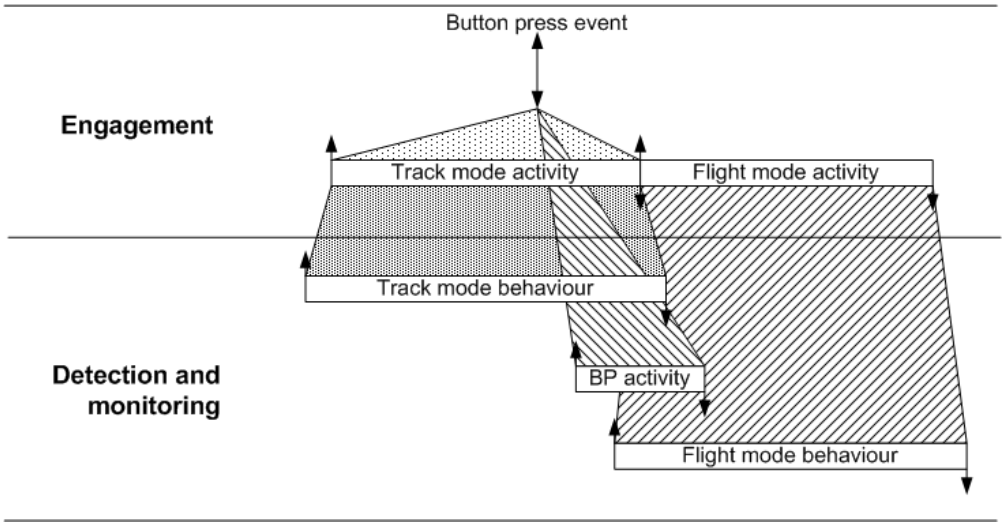


Figure 3.8: Track — Flight mode change

Let:

$$T^e = \text{Track mode activity (in } \textbf{Engagement}\text{)}$$

$$F^e = \text{Flight mode activity (in } \textbf{Engagement}\text{)}$$

$$Bp^e = \text{Button-press event (in } \textbf{Engagement}\text{)}$$

The relationships are:

$$\uparrow T^e \prec Bp^e$$

$$Bp^e \prec \uparrow F^e$$

$$\downarrow T^e \prec \uparrow F^e$$

In the case of a band where modes are described as behaviours, and the transitions as activities the position is less clear. It is helpful for each mode (behaviour) to define an entry and an exit event[10] — corresponding to the notion of the entry and exit points of a mode. Taking this approach the transition activity would be constrained to start after the behaviour describing the old mode had been entered, and would be constrained from completing until the exit event of the behaviour had occurred. Likewise, the entry event of the behaviour describing the new mode could not occur until after the start event of the transition activity, and any exit event of the new mode could not occur until after the end event of the transition. In this case it should be noted that there is no direct relationship between the two behaviours representing the two modes. Equations for these precedence relationships in the context of the *Track* to *Flight* transition in the **Detection**

[10]Rather than a single event these could be sets of events with one relating to each possible transition into or out of the mode.

and monitoring band are given below. As shown in Figure 3.8 this leads to overlapping between the two modes, perhaps suggesting that this band does not have the appropriate degree of temporal abstraction at which to view the modes and the inter-mode transitions. In "real" systems this overlapping of modes is an effect that is often observed, in general this can be attributed to not having any form of temporal abstraction, and thus simply viewing the system at too fine a level of detail.

Let:

T^{dm}_{entry} = Entry event to the *Track* mode behaviour (in **Detection and monitoring**)

T^{dm}_{exit} = Exit event from the *Track* mode behaviour (in **Detection and monitoring**)

F^{dm}_{entry} = Entry event to the *Flight* mode behaviour (in **Detection and monitoring**)

F^{dm}_{exit} = Exit event from the *Flight* mode behaviour (in **Detection and monitoring**)

Bp^{dm} = Button-press activity (in **Detection and monitoring**)

The relationships are:

$$T^{dm}_{entry} \prec \uparrow Bp^{dm}$$

$$T^{dm}_{exit} \prec \downarrow Bp^{dm}$$

$$\uparrow Bp^{dm} \prec F^{dm}_{entry}$$

$$\downarrow Bp^{dm} \prec F^{dm}_{exit}$$

The first-class artefact, or constraining approach

Looking now at the alternative approach, that where the mode is defined as a first-class artefact; this is, as noted previously, a somewhat simpler case. The fact that the mode is, in effect, independent of the events and activities removes large number of the issues discussed above for the "composed" approach. However, in the earlier discussion on this representation of modes the issue of inter-mode transitions was not addressed.

Given the modes are defined in a single band, and accepting the mutual exclusivity of modes, the system must be in one of those modes; it therefore follows that the transition between modes must be an event in that band. This implies that the "boundary" of a mode is defined by the transitional events that "surround" it — the ways of entering and leaving the mode.

In the case of the *Track* to *Flight* transition there is little change over the first approach except that the only band of interest is that in which the mode model is defined (**Engagement**), in this band the transition is reduced to an event — that of button-press. Thus the

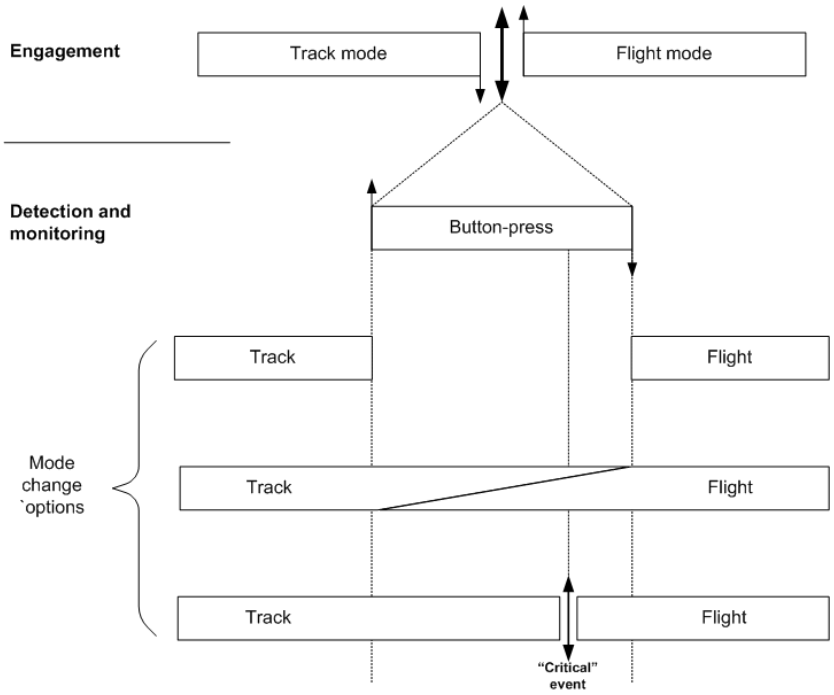


Figure 3.9: Mode change options

three events: end of *track*, start of *flight*, and *track-flight* transition are coincident and thus equivalent.

$$\downarrow T^e \equiv \uparrow F^e \equiv TF^e$$

Taking this approach it is not possible to define precedence relationships between the mode exit and entry events in the **Engagement** band, and the start and stop events of the button-press activity in the **Detection and monitoring** band. However there is now the opportunity to express the relationship (in the **Detection and monitoring** band) between the button-press activity and the modes using the constraints.

There appear to be three possible approaches to defining such constraints depending on how the transition is viewed; these are:

1. The transition is modeless;
2. The transition spans both modes — the mode during the transitional period being indeterminate;
3. There is some "critical" event in the transitional activity that denotes the actual point of mode switch.

Each of these approaches is shown in the Figure 3.9. Considering the nature of the transition at a significantly lower band, e.g. **Thread**, helps in identifying which of these approaches is correct. In this lower band the single transitional button-press activity will map onto a number of events and activities — a button-press behaviour. Viewed in this

band the nominal start of the behaviour will map onto the start of the button-press activity, and the nominal end of the behaviour will map onto the end of the button-press activity. Between these nominal events bounding the behaviour there can be other activities in the lower band, which in-turn map onto other modal activities in the higher bands, thus it is not possible to infer that all modal related activities are completed prior to the start of the button-press behaviour and hence the first, modeless option is not correct. Another issue to be considered in deciding which of the other two approaches is correct is that of consistency of view across the bands — there has to be consistency up through the bands, i.e. if the system is in a certain mode in one band then it should be in the same mode (or possibly a parent mode) when viewed in a higher band.

Irrespective of which of the other two options is correct, the constraints are the same and can be stated as:

Track pre $\uparrow Bp^{dm}$	Being in *Track* mode is a pre-condition for the start of the Button-press activity
Flight post$\downarrow Bp^{dm}$	Being in *Flight* mode is a post-condition for the end of the Button-press activity

This indicates the real strength of this approach — it allows relationships to be formed between artefacts in different bands and gives context to the lower levels of the model.

The final two options are, in fact, only distinguished by the band in which they are viewed; the critical event could ultimately be viewed as an activity or behaviour in some lower band, and then the modal view across that activity will be ambiguous. Of course it could be the case that mode was of no interest in this lower band, end hence there would be no such ambiguity.

A comparison of the composed and constraining approaches

Whilst the two approaches take a fundamentally different view of the definition of mode they do not take an incompatible view of the inter-mode transitions. In both cases the transitions can be reduced to an event in the highest band of the mode model. There is perhaps a difference in emphasis here in that with the composed approach there is more of an emphasis on the contents of the mode in different bands, and this leads to a more detailed view of the transition in terms of the events and activities, and their associated precedence relationships.

It is difficult to compare the approaches directly; this is especially the case in the absence of the remainder of the overall model of the system, as the constraining approach mode

only has any benefit when it can be used to apply constraints to other elements of the model. The scalability of the composed approach is certainly questionable.

3.4 Summary

In this chapter the nature of modes has been considered in some depth, both in relation to the related concept of state, and their incorporation into the timebands framework. Put quite simply modes represent the key behaviours of the system when viewed at some level of abstraction. The two approaches to modelling the early lighting system example highlighted the difference, or orthogonality, between mode and state, and it has been suggested that there might be a third dimension, itself orthogonal to the other two, a candidate for this being function.

If a behavioural view is taken, as is the case with the timebands approach, then the concept of systems modes is at the very least useful, and potentially essential: hence it is appropriate to enhance the timebands framework with modes.

Whilst it is possible to develop the mode model in any of a number of bands, it is clear that:

- Any single hierarchical level of a mode model must be defined completely in one band; and
- All inter-mode transitions for that level must be mapped to events in that band.

This leaves open the question as to how to represent the mode itself, which is dependent on the approach adopted, i.e. whether it is composed from existing artefacts, or defined as something entirely new to the framework.

Two distinct approaches to the definition of the mode have been proposed; in the next chapter one of these, the constraining approach, will be developed further and formally defined. This formalism will also provide a sound basis upon which the important issues of hierarchy and concurrency will be considered in more detail. Even in advance of this formalism it is clear that modes provide an extra dimension to the timebands framework for relating the lower-level artefacts to higher level system behaviours — a new abstraction providing cross-band constraints. Allied to this, perhaps an interesting observation at this point is that of the lack of importance placed on the concept of mode when considering the architecture of systems in general — this seems to be a significant oversight when modes appear to be a particularly useful concept, which is more than simply function or state, for connecting the higher levels of system organisation and behaviour.

Chapter 4

System Modes: their Inclusion in the Timebands Framework

In the previous chapter, the concept of system modes and their relationship with the timebands framework was considered in some depth, with two possible approaches to their incorporation being identified — a constraining approach, and a composed approach. In this chapter the formal representation of the timebands framework is extended to incorporate the first of these approaches; extending the formal framework to include modes ensures that the concepts of the framework are used consistently, and provides a complete definition against which tools can be constructed. The extensions will be explored through their application to one of the simple example systems described in the previous chapter, and then be used to consider in more depth the issues of modal hierarchy and concurrency. The remainder of this chapter is divided into the following parts:

- A formal description of the modal extensions to the framework — to some extent this also reiterates the informal description of the constraining approach proposed in the previous chapter;
- The application of these extensions to the simple example air-vehicle system originally presented in the previous chapter. This includes the formalisation of the modes and their transitions, and the application of modal constraints to some of the activities of the model;
- Development of an approach to modal hierarchy, the issues exposed by the formal approach, and how the formal description can be extended to accommodate such hierarchy;
- Model concurrency, again considered in the light of the formal framework.

4.1 Extension to the formal framework

The approach adopted in extending the formal framework is to provide a new, first-class element for the timebands framework, i.e. one that is independent of the existing elements — the activities and events. This approach allows models to adopt the modal view where it is beneficial to the domain of the system being described, whilst not being forced so to do when modes are not considered necessary or useful. By creating a new, independent element to describe modes it is possible, subsequently, to use these modes, and the models constructed from them, to describe constraints on other parts of the overall timebands model.

There are three aspects that need to be considered in developing the extensions to the formalism, these are:

- The modes;
- The transitions;
- The overall mode model — the combination of the modes and the transitions.

Each of these is discussed in the sections below, starting with the modes themselves.

As with the original formal description of the framework, given by Burns *et al.* [11], the Z notation, as defined in by Spivey [99], is used to describe the extensions.

4.1.1 Modes

In the constraining model a mode makes no reference to any other parts of the model except for the band for which it is defined. Within this band the mode will have some life (temporal extent) or persistence, i.e. the system will be in a particular mode for a, potentially variable, period of time. This persistence can be measured in clock-ticks of the band of the mode, with the minimum possible persistence being one tick; modes of zero persistence do not appear to be useful for describing the fundamental system behaviours inherent in a mode, as these would, in effect, be equivalent to events. An upper bound is placed on the persistence so as to provide some indicator of the overall temporal range of the particular mode.

The bounds of persistence can be used to validate the compatibility of modal constraints on activities with respect to their own attributes, e.g. for an activity constrained to “execute” within a given mode, its duration should be compatible with the persistence bounds of the mode. The outcome of this can be to identify a clearly invalid model, where the

duration of the activity exceeds with the upper bound on the persistence of the mode; or a potential issue where the duration of the activity is within the persistence bounds — indicating an activity that might on some occasions have to be aborted, or terminated prematurely, on exit from the mode.

Formally the modes have two characteristics: their band, and their persistence. Using $\mathcal{M}$ and $\mathcal{B}$ to signify the modes and bands respectively, the band of a mode is defined thus:

$$band : \mathcal{M} \rightarrow \mathcal{B}$$

and the persistence is defined:

$$persistence : \mathcal{M} \rightarrow \mathbb{N}_1 \,..\, \mathbb{N}_1$$
$$\#persistence \geq 1$$

The persistence is expressed as a contiguous range, describing a set of persistence values that the mode might take. The constraint simply expresses the fact that the maximum persistence bound must exceed or be equal to the minimum persistence bound, i.e. there should be at least one possible persistence value. As noted earlier, the bounding values of the persistence are in clock ticks of the band — and must be positive values.

4.1.2 Transitions

As discussed in the previous chapter, in the case of the constraining approach a transition between modes is instantaneous, and hence it can be mapped to an event — thus no new elements need to be introduced to the framework to describe the transitions. However, it should be noted that not all events in a band correspond to (or result in) inter-mode transitions; it is thus perhaps useful to consider the relationships between the modes and the "transitional" events in some detail before considering the overall mode model.

A "transitional" event C is used to relate the change of the system from mode A into mode B, hence there is a logical mapping:

$$A \xrightarrow{C} B$$

This mapping poses a two distinct questions:

- Can a particular event only cause a single mode transition?
- Can the transition between two particular modes only be caused by a single event?

The first of these can be answered by considering the transitions *Acquisition* and *Track* modes to *Flight* mode given in the tracking/interceptor vehicle example described in the previous chapter. Here the transitions are both due to the same event — pressing the fire button. Hence it would clearly be over-constraining to require that transitional events be restricted to unique inter-mode transitions.

The second case, that of multiple events causing the same inter-mode transition, is not directly addressed by the example; however the *BIT* mode of that system may be exited, to *Acquisition*, for a number of reasons, including successful completion and timeout. It would be quite reasonable, from the viewpoint of systems architects, to treat these as two distinct events, which ultimately map to different activities in a lower band — hence transitions between an particular pair of modes, and in a single direction, can be due to any of a number of transitional events.

The simple model shown in Figure 4.1 shows both of the kinds of situation described above. Event X will cause the model to enter mode *A* from either of modes *B* or *C*; and either of the events Y or Z will cause the model to enter mode *C* when it is in mode *B*.

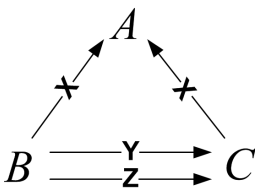


Figure 4.1: Simple inter-mode transitions

Given these observations there are number of ways in which, formally, the relationships between the modes and the transitional events may be expressed. The simplest approach would be to define each transition as an n-tuple of the event, the current mode, and the resulting mode, i.e:

$$Event \times Mode_{current} \times Mode_{next}$$

With the set of transitions for a model then being defined by:

$$\mathbb{P}(Event \times Mode_{current} \times Mode_{next})$$

An alternative, but similar, approach would be to define the transition tuples of events and mode pairs:

$$Event \times (Mode_{current} \times Mode_{next})$$

In this case the set of transitions may be defined by:

$$Event \nrightarrow (Mode_{current} \times Mode_{next})$$

Both of these approaches imply a number of constraints that have to be satisfied in order that their definitions are consistent and correct, these include there not being multiple associations of a particular pair of event and current mode. Considering this particular constraint further leads to yet another alternative approach, one where an event and current mode pairs lead to the new mode; this is given by:

$$(Event \times Mode_{current}) \times Mode_{next} \tag{4.1}$$

With the set of transitions for a model being given by:

$$\mathcal{T} == Transitions == (Event \times Mode_{current}) \nrightarrow Mode_{next}$$

This captures the (functional) constraint of an event-mode pair only being able to lead to a single possible outcome, and the additional fact that not all event-mode pairs result in a mode transition (partial). This is the approach that will be used in the formal definition of the overall mode model. Considering the simple mode model in Figure 4.1 the formal definition of its transitions is given by:

$$\begin{aligned} &SimpleTransitions == \{ \\ &\quad ((Z, B) \mapsto C), \\ &\quad ((Y, B) \mapsto C), \\ &\quad ((X, B) \mapsto A), \\ &\quad ((X, C) \mapsto A)\} \end{aligned}$$

Formally the set of transitions may be defined thus:

$$\mathcal{T} : (\mathcal{E} \times \mathcal{M}) \nrightarrow \mathcal{M}$$

When considered more fully there are a number of additional constraints that need to hold for a set of transitions relating to a model, these are:

- The modes on the left and right of each transition must be different — there are no *to-self* mappings;[1]
- All modes and events must be defined to be in the same band — band consistency;
- There must be no "ambiguity" in the events — equivalence between events could lead to ambiguities in the mapping, e.g. $(E_1, M_1) \mapsto M_2$ and $(E_2, M_1) \mapsto M_3$ are incompatible if $E_1 \equiv E_2$ — event uniqueness.

[1] To-self mappings are often shown in state diagrams where the action of the machine occurs on the transition between states, in the case of modes the position is that all visible behaviours result from being in a mode when viewed in the band of the model; the transition is simply an event.

Taking account of the above constraints the set of transitions may be refined thus:

$$\mathcal{T} : (\mathcal{E} \times \mathcal{M}) \nrightarrow \mathcal{M}$$

$$\begin{aligned}
&(\forall\, t_d : \mathrm{dom}(\mathcal{T});\ t_r : \mathrm{ran}(\mathcal{T}) \bullet \mathit{second}(t_d) \neq t_r) \land \\
&(\exists\, b : \mathcal{B} \bullet \forall\, t_d : \mathrm{dom}(\mathcal{T});\ t_r : \mathrm{ran}(\mathcal{T}) \bullet \\
&\quad \mathit{band}(\mathit{first}(t_d)) = \mathit{band}(\mathit{second}(t_d)) = \mathit{band}(t_r) = b) \land \\
&(\forall\, t_1, t_2 : \mathrm{dom}(\mathcal{T}) \mid \mathit{first}(t_1) \neq \mathit{first}(t_2) \land (\mathit{second}(t_1) = \mathit{second}(t_2) \bullet \\
&\quad \neg\, (\mathit{first}(t_1) \equiv \mathit{first}(t_2)))
\end{aligned}$$

The constraints address the issues identified in the previous section namely:

- Lack of to-self transitions
  $\forall\, t_d : \mathrm{dom}(\mathcal{T});\ t_r : \mathrm{ran}(\mathcal{T}) \bullet \mathit{second}(t_d) \neq t_r$

- Band consistency for modes and transitional events:
  $\exists\, b : \mathcal{B} \bullet \forall\, t_d : \mathrm{dom}(\mathcal{T});\ t_r : \mathrm{ran}(\mathcal{T}) \bullet$
  $\mathit{band}(\mathit{first}(t_d)) = \mathit{band}(\mathit{second}(t_d)) = \mathit{band}(t_r) = b$

- Event uniqueness
  $\forall\, t_1, t_2 : \mathrm{dom}(\mathcal{T}) \mid \mathit{first}(t_1) \neq \mathit{first}(t_2) \land (\mathit{second}(t_1) = \mathit{second}(t_2) \bullet$
  $\neg\, (\mathit{first}(t_1) \equiv \mathit{first}(t_2))$

It is convenient to also be able to define the band of a set of transition; given the above definition ensures band consistency, this is given by:

$$\mathit{band} : \mathcal{T} \rightarrow \mathcal{B}$$

$$\begin{aligned}
&\exists\, b : \mathcal{B} \bullet \forall\, t_d : \mathrm{dom}(\mathcal{T});\ t_r : \mathrm{ran}(\mathcal{T}) \bullet \\
&\quad \mathit{band}(\mathit{first}(t_d)) = \mathit{band}(\mathit{second}(t_d)) = \mathit{band}(t_r) = b \Leftrightarrow \mathit{band}(t)
\end{aligned}$$

There are additional constraints that are also required; these are:

- Completeness — all the modes of the model must exist on the left and/or the right,

- Reachability — all of the modes must be the destination of a transition, unless they are defined as being some form of initial mode.

However these may only be stated at the level of the mode model where all the modes are identified, and this is the next aspect to be defined.

4.1.3 The overall mode model

Having defined the modes and the transitions these definitions can be used in the construction of the overall mode model. In addition to the possible modes and the transitions the current mode is also captured, whilst the band is included purely to simplify the expression of the constraints. Using $\mathcal{M}$ to denote the universal mode type, the Z schema describing the mode model is:

ModeModel

$$
\begin{array}{l}
modes : \mathbb{P}\,\mathcal{M} \\
band : \mathcal{B} \\
transitions : \mathcal{T} \\
current : \mathcal{M} \\
\hline
\forall\, m : modes \bullet band(m) = band = band(transitions) \\
\forall\, m : modes \bullet \exists\, t_d : \mathrm{dom}(transitions);\ t_r : \mathrm{ran}(transitions) \bullet \\
\quad (second(t_d) = m \lor t_r = m) \\
\forall\, m : modes \bullet \exists\, t : transitions \bullet second(first(t)) = m \lor second(t) = m \\
current \in modes
\end{array}
$$

For convenience the *ModeModel* schema will also be referred to as $\mathcal{M}^{\mathcal{M}}$.

The constraints cover the issues of band consistency and those identified at the end of the previous section, namely:

- Completeness
  $\forall\, m : modes \bullet \exists\, t_d : \mathrm{dom}(transitions);\ t_r : \mathrm{ran}(transitions) \bullet$
  $(second(t_d) = m \lor t_r = m)$

- Reachability
  $\forall\, m : modes \bullet \exists\, t : transitions \bullet second(first(t)) = m \lor second(t) = m$
  This is perhaps a little weak in that it is not ensuring that the modes are reachable transitively, this issue will be addressed later.

In addition the final constraint $current \in modes$ identifies that the current mode must be a member in the set *modes* for this model.

The current mode of the model can be accessed through:

$$CurrentMode == \lambda\, ModeModel \bullet current$$

Considering the case of a mode change operation caused by an event, the delta schema defining this operation is given by:

ValidModeChange

$$\Delta ModeModel$$
$$e? : \mathcal{E}$$

$$(e?, current) \in \operatorname{dom} transitions$$
$$current' = transitions(e?, current)$$
$$modes' = modes$$
$$transitions' = transitions$$
$$band' = band$$

Here the constraint ensures that for the given event and current mode pair there is a possible transition. If this is the case then the current mode is updated to reflect this. In all cases the modes, transitions, and the band are invariant, thus this schema only applies to the case where the event causes a transition relative to the current mode; there is also another case to be accounted for, that where the event does not cause a mode transition given the current mode.[2] This circumstance is defined by the schema:

NoModeChange

$$\Xi ModeModel$$
$$e? : \mathcal{E}$$

$$(e?, current) \notin \operatorname{dom} transitions$$

Thus the overall schema for defining the effect of an event on the model is given by:

$$PossibleModeChange \mathrel{\widehat{=}} ValidModeChange \lor NoModeChange$$

Having developed the formalism for the individual modes, the transitions and the overall mode model, these can now be considered with a view to the use of modes in particular as constraints on other aspects of the model.

[2]Actually the event may never give rise to the mode transition, and may even be in a different band — but the important thing here is that it does not does not result in a mode transition given the current condition.

4.2 Using modes as constraints — invariants, and pre- and post-conditions

There are a number of ways in which modes may be used as constraints on other aspects of the timebands model; these include:

- Pre-conditions on the start of an activity;
- Post-conditions on the termination of an activity;
- Invariants which hold over the duration of an activity,
- Invariants which hold for the occurrences of an event, both within a given mode and explicitly not within a mode — two quite distinct cases.

In this section, each of these will be considered informally before providing formal descriptions for each case.

4.2.1 Informal description

It might be considered that the invariant over the duration of an activity is superfluous, and could be replaced by the use of both the pre- and post-condition over the activity — however this is not the case. Stating that the activity A only starts in mode **M** and the mode at its termination is also mode **M** does not prevent there from being a change to some other mode for a time before reverting to mode **M** prior to the termination of activity A — quite different from the case of the invariant which states that activity A only executes in mode **M**.

In addition to the "obvious" applications of these constraints there are others which are less-so. Simple negations can capture modes in which activities cannot start or end, or ever execute (in this case including transient execution within a mode). A further use could be to capture the fact that whilst the activity may be able to execute in any mode it does not execute during, or over, an inter-mode transition.

When considering the cases of the pre- and post-conditions it becomes clear that these are special cases of the invariant applied to an event. As the start and end of an activity is defined by events, the modal pre-condition becomes an invariant on the start event, the system is always in mode **M** for the any occurrence of the start of activity A; and likewise for the modal post-condition the system is always in mode **M** for any occurrence of the end of activity A.

Clearly these simple constraints provide a very flexible framework of relating activities and events to modes, but they are not without limitations; the first of which concerns the use of the modes as constraints on entities within other bands. In this case, in order that a constraint may be applied to an event or activity, there must be an ordering between the band of the artefact and that of the mode; without such an ordering it is not possible to relate the two bands in any way, and thus not possible to state the true meaning (in temporal terms) of the constraint. In practice this means that it is only possible to constrain directly the elements defined in the same band as the mode. Thus when considering activities in lower bands, it is the event "projected" into the modal band from the activity in the lower band that is constrained rather than the activity itself — thus the only constraint on such an activity is the invariant over mode, and the pre- and post-conditions are not possible constraints.

The next issue is that relating to the use of constraints with events; in this case only the invariant, or its negation is possible — as events start and end at nominally the same instance it will always be the case that the mode at the time of the start event will be the same as the mode at and after the end event. In the case of the mode and the event being in the same band only two possible situations are possible. Either the event occurs in a given mode; or it is a transitional event between two modes, in which case it occurs when the system is not in a defined mode for the mode-model of that band. There is something of a paradox here. The event defines the end of one mode and start of another, which, superficially at least, appears to indicate that the system is in two modes at the same time — so is the system in no mode or multiple modes at the point of transition? However, in the formal definition above, the system (once "started") is always in a single mode. Although these appear to be inconsistent, there is a relatively simple view, which solves the problem: the equivalence of the transitional event and the start and end events for the two modes indicates that the event is in fact modeless.

In the lower band, the constrained event will map to an activity,[3] here the constraint is transformed from an invariant on the event to one over the activity. Within the band of this activity the invariant will also apply to other events and activities which can said to be "within" the constrained activity — i.e. those whose start and end events are "contained" within those of the constrained activity. Taking this a step further it is possible, in this lower band, to infer further constraints; for instance, if another activity's start event occurs within the constrained activity a pre-condition of the mode on that activity may be inferred — likewise with the end event and a post-condition. So long as a mapping between the bands exists then the constraints can be used down through the model.

[3]Or an event — however this case is unimportant here.

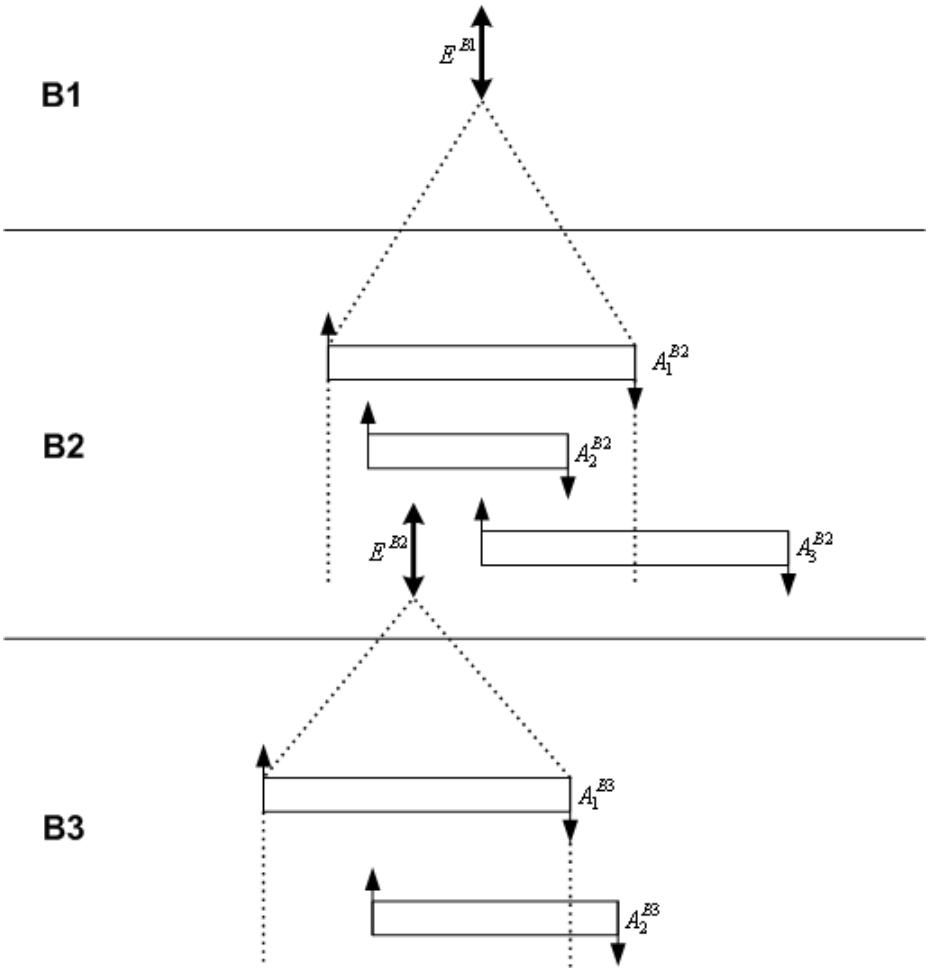


Figure 4.2: Example of cross-band constraint mapping

As an illustration of this transfer of constraints, Figure 4.2 depicts a simple model consisting three ordered bands, `B1`, `B2` and `B3`. The uppermost band, `B1` is the modal band, and contains a single event, E^{B1}. This results from, or maps to, an activity, A_1^{B2}, in the next lower band, `B2`; also in this lower band are other activities, A_2^{B2} and A_3^{B2}, and an event E^{B2}. A further lower band, `B3`, is also defined; this contains two activities A_1^{B3} and A_2^{B3} — the former of which maps to the event E^{B2} in the next higher band, i.e. `B2`.

Given the event E^{B1} is constrained to occur within mode M (defined in `B1`), it follows that activity A_1^{B2} is also constrained to occur within mode M. Following on from this, as activity A_2^{B2} and event E^{B2} can be seen to be "within" A_1^{B2}, they too are constrained to occur within mode M. However when activity A_3^{B2} is considered, it can only be seen to start "within" activity A_1^{B2}, hence here the constraint is weakened to a pre-condition on Mode M.

Finally considering at the lowest band, `B3`, the activity A_1^{B3}, the cause of the event E^{B2}, is also constrained to occur within mode M; activity A_2^{B3} starts "within" A_1^{B3} but completes outside of its bounds hence the constraint is weakened to a pre-condition. However here

there are other considerations, which might allow the constraint to actually be stated as an invariant — that of the granularity relationship between the lowest and highest bands and the precision of the highest band. If the duration of A_2^{B3} were to be less than the precision of the uppermost band then it can be argued that the constraint can be strengthened; whether or not the same applies to the case where the duration of the activity is less than the granularity of the upper band depends on the precise numbers and thus cannot be generalised.

The final issue relates to the mapping of modes to elements of the timebands model in higher bands; here there is no mapping (in the correct direction) and hence no modal constraint mappings may be defined.

4.2.2 Formal description

Having considered the constraints informally their formal description may now be developed. The notations used for the five different constraints are:

- Activity pre-condition — $\mathbb{P}\,\mathcal{M} \overset{pre}{\Rrightarrow} \mathbb{P}\,\mathcal{A}$;
- Activity post-condition — $\mathbb{P}\,\mathcal{M} \overset{post}{\Rrightarrow} \mathbb{P}\,\mathcal{A}$;
- Invariant over the duration of an activity — $\mathbb{P}\,\mathcal{M} \overset{inv}{\Rrightarrow} \mathbb{P}\,\mathcal{A}$;
- Invariant over the occurrence of an event — $\mathbb{P}\,\mathcal{M} \Rrightarrow \mathbb{P}\,\mathcal{E}$;
- "Negated" invariant over the occurrence of an event, the event is restricted to not occur in any of the given modes — $\mathbb{P}\,\mathcal{M} \nRightarrow \mathbb{P}\,\mathcal{E}$.

In each case the notation identifies a set of modes as constraints over a set of other elements; in outline the constraints identify precedence relationships between the entry to, and exit from the modes, and the start and end events of an activity, or a standalone event. As an example using $\uparrow\mathcal{M}$ and $\downarrow\mathcal{M}$ to indicate the entry to, and exit from a mode respectively, the following three informal equations give an indication of these relationships for a single mode and a single activity:

$$\mathcal{M} \overset{pre}{\Rrightarrow} \mathcal{A} == \uparrow\mathcal{M} \preceq \uparrow\mathcal{A} \preceq \downarrow M$$

$$\mathcal{M} \overset{post}{\Rrightarrow} \mathcal{A} == \uparrow\mathcal{M} \preceq \downarrow\mathcal{A} \preceq \downarrow M$$

$$\mathcal{M} \overset{inv}{\Rrightarrow} \mathcal{A} == \uparrow\mathcal{M} \preceq \uparrow\mathcal{A} \preceq \downarrow\mathcal{A} \preceq \downarrow M$$

Unlike activities, which have a single entry event and a single exit event, the entry to, and exit from a mode are defined by sets of events derived from the inter-mode transitions.

In order to be able to define the entry and exit events for a mode it is necessary to be able to identify the transitions relating to that mode model. This requires reference to the mode model of which the mode is a "member", hence there is a function to identify the mode model of a particular mode:

$$model : \mathcal{M} \rightarrow \mathcal{M}^{\mathcal{M}}$$

Given this definition the mode entry and exits may be defined thus:

$$\uparrow_ : \mathcal{M} \rightarrow \mathbb{P}\,\mathcal{E}$$
$$\downarrow_ : \mathcal{M} \rightarrow \mathbb{P}\,\mathcal{E}$$
$$\forall\, m : \mathcal{M} \bullet (\uparrow m = \{e : \mathcal{E} \mid \forall\, t : \mathrm{dom}\, model(m).transitions \mid m = second(t) \bullet first(x)\} \wedge \downarrow m = \{e : \mathcal{E} \mid \forall\, t : model(m).transitions \mid m = second(t) \bullet first(first(x))\})$$

Having defined the entry and exit event sets for a mode attention may now turn to the definition of the constraints, and given the earlier discussion regarding the equivalence between modal pre- and post-conditions on an activity as a whole, and invariants over its start and end events, the first element to be defined is that of the invariant applied to an event.

$$_ \overset{inv}{\Rrightarrow} _ : \mathbb{P}\,\mathcal{M} \times \mathbb{P}\,\mathcal{E}$$
$$\forall\, E : \mathbb{P}\,\mathcal{E};\ M : \mathbb{P}\,\mathcal{M} \bullet M \Rrightarrow E \Leftrightarrow (\forall\, e : E;\ m : M \mid band(m) = band(e) \bullet (\forall\, e_s : \uparrow m;\ e_t : \downarrow m \bullet e_s \preceq e \preceq e_t))$$

The second case is that where the event is constrained to not occur in a set of particular modes; the extreme case for this constraint is that for the inter-mode transitions, none of which cannot occur in any mode. This constraint is defined as:

$$_ \nRightarrow _ : \mathbb{P}\,\mathcal{M} \times \mathbb{P}\,\mathcal{E}$$
$$\forall\, M : \mathbb{P}\,\mathcal{M} \bullet \neg\ \exists\, E : \mathbb{P}\,\mathcal{E} \bullet M \nRightarrow E \Leftrightarrow (\forall\, e : E;\ m : M \mid band(m) = band(e) \bullet (\forall\, e_s : \uparrow m;\ e_t : \downarrow m \bullet e_s \prec e \prec e_t))$$

In this case it should be noted that the precedence relationships are defined to be strict; without this constraint it would not be possible to define the transitional events as being modeless.

Finally, following on from the constraints over the events, those over the activities can also be defined:

$$
\begin{array}{l}
_ \overset{pre}{\Rrightarrow} _ : \mathbb{P}\,\mathcal{M} \times \mathbb{P}\,\mathcal{A} \\
_ \overset{post}{\Rrightarrow} _ : \mathbb{P}\,\mathcal{M} \times \mathbb{P}\,\mathcal{A} \\
_ \overset{inv}{\Rrightarrow} _ : \mathbb{P}\,\mathcal{M} \times \mathbb{P}\,\mathcal{A} \\
\hline
\forall\, A : \mathbb{P}\,\mathcal{A};\ M : \mathbb{P}\,\mathcal{M} \bullet \\
\quad M \overset{pre}{\Rrightarrow} A \Leftrightarrow (\forall\, a : A;\ m : M \mid band(m) = band(a) \bullet \\
\qquad\qquad (\forall\, e_s : \uparrow m;\ e_t : \downarrow m \bullet e_s \preceq \uparrow a \preceq e_t)) \\
\quad M \overset{post}{\Rrightarrow} A \Leftrightarrow (\forall\, a : A;\ m : M \mid band(m) = band(a) \bullet \\
\qquad\qquad (\forall\, e_s : \uparrow m;\ e_t : \downarrow m \bullet e_s \preceq \downarrow a \preceq e_t)) \\
\quad M \overset{inv}{\Rrightarrow} A \Leftrightarrow (\forall\, a : A;\ m : M \mid band(m) = band(a) \bullet \\
\qquad\qquad (\forall\, e_s : \uparrow m;\ e_t : \downarrow m \bullet e_s \preceq \uparrow a \wedge \downarrow a \preceq e_t))
\end{array}
$$

4.3 Application of the formal approach to an example system

In order to demonstrate the extensions to the formal framework described above they will now be applied to an example system. There are two aspects to the application of this formal description to an example system:

- The description of the mode model;
- The use of the modes to constrain the other elements of the system.

These are described, in turn, in the sections below; however before these formal descriptions further details of the system are given.

The example system was originally outlined in the previous chapter, it is a simple semi-autonomous tracking/interceptor vehicle that when powered performs a built-in test operation before attempting to acquire a “target”. The operator of the system has some limited controls, including the ability to control the power, select and reject the item tracked by the system, and “launch” the vehicle; there is also the possibility of reverting the system to a built-in test operation at any time prior to flight. The vehicle can also autonomously select a target to track if launched without a target. The diagram given in Figure 4.3 give a detailed view of the modes of the system and the associated inter-mode transitions, whilst Table 4.1, originally given as Table 3.2 in the previous chapter, defines the characteristics of each band.

From the figure it is clear that the model has an element of hierarchy, with the *Unpowered* and *Powered* modes being at one level, and the *BIT, Acquisition, Track* and *Flight* modes

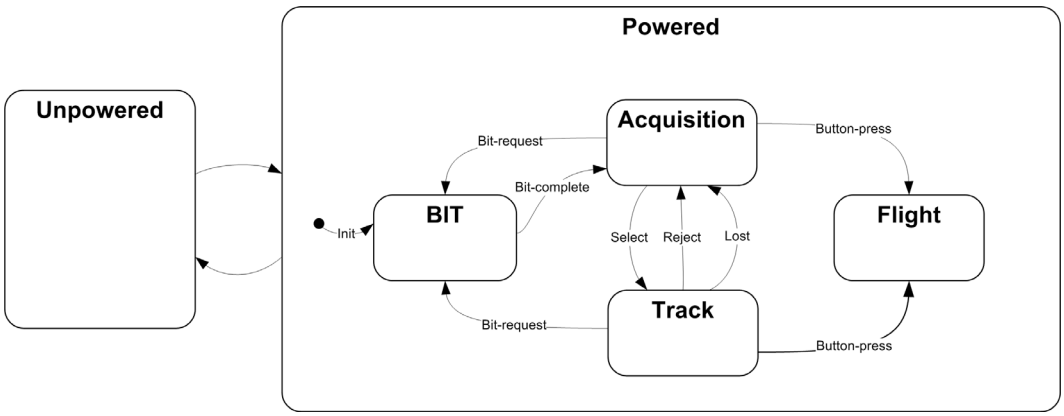


Figure 4.3: Mode diagram for example system

Band	Scale	Granularity (G)	Precision (ρ)
Sortie	10 hr	1/2 hr	3 min
Engagement	60 min	1 min	1 sec
Detection and monitoring	10 sec	50 ms	5ms
Stabilisation	500 ms	5 ms	250 μs
Thread	500 μs	10 μs	1 μs
Platform	5 μs	100 ns	—

Table 4.1: Precisions and granularities for the autonomous vehicle example

being contained within the **Powered** model. However, in the initial application of the formalism only the content of the **Powered** model will be considered — the hierarchical elements of this system will considered separately, later in this chapter.

4.3.1 Formal description of the mode model

The system under consideration consists of four modes: *BIT, Acquisition, Track* and *Flight*. The *BIT* mode is the "default" mode of the system, the one that is entered "immediately" at power-up. No transitions from the *Flight* mode are possible — this is the terminal mode of the system. The issues regarding the impact of the removal of power when the system is in the *BIT, Acquisition* or *Track* modes are ignored for now, but will be considered later as part of the discussion on modal hierarchy. In addition to the simple existence of the modes, the persistence of each also has to be captured; Table 4.2 details these values (all values are in seconds).

Clearly, given the lower limits of persistence of the *Acquisition, Track* and *Flight* modes, the model must exist in the **Detection and monitoring** band (or lower) — however the upper bounds for the *Acquisition* and *Track* modes clearly exceed the scale of the **Detection and monitoring** band, and are times that would be more usually considered in

Mode	Lower bound	Upper bound	Comment
BIT	15	15	Assumes that *BIT* cannot be aborted
Acquisition	1	36000	Upper bound is **Sortie** scale
Track	1	36000	Upper bound is **Sortie** scale
Flight	1	30	Dependent on power and intercept range

Table 4.2: Persistence bounds for modes

the context of the **Sortie** band. This is not a problem, but it does serve to highlight the likelihood, or even inevitability, of partial ordering of bands once modes are considered.

Moving on to define the elements of the mode, formally the modes are defined as:

$$VehicleModes == \{BIT, Acquisition, Track, Flight\}$$

With their persistence ranges for each mode being defined as:

$$persistence(BIT) == 15 .. 15$$
$$persistence(Acquisition) == 1 .. 36000$$
$$persistence(Track) == 1 .. 36000$$
$$persistence(Flight) == 1 .. 30$$

There are eight transitions in the system (excluding the init transition into *BIT*, again this will be considered later in the context of modal hierarchy) — however the number of events is only six, with two of the events giving rise to different transitions depending on the mode at the time of their occurrence.

The possible transitions are formally defined as:

$$\begin{aligned} VehicleTransitions == \{ & \\ & (BitComplete, BIT) \mapsto Acquisition, \\ & (BitRequest, Acquisition) \mapsto BIT, \\ & (BitRequest, Track) \mapsto BIT, \\ & (Select, Acquisition) \mapsto Track, \\ & (Reject, Track) \mapsto Acquisition, \\ & (Lost, Track) \mapsto Acquisition, \\ & (ButtonPress, Track) \mapsto Flight, \\ & (ButtonPress, Acquisition) \mapsto Flight\} \end{aligned}$$

Finally, following on from the modes and the transitions it is now possible to define the bindings for the overall mode model for the simple vehicle example:

$$
\begin{aligned}
&\mathit{SimpleVehicleModel} == (\!| \\
&\quad \mathit{modes} \rightsquigarrow \mathit{VehicleModes}, \\
&\quad \mathit{band} \rightsquigarrow \mathit{Detection\ and\ Monitoring}, \\
&\quad \mathit{transitions} \rightsquigarrow \mathit{VehicleTransitions}, \\
&\quad \mathit{current} \rightsquigarrow \mathit{BIT} |\!)
\end{aligned}
$$

4.3.2 Using the mode model to define constraints

Having a mode model of the system allows modal constraints to be placed on the other elements of the system. using this simple model a few of these can be considered.

Within the *BIT* mode there are a number of activities, the vast majority of which are constrained to only execute in the *BIT* mode, examples are:

- Test memory — Mtest
- Test peripherals — Ptest
- Report results — Repres

These are all of the $\mathbb{P}\,\mathcal{M} \overset{inv}{\Rrightarrow} \mathbb{P}\,\mathcal{A}$ form. Thus the invariant for these may be defined simply as:

$$\{\mathit{BIT}\} \overset{inv}{\Rrightarrow} \{\mathit{Mtest}, \mathit{Ptest}, \mathit{Repres}\}$$

There is, of course, the restriction that these activities do not execute in any of the other modes, i.e:

$$\neg\ (\{\mathit{Acquisition}, \mathit{Track}, \mathit{Flight}\} \overset{inv}{\Rrightarrow} \{\mathit{Mtest}, \mathit{Ptest}, \mathit{Repres}\})$$

Another example is that of a background health monitoring activity, HMon, that is initiated in the final stages of *BIT* and continues until the vehicle is in *Flight*; here there are two constraints, these are:

$$\{\mathit{BIT}\} \overset{pre}{\Rrightarrow} \{\mathit{HMon}\}$$
$$\{\mathit{Flight}\} \overset{post}{\Rrightarrow} \{\mathit{HMon}\}$$

Additionally it could be stated that the HMon activity is always executing during the both the *Acquisition* and *Track* modes.

$$\{\mathit{Acquisition}, \mathit{Track}\} \overset{inv}{\Rrightarrow} \{\mathit{HMon}\}$$

4.4 Hierarchical mode models

So far the mode model has been considered only in the context of the single band. Whilst it would be entirely possible to define a mode model in each band the relationships between these models have not been considered — currently they would have to be viewed as being entirely independent of one another. In this section the possibilities, and practicalities of having mode models in different bands related to each other are considered; this builds on the formal definition presented earlier in this chapter — as part of this development the issues of the init transition into the initial mode, and the termination of a lower-level model are also considered.

The fundamental issue here is the distinction between temporal and "behavioural" hierarchies. Clearly the timebands framework is focused on the temporal aspects, whilst modes generally relate strongly to the behaviour of the system — the question here is: is there a relationship between the behaviours at the different level of the architecture that is usefully reflected in the bands of the temporal hierarchy? In this section a simple hierarchical mode model is developed that allows the fundamental issues to be explored, and the temporal constraints between the levels of the model to be developed. The model used will again be that of the simple air vehicle described earlier; in this case the *Powered* and *Unpowered* modes will be at the highest level, with the *BIT, Acquisition, Track* and *Flight* modes being at the next lower level — in this case being seen as a model of the *Powered* mode.

There are a number of points to consider with respect to the mapping of this model to the time bands of the system; these are:

- The band of the higher level model — recalling from earlier that the lower-level model is defined in the **Detection and monitoring** band);
- The relationship between the mapping of the *Unpowered* to *Powered* transition in the higher model and the init transition into *BIT* mode in the lower model;
- The occurrence of the reverse transition, i.e. *Powered* to *Unpowered*, on the lower model;
- The general case of inter-mode transitions where each mode is mapped to an underlying mode model.

These issues are considered, both for this specific example model, and more generally in the sections below. As a basic principle, each higher band mode either maps to a complete

model in a lower band or to nothing — there is not partial mapping of a single mode to a part of a lower-level model.

4.4.1 Band allocation

Clearly the band of the higher level of the modal hierarchy cannot be in a lower time band than that of the component model; the key question here is: does the higher level model need to be in a higher band or can it be in the same band as the component model?

The answer to this depends largely on the rationale behind the creation of the hierarchy. If the hierarchy is merely a convenience to help describe the system then there would be no reason to enforce a band distinction between the two models. On the other hand however, if the models are focused on temporal properties of the system then it would be appropriate for the bands of the two models to be distinct. The accommodation of the former case is relatively simple in the formal sense — the simple use of schema composition would allow the models to be combined, though there is some need to distinguish "atomic" modes from ones which may decompose to more detailed models in another band.

Of more interest in the timebands context is the case where there is a clear temporal segregation between the models — here the temporal segregation strongly suggests that the models should be defined in different bands. Given the models are in different bands there are two related questions; these are:

- How should the bands for the two models be related?
- What are the relationships between the elements of the models at the two different levels?

Taking these in turn. In the first case, as there is a relationship between the elements of the models then it naturally follows that there should be a relationship (ordering) between the two bands. There is nothing to demand that the bands should either be, or not be, adjacent; also, at this point, there does not appear to be the need to impose the constraint that the ordering between the two bands is total. In the case of the simple example model used here, the higher level model is defined in the **Sortie** band, which only has a partial ordering with respect to the **Detection and monitoring** band.

The second issue, that of the relationships between the elements, also appears to be rather straightforward, but is perhaps best addressed in the light of the consideration of the init transition, and it is this aspect which is considered next.

4.4.2 The init “transition”

The init “transition” in the *Powered* model occurs on the transition between the *Unpowered* to *Powered* modes, the question here is: is this lower-level transition really a transition (mapped to an event), or is it actually an activity in the band of the lower-level model — i.e. not really a transition at this lower level? In the formal model the init “transition” has not yet been described, recalling that it is necessary for the duration of a “normal” inter-mode transition to be constrained by the precision of the band, then the duration of the initialisation would be no greater than five milliseconds, the precision of the **Detection and monitoring** band. This is clearly an unrealistic constraint, stabilisation of the power supplies alone might account for a period in excess of 100 milliseconds, and when other actions are correctly accounted for this could amount to something approaching half a second, a time easily in excess of the granularity of the band — 50 milliseconds; clearly this cannot be represented by an event.

A more reasonable approach here is to continue to view the *Unpowered* to *Powered* transition being a normal inter-mode transition, with the underlying duration of the event being less than the precision of the **Sortie** band, and to consider the init “transition” as being simply an event in the **Detection and monitoring** band, i.e. one with an underlying duration of less than the precision of that band.

Considering the **Sortie** band and the *Unpowered* to *Powered* transition, in the light of this constraint there is the requirement that the duration of this transition is less than the precision of the **Sortie** band (3 minutes) — clearly, in this case, the constraint is easily met. Interestingly in this model, mapping the higher model level to the **Engagement** band would have still allowed the constraint to be met. More generally, it may not be the case that it is possible to map adjacent levels of the modal hierarchy to adjacent band, and meet the requirement that a transition in the higher model always maps to an activity or event (in the lower band) whose duration is constrained by the precision of the higher band.

This consideration of the init transition highlights two things:

- There is a constraint on the duration of the init transition to be less than the precision of the band of the higher model, and less than the precision of its band if the init transition is to be represented by an event; and
- There is a need to include the init transition in the formal model.

The formal model will be extended to accommodate the init transition later in this chapter after consideration of the other aspects of higher level inter-mode transitions, the next of which is modal termination.

4.4.3 Mode model termination — Powered to Unpowered

At first sight mode termination (or term transition) is similar to model initiation — though it is somewhat implicit rather than being explicitly stated on the diagram. Assuming this similarity there is a requirement that the model at the lower-level is exited in a period less than the precision of the band in which the higher level model is defined, and less than the precision of the lower band if the termination is to be mapped to an event.

The obvious case to consider here is that of the *Powered* to *Unpowered* transition; the power may be removed from the system at any time, and for a number of different reasons. Clearly this event is potentially asynchronous with other system operations, and is not necessarily expected — thus it can change the mode of the system at any time. This differentiates the term event from that of the init, the latter is deterministic in that both the preceding and succeeding conditions are know *a priori*, whereas in the former the preceding condition is unknown for any given occurrence of the event.

Taking the *BIT* mode as an example: the mode will be "aborted" by the power-off transition, thus potentially resulting in a persistence which is in incompatible with the bounds defined for that mode. This leads to the question: how can the termination of modes be accommodated within the framework?

The persistence is a property of a mode, providing bounds for the minimum and maximum time the system may be in that mode for any occurrence. A simple way to accommodate the possibility of unpredictable mode termination events would be to have a lower persistence bound of zero, or not even state a lower bound. However, this does not seem particularly desirable as it potentially fails to capture the normal behaviour where the mode would not be exited until the lower bound had been exceeded. An alternative approach would be to consider the premature termination of a mode as an "abnormal" condition, and that the persistence only addresses the "normal" case. However, the real issue is not so much one of abnormality as one of separation, in these premature termination cases the actual mode change is not local to the model where termination occurs, but ultimately it is an expected (*cf.* abnormal) behaviour of the system. Yet another approach would be to add an additional property to the mode that indicates that it may be terminated (due to a higher level modal transition) prior to it lower persistence bound; when the attribute is not given then the mode must persist for the minimum bound of its persistence.

Perhaps what the various options above best indicate is that there is no simple way of capturing the premature termination of a mode in a formal sense. Given the premature termination of a mode is due to some (expected) transition in a higher level model then it should follow that the persistence of the higher level mode should, as a minimum, be "compatible" with those of the modes in the lower-level model. Thus the minimum persistence of the higher mode sets the maximum bound on the lower limit of the persistence for each of the modes in the lower-level model, and the maximum persistence sets the upper bound for the upper limit for each mode in the lower-level model. Thus in this case if the lower limit for persistence of the *BIT* mode is 15 seconds and the lower limit for persistence of the *Powered* mode is (say) one second, then the models are inconsistent. However, in a consistent model no transition could ever be considered to be unexpected, and there would be no violation of the persistences of lower-level modes.

Generalised hierarchical mode-to-mode transitions

There is a more general case which also should be considered: that of the mode to mode transition where each of the modes is mapped to a mode model in a lower band. In these cases it appears that the timing constraints have to be tightened.

In the init and term cases considered independently above the fundamental timing constraints are that the duration of either of these "transitions" should have a duration of bounded by the granularity of their containing band, and the precision of the higher band. However when considering the more general case it is clear that for each high level transition there is both an init and a term "transition"; this leads to the strengthening of the constraint to require that the sum of the durations of these two are less than granularity of the lower band, and less than or equal to the precision of the higher band; i.e:

$$\rho_{Bh} > G_{Bl} > \#init + \#term$$

This could be further complicated if the two lower-level models were not defined to be in the same band; in this case it the granularity constraint would have to be the finest defined for either of the bands; this strongly suggests that it is necessary for all lower-level models to be defined in the same band

4.4.4 Extensions to the formal model to accommodate hierarchy

In order to accommodate hierarchical models there is the need to extend the formal definition of the overall model. Here two additional items are required:

- Recognition of the init condition — the inclusion of a set of initiation event/mode pairs;

- Recognition of the term condition — the inclusion of set of events, which cause the model to terminate.

Following from these there is a need to provide mappings between the related models in the different bands. In the timebands framework in general the mappings from one band to another map from the lower to the higher bands thus an activity in one band can be seen as an event in a higher band, but has no real "meaning" in any lower band. With hierarchical mode models a mode at one level maps to a constituent part of a mode at a higher level, it is not necessary for the mode to map to a single event at the higher level. The only requirement is that the persistence of the overall model, and hence the persistence values of its constituent modes, are bounded by that of the higher level mode, i.e. the lower-level model does not come into existence prior to entry to the higher-level mode and it does not terminate after the higher level mode has exited.

Taking the individual mode model first, the schema for the model now becomes:

ModeModel

$$
\begin{array}{l}
modes : \mathbb{P}\, \mathcal{M} \\
band : \mathcal{B} \\
transitions : \mathcal{T} \\
current : \mathbb{P}\, \mathcal{M} \\
init : \mathcal{E} \leftrightarrow \mathcal{M} \\
term : \mathbb{P}\, \mathcal{E} \\
\hline
\forall\, m : modes \bullet band(m) = band = band(transitions) \\
\forall\, m : modes \bullet \exists\, t_d : \mathrm{dom}(transitions);\ t_r : \mathrm{ran}(transitions) \bullet \\
\quad (second(t_d) = m \lor t_r = m) \\
\forall\, m : (modes \setminus \mathrm{ran}(init)) \bullet \exists\, t : transitions \bullet second(t) = m \\
current \subseteq modes \\
\forall\, i : init \bullet (second\,(i) \in modes \land band(first(i)) = band \land first(i) \notin term) \\
\forall\, e : term \bullet ((band(e) = band) \land \\
\quad (\forall\, t : transitions \bullet second(first(t)) \neq e)) \\
\#current \leq 1
\end{array}
$$

The extra init relations and term set have been added to the signature, and the constraints are strengthened to ensure that the init and term events are all in the correct band, and that the possible init modes are also consistent with the overall set of modes, and by inference the band of the model. The modes that are included in the init relations are

used to address the weakness in the reachability constraint that was identified earlier. Finally it identifies that the term events must be disjoint with the init and transition events. It should be noted that the state of this schema is only valid within the bounds of the init and term "conditions".

In order to relate, or refine, modes and mode models at different levels of the hierarchy there are a number of constraints that have to be captured, namely that there is a mapping between the band of the mode model and that of the individual mode, and the persistence of the components of the mode model meet the constraints imposed by the higher level mode; this, using $\mathcal{M}^{\mathcal{M}}$to represent the *ModeModel*, is given by:

$$
\begin{array}{|l}
_ \sqsubset _ : \mathcal{M} \times \mathcal{M}^{\mathcal{M}} \\
\hline
\forall\, m : \mathcal{M};\ mm : \mathcal{M}^{\mathcal{M}} \bullet \\
\quad m \sqsubset mm \Leftrightarrow mm.band \sqsubseteq band(m) \land \\
\quad\quad (\forall\, mm_m : mm.modes \bullet \\
\quad\quad\quad \max(mm_m.persistence) \geq \max(persistence(m)) \land \\
\quad\quad\quad \min(mm_m.persistence) \geq \min(persistence(m)))
\end{array}
$$

The persistence values in each case (mode and mode model) need to be mapped back to a common time-base, in this case the granularity of the lower band (this is not shown above).

4.5 Concurrent modes

Following on from hierarchical modes the final mode related issue to be addressed is that of concurrent modes. As a first step it is worth considering what is meant by concurrent modes, and how they might be applied to systems in a timebands context.

Concurrent modes are ones in which the system can be simultaneously — as noted in the previous chapter this appears to run somewhat counter to the concept of the system mode being a fundamental behaviour of the system. Whilst the concept might not be particularly useful in the general case, when considering modal hierarchy, it might be useful to consider the models that relate to more detailed parts of the system being concurrent, but non-interacting, when considered in the context of the parent model. Thus the air vehicle system, from the earlier example, could be described as having two concurrent models: one relating to the sensing system, the other relating to flight control, as shown in Figure 4.4. These would be independent models and their mode changes would be unrelated, i.e. the mode of one element does not infer anything about the mode of another. There is however a risk with this approach: the models become focused

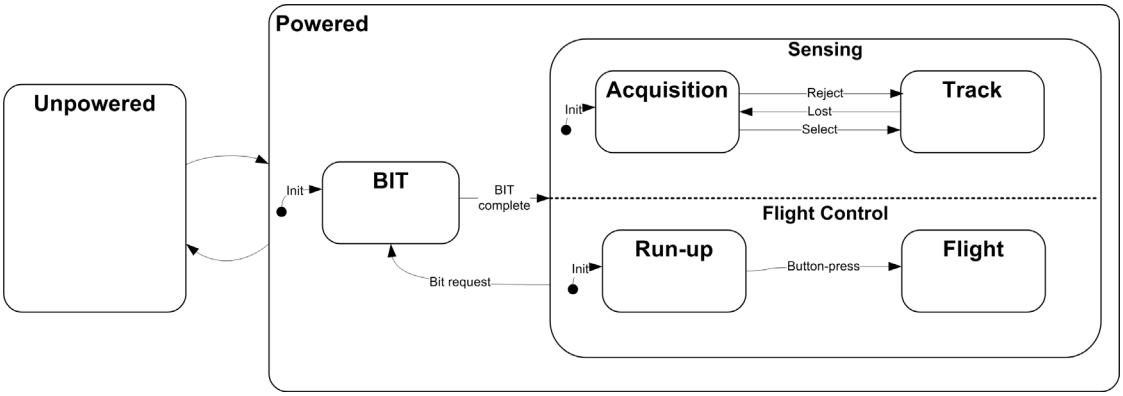


Figure 4.4: Example system using concurrent modes

on state, and describe potentially transient conditions rather than some fundamental, persistent behaviour.

Clearly, as can be seen in Figure 4.4 above, it is possible to describe the system in such a way but consideration of a single transition, that of *BIT Request*, exposes a significant issue. This transition forces the system into *BIT* mode at any time when the system is in the *Acquisition* or *Track* modes; in this concurrent model the **Sensing** element of the system will be in either of those modes for its entire existence, the **Flight Control** element of the system is either in the *Run-up* or *Flight* modes. Whilst it would be possible to return to *BIT* whilst the mode of this model is *Run-up*, it would not once the mode has changed to *Flight*. In this case the mode transition become conditional on the modes of the different elements, and results in the model becoming significantly more complex, and of debatable utility.

Perhaps, however, the most compelling justification for concurrent modes and mode models is the case where the overall system is a system-of-systems. Here each of the component systems will (potentially) have its own modes, and the transitions between these modes will be independent of each other; the systems were potentially developed independently, and can be viewed as being concurrent as the component systems are operating together.

If it is accepted that concurrent modes are a desirable concept, and hence there is a need to support concurrent mode models within the formal framework there are two important questions that need to be addressed; these are:

- Should the concurrent models be limited to being in the same band?
- Should the transitions between modes in concurrent model be coordinated?

At first sight it would appear sensible that the concurrent models should be restricted to exist within the same band, as the alternative would make the init and term conditions

across all models difficult to relate. However, when considering the system-of-systems case, this restriction appear to be excessive; in this case it is unlikely that the timebands model for each component system will be the same, thus if such a restriction were to be imposed it would simply not be possible to have these as concurrent models. It appears that if there are to be concurrent models then it is necessary that they are allocated to bands below the level of the overall model, and that there is a mapping from each of the lower bands to that of the overall model.

On the second question, that of whether mode transitions should be coordinated across the concurrent models, it is only when different transitions in separate models result from the same system-wide event that this consideration becomes an issue. Here the general constraint that the duration of the transition is less than or equal to the precision of the higher band is sufficient at this level, at the lower level the granularity constraint is also sufficient if all the concurrent models are defined to be in the same band. In the case where the lower-level models are defined in different bands then the granularity constraint should be strengthened to ensure that all transitions are completed within the minimum granularity of all of the bands applicable to the lower-level models. Whilst in general transitions due to a common event in the lower-level models will be rare there are two obvious cross-model transitions, these are the init and term transitions, which are universal to all lower-level models. Recalling the relationship between the granularity of the lower band and the precision of the higher band, in the case of the concurrent or distributed model, this must hold for all of the lower-level bands in which the models are defined, i.e:

$$\forall\, b : \mathit{LowerModelBands} \bullet G_b < \rho_{Bh}$$

Considering the case of systems-of-systems, it may be useful to distinguish between locally concurrent mode models and those of a more distributed nature, the latter of which would appear to be more applicable to a system-of-systems. In this case it is not realistic to expect all of the lower level models (in different systems) to be defined in the same band.

4.6 Summary

Developing the formal model and considering the issues raised in the hierarchical and concurrent models has helped to refine the overall concept of mode in a time band context. It is perhaps however useful to think of modes as being an approach to capturing the complex temporal relationships that exist at the major points of transformation in the existence of a system.

Whilst modes have historically been useful in certain domains there are two questions that need to be answered in the timebands context in particular; these are:

- Is modal hierarchy useful?
- Is modal concurrency useful?

In neither case is the answer simply yes or no.

There is certainly a case that can be made for modal hierarchy, even in the simple example it is clear that hierarchy helps in distinguishing between the *Powered* and *Unpowered* modes at one level, and the behaviours of the system in the *Powered* model at another. However there could be the temptation to take the hierarchy to a level where the important concept of a persistent behaviour was lost, and had been replaced by a state or function, i.e. not a mode at all, capturing some transient behaviour that might actually be better expressed in an activity in a lower band.

Moving on to consider the utility of concurrency; the system-of-systems issue strongly suggests that there is a case for modal concurrency; however, similar to hierarchy this could be taken too far. Whilst the issue of the init transition is relatively straightforward the concept of the term transition is less so, the issue here being the asynchronous nature of the models and the event; there is the need for some concept of coordinated event distribution, and an understanding of the temporal aspects of that distribution, which perhaps limits the utility of this approach.

The development and extension of the formal model of timebands to include a distinct notion of mode was relatively straightforward and accomplished with no detrimental impact to the original model. The extension of the model represents a useful contribution to the timebands framework; one that is necessary for its application to industrial sized problems.

The practical issues of modes will be returned to in the case study in Chapter 6; the next chapter will address the issues of the process of band selection and characterisation, in-part considering in which band, or bands, modes are best represented.

Chapter 5

Timebands: Creating a Model — Band Selection and Artefact Allocation

With the addition of modes, described across the two preceding chapters, the definitions of the elements of the timebands framework are complete in terms of both the properties of the elements themselves, and their basic interactions. However this completeness in itself does not directly support the development of the timebands models; there are four aspects to consider, these are the:

- Identification and selection of the appropriate bands for a particular system;
- Allocation of artefacts to those bands;
- Assessment of the utility of the models;
- Measurement and assessment of the model's quality once it has been produced.

Clearly all of these aspects need to be addressed if the utility of the timebands framework is to be maximised.

In this chapter, some of the fundamental issues associated with the development of a timebands model of a system are considered, largely from a theoretical viewpoint. Issues relating to the utility of the model will be considered as part of the case study and its evaluation in the following two chapters; the issues relating to the quality of the model, and the metrics that may be used in its assessment, are outside of the scope of this thesis

and are left as an area for future study. Here two fundamental and related aspects are considered:

- Band selection — what are the issues that affect the selection of the bands for the model. This proceeds to develop a general process to be followed for band selection.
- Artefact identification and allocation — what are the issues that affect the allocation of artefacts, the events, activities and modes, to particular bands, this is closely related to the previous aspect. As part of this, the issues regarding the selection or identification of the artefacts are also addressed.

In considering these issues possible structures for models of new systems, existing systems, and systems composed from other systems (systems-of-systems), with linear and divergent/convergent structures, are also considered.

5.1 Background

So far in this thesis the bands for the simple example models have been defined in a rather *ad hoc* manner; it does not seem likely that such an approach will scale well for larger, more complex models, and a more rigorous approach is required. In order to address the issues of band selection and artefact allocation it is first necessary to identify the particular elements/aspects of the framework that are central to these topics.

The bands each define their own abstract clocks, these provide the fundamental temporal reference for the bands, and their artefacts, in isolation from other bands; the bands are, in turn, related to one another through the concepts of granularity mappings and precision. These concepts also define the context for the artefact that exist within a single band in relation to artefacts in another. As an addition in this thesis the concept of scale has been proposed (in Section 2.5.2), this places an overall upper limit on the time that the band might encompass.

In defining the bands it is essential that the boundaries of the system are clearly identified — what are the highest and lowest bands of interests; following on from the identification of these bounding constraints the criteria that should be used for the selection of the "internal" bands can be defined. Neither of these steps is necessarily a simple or straightforward undertaking, which to some extent is due to the diversity in the original motivation for developing the timebands model for the particular system.

There are at least two contrasting scenarios for which a model may be developed:

- To provide a description of an existing system;

- To support the development of a new system.

In the former case the bands reflect the ultimate design/implementation, whereas in the latter are more likely to reflect the requirements and, progressively, the emerging design. These apparently simple scenarios can be further complicated by the motivation for developing the model; in the case of an existing system this could be to allow the system to be improved or extended, either of which will influence the model — however, it seems improbable that the same bands would be proposed for a single system viewed from each of these quite different perspectives. It also seems highly improbable that there will be a uniform structure that encompasses all systems; a simple system may, possibly, be described using a simple linear hierarchy, but in the case of complex multi-element systems it is unlikely that such a model would prove to be the appropriate approach to the problem of integrating a set of component models into a coherent whole.

In considering these issues the remainder of this chapter is split largely into three parts:

- The nature of the bands;
- The strategies for band selection, considering both new and existing systems, and the issues of models for systems-of-systems, ultimately defining a unified approach;
- The identification of the artefacts and their allocation to the bands.

Finally the elements of band selection and artefact allocations are brought together into a single process for creating a timebands model; this process is explored in depth through its application in the case study presented in the following chapter.

5.2 The nature of the bands

There is a fundamental principle regarding the bands: they must be related closely to temporal influences or dynamic properties of the system, and not simply be arbitrary scales of time; simply selecting an arbitrary scale and "forcing" the artefacts to fit will result in a distorted model. Considering Newell's work on cognition [70]; he identified a number of bands relating to human cognition and natural world effects; these were regularly spaced with a factor of 10^3 between each one. In his approach each of the bands was associated with a particular type of activity or class of action — this poses the initial questions:

- Is such a regularly spaced model universally applicable for the timebands approach?

- Is a factor of 10^3 between each band appropriate?

Recalling from Chapter 2, Newell's work was focused on exclusively human perception and considered temporal issues across a number of aspects, ranging from biological processes to social interactions and structures, with the natural world being considered at the outer reaches. This "human-centric" perspective is, in some senses, rather limited when compared to the issues of a complex socio-technical system where there are the added influences of the potentially diverse technical elements of the system, and its overall organisation. These aspects can be viewed as being in some way orthogonal to the single-human perspective. This orthogonality makes it more difficult, or even impossible, to determine the characteristics (range/scale, granularity and precision) of all of the bands from a single perspective, the technical and organisational perspectives also need to be given due weight. However, for systems where there is a significant element of single-human interaction, Newell's Cognitive and Rational bands are likely to be of particular interest.

The diversity of the human-centric, technological and organisation influences also tends to suggest that no regularly spaced model is likely to be universally applicable as the influences of each aspect will vary from system to system. However it is not unreasonable to assume that a semi-generic model could be developed that applies to a particular class of system or, alternatively, particular forms of implementation technology or organisational structures. At this point it is also worth recalling another difference between Newell's approach and that taken for the timebands framework: in the former there is a single system for the measurement of time — all durations are related to the second (fractions and multiples thereof, with an absolute mapping between bands). Contrastingly the timebands framework does not place this restriction, and allows for arbitrary discrete time "systems" in each band that do not, necessarily, have a precise mapping to one another.

Considering the weight of each of the human-centric, technological and organisation influences it is clear that each dominates the definition of the bands at different regions of the temporal spectrum. There appear to be three distinct regions of this temporal spectrum from the viewpoint of the selection of bands. Starting from the finest grained these are:

- The technological region — characterised by actions of electronic devices, machines or autonomic physical responses;
- The single-human region — characterised by cognition, actions and tasks (this can

be described as Newell's region);

- The organisational region — characterised by human-human interaction and collaborative decisions, moderated in some cases by the legislative framework and/or standards.

In the main, the regions are quite distinct, though the structure within each of the upper and lower regions will be affected by the nature of the system being modelled;[1] it is possible to view the bands as being subdivisions of the regions — though it is not immediately clear how useful this approach might be. However there is clearly a difficulty in determining the relative weightings at the points where the contributions from different influences are similar, i.e. at the boundaries between the regions. At such points the influences could be potentially ambiguous; these could, it appears, be addressed in any of several ways, including:

- The bands could be made very narrow — with each one focused on a particular aspect taken from a single domain. This could result in a very large number of bands, resulting in issues in the definition of appropriate vales of granularity and precision, and significant difficulty in consistently allocating artefacts to particular bands; thought this might be addressed to some extent by viewing the model as parallel hierarchies of bands (see below).

- The scale of each band could be made quite wide, but the artefacts are then constrained to only allow a subrange of its full scale; this will limit the number of bands but enforce clear distinctions between them.

- The bands could be made very wide — so wide that the influences of both areas are accommodated. The logical conclusion to this is, of course, that the models consists of a single band, which does not really address the problem, and defeats the timebands approach!

- Parallel bands for the different domains could be determined and then related through the other properties of the model; these would differ from one another in respect of their precisions, and their granularity relationships with other bands. This is possible as the bands within the model are only partially ordered, and thus there is no strict requirement to have a purely hierarchical model. However this introduces some complexity in defining rigorous inter-band relationships for all cases,

[1] The single-human region can be assumed to be largely constant, it is only really subject to slow evolutionary forces.

and probably does not solve the problem as at some point the relationships between the different areas (e.g. technical and human) will not be very well defined.

- A fixed set of regularly spaced bands could be used (as in Newell's model), however this suffers from the problem of defining the correct set of bands. The human and, possibly, organisational bands for a specific domain are relatively easy to define (simply following Newell's model would be a good starting point for the human band), and they are unlikely to need changes (human and social evolution are slow processes). However, the definition at the technical level would be more problematic as choice of technology could result in significantly different characteristics for the bands, and this is an area where the evolution is very rapid. To some degree, the same problem is true of the organisational end of the spectrum, whilst the organisations themselves may change relatively slowly (*cf.* technology) their interactions can change relatively suddenly due to, amongst other things, political and legislative changes. A further issue with defining a fixed set of organisational bands would be the issues of the scale and diversity of the organisation; small, simple, focused organisations are more likely to be accommodated by a static set of bands than are a larger, more complex, diverse ones — domain specific bands in this region might be required for a "fixed set of bands" approach to be a practical proposition.

Clearly, none of the above provides a simple panacea to the problem, and this ambiguity (or mismatch) "problem" tends to suggest again that it is unlikely that there will be a uniform spacing of the bands in any model that spans a wide temporal range. The problem is further complicated when the issues around systems-of-systems are considered as, in this case, each component system could have its own timebands model, resulting in potential incompatibilities and conflicts when the components are composed into a single system. It should be noted however that this is not so much a problem of the timebands framework itself — it simply highlights the general issues of constructing systems-of-systems. Indeed the fact that the timebands model can expose such problems suggests that it should also be able to help in their resolution — this will be investigated in the context of the case study in the next chapter. These issues are discussed in the sections below where the selection of bands for new and existing systems, and the issues of system-of-systems are considered in more detail, before ultimately proceeding to define a process for the selection of the bands for a model.

Finally, before moving on to consider the issues of band selection for each of the different cases, and leaving the specific issue of time for the moment, Newell also noted that, not only did his temporal scale increase in an exponential manner but that the same was also

true of the physical scale of the entities associated with each band. An example of this effect would be the neuron and neural circuit in the Biological band, and the operation and task in the Cognitive band. It might be interesting to consider the relevance of scale in band selection for timebands models, thought the concept of physical size might be difficult to apply in some cases — however this topic is outside the scope of this thesis.

5.3 Band selection

As noted in the introduction to this chapter the strategy for band selection is a rather unexplored aspect of the exploitation of the timebands framework. The limited case studies to-date have all considered the application of time bands to existing systems, none have considered the issues associated with the application of the framework to a system in the requirements or initial design phase. It is unlikely that considering just one of these perspectives will provide an answer to band selection, each of these basic scenarios needs to be considered. Given the diversity of both the types of system and the reasons for creating the models, it is perhaps convenient to subdivide the problem and consider the cases of new and existing monolithic systems, and that of systems-of-systems independently at first before attempting to "amalgamate" the results into an overall process for band selection.

5.3.1 Bands for a new system

In the case of the definition of the bands for a new system, it is the requirements of the system that are the dominant "selection" criteria — as fundamentally they provide the only information available. Often however there will be very few real requirements that need to be considered in terms of the time bands (i.e. those related to time) — but it is likely that more will emerge as the system is decomposed and developed. Typically, the requirements will be functional and will relate to how the system is deployed and used; these are the issues that are usually addressed in the higher (organisationally related) bands, leaving the lower (technologically related) bands to be defined during the design and implementation phases of the system development. A significant element of the deployment requirements on the system will relate to the environment in which the system will be used. This strongly suggests that the model of the system will encompass significant elements of the environment, which, in addition to any discrete events will also include elements exhibiting continuous behaviour. Whilst in the main this results in a very much top-down approach to band selection, and immediately gives some guidance in determining the highest limiting band; however, it should be remembered that environmental influences can potentially impact any level of the model. The treatment of the

continuous dynamics of the environment is discussed in more detail later in this section, and later again in relation to the identification of artefacts.

The requirements will often have not given any consideration to the time bands of the system, hence it is unlikely that there will be a clear mapping to any defined banded model; thus the requirements have to be analysed or interpreted to find the fundamental temporal "keys" that allow the selection of the bands. Small systems, those with few stakeholders all of whom share a common view of the system, should not pose too many issues in band selection — the bands should be largely self-apparent. However for larger systems, those with multiple stakeholders where there is a diversity of view of the system, the band selection process will be more problematic — there is clearly the possibility of the temporal requirements of the different the stakeholders clashing in some way. These potential clashes can be resolved in either of two ways:

- Refine or reformulate the requirements to get a higher degree of temporal consistency;
- Treat one stakeholder as being dominant, and modify the other requirements to fit this environment.

Given the difficulty in determining a set of requirements the first approach is unlikely to be easy, especially if taken to the extreme. The latter is also unlikely to be successful, the dominant stakeholder will inevitably change over the life of the system, and the life of the system might be considerable, for example in excess of fifty years[2] for a large, complex system such as a nuclear power station. Thus, it would not be possible to focus sensibly on a single area; clearly the model will only realise its full benefits if there is buy-in from all stakeholders. Another issue here is that all stakeholders are not equal, and neither is their interest in the architecture. Booch [8] notes that the operators of a system, who are clearly stakeholders in their own right, may be entirely unconcerned about the overall architecture or the technologies used to implement the solution. Their major interest is that the system does what is required on demand, and that it is usable. The same is clearly not true for the other stakeholders; some will have concerns regarding how the architecture will support long-term maintenance, others will be more interested in how the system might be evolved into different roles, and still others in how it might inter-operate with other systems. Whilst these interests are diverse they are not necessarily incompatible. It is important that the stakeholder "requirements" are kept as separate as possible — any unnecessary linkage between requirements will make it more difficult to construct a useful model.

[2] From concept design, through to completion of decommissioning.

At this point there is another aspect to consider, which is closely allied to that of the stakeholder: that of the legislative framework within which the system has to operate, e.g. restrictions on who may use the system, or any working time constraints. Does this, in effect, help to describe a nominal set of bands? At the organisational levels these issues will be of significant importance; at lower levels, however, these issues are generally of secondary importance and thus, rather than allowing these to drive the structure of the model, they should perhaps be considered as constraints against which the solution (the timebands model) is assessed.

If the requirements describe modes of operation then these can supply a focus for one (or more) of the bands, but again it is unlikely that the modes of the system will have been given any real consideration from a temporal perspective, except possibly in terms of constraints on the inter-mode transition times and the frequency of transitions. So naive use of modes to define a band without the context provided by the adjacent bands is unlikely to be successful, with the probability of them spanning several bands as a result.

In the absence of many explicit requirements it is perhaps useful to consider the different stakeholders as the fundamental "guide" for the definition of the higher bands in the system, as it is in these regions of the model that there is the most direct multi-human and organisational interaction. These stakeholders could include the procurers, maintainers and operators; each of which might have potentially different temporal view of the system — not necessarily when considered at the level of the individual, but more so when considered as organisations.

Undoubtedly there will be some overlap between the areas, but taking such an approach will allow the mismatch between these areas to be identified and resolved at the appropriate level minimising the risk of sinking to the lowest common denominator, and resolving the conflicts in the lower, finer-grained bands. Resolution of the conflicts through the formulation of the inter-band relationships will provide a structure at the top end of the model that provides a set of requirements and constraints for the design and implementation captured in the lower levels. This leads to the question: how to bring these two aspects of requirements and design together?

Arguably the modes of the systems could be that bridge in that they start to describe more detailed behaviours that the stakeholders expect of the system, which the lower levels have to implement. This does not restrict the models to having modes defined in a single band, it just requires that at least one of the bands has a significant modal content, and that the mode model in any band is complete. Clearly this approach is only suitable

for systems that have such a modal content, but as modality is common in large real-time embedded systems, even if in some cases the mode model is quite simple.

It is however important that the modes are derived largely from the higher level requirements, and are not overly influenced by the emergent detailed design and implementation. Complex modal models are usually the result of the definitions of the modes emerging from the solution (the way the system works) rather than being explicit in, or driven by, the requirements (what the system has to do). Often such complex models include modes that are transient in nature, thus making the model is difficult to contain in a single band; such transient modes would better be modelled as transitions in a coarser-grained band. However such issues may be readily identified as the potential mode model in these cases is likely to span a number of bands.

The lower levels of the model are largely concerned with issues in the design and implementation domains; there may be some requirements that permeate directly, and obviously, to these levels, but these will generally be few in number.[3] In defining the lower bands, there are two basic approaches that can be considered:

- The definitions could simply follow on from the bands defined at the higher levels becoming progressively finer grained;
- The bands could be defined from the bottom, the limiting case, and become coarser until the join with the bands at the higher end of the model.

There is a clear contrast between these approaches, the former could be viewed as a being system driven, whist the latter as being technology driven.

Neither of these approaches is without problem: in the first there is no preset limit to the granularity of the bands with no guidance on the level of detail — this allows the second bounding case, that of the lowest band, to "evolve". In the second the initial lower bounding case could be somewhat arbitrary, and, whilst this prevents the unbounded granularity problem, there is the issue of ensuring a "sensible" joint between the two structures. Another aspect that can assist in setting the lower level bands are the dynamics of continuous processes in the environment; whilst this approach may not be universally applicable, in systems that are control oriented these dynamics are an important influence. In the case of the technology driven approach, i.e. defining bands from the bottom upwards, the dynamics may help determine the lowest band.

[3]The requirements stated at these lower levels may not really be requirements, but very strict design constraints — arguably these might be better formulated as higher level requirements which the model will be able to demonstrate have been met by the implementation expressed in the lower levels.

Without a defined strategy there is a risk that the bands in this lower region will evolve (or even be defined) as the implementation proceeds; this is potentially little better than not having the model at all, or at least leaves the same problems as applying a banded model to an existing system, as described in the next section. In order to avoid this risk the band structure needs to be defined for these levels before any detailed design or implementation commences. The structure will, of course, be influenced by the fundamental design decisions, e.g. hardware or processing architecture, and by any residual system requirements, but it should be independent of the details of the operation of interfaces and execution of any software — the model should place "requirements" on the implementation, and not vice versa.

5.3.2 Bands for an existing system

In contrast with the bands for a new system, those for an existing system will be strongly influenced by the design and implementation of the system — to not accept this would be to ignore reality. Thus the original high level requirements, if indeed known or understood, may be of little importance to the definition of the bands for these systems. The important issue for this class of systems is the underlying motivation for generating the model; be it, at one end of the spectrum, to support the resolution of the obsolescence issues, through the extension of the system's own capabilities, to the integration into a greater overall system, at the other. Here, given the system exists, there might be the temptation to see the bands as being defined using a bottom-up approach, hence it would be the bottom bounding band that is the more easily defined. However this is not necessarily the correct approach for, as noted above, the motivation for creating the model is a significant driver in its own right — after all, the model is not being produced to simply describe the existing system, but is being produced to support the achievement of some other goal. Whilst a comprehensive understanding of the environment of the system was essential when considering the bands for a new system, the total environment is less significant in determining the bands for an existing system; in this case the environment only needs to be considered in respect of the motivation for creating the model.

There are many aspects that can, and need to be considered when creating the timebands model for an existing system, but perhaps more important than any specific aspect of the system itself is the issue of where to start. As suggested above, the bands could be driven from the bottom, i.e. by the technological solution, from the top, i.e. by the deployment and usage of the system, or finally from somewhere in the mid-range, i.e. by the interactions between the operators and the technology, and possibly the system modes. Any of these could be used, the "correct" one being potentially different for each system, or every view of a particular system.

Recalling the diverse reasons for applying time bands to an existing system can vary, four possibilities are:

- It might be to change the overall deployment and use of the system, potentially requiring a relatively complete model of the system;
- It might be to aid its integration with other systems, which might only require the higher bands in the system to be defined;
- Another option could be the resolution of a detailed hardware obsolescence issue — hence in this case only the lower bands are of interest;
- A final option might be to identify risks when considering making changes to an existing system.

From the above it is apparent that both the upper and lower bounding bands will depend on the reason for constructing the model — there is little to be gained in providing detail in the models that is of no interest to any party. Additionally however, the existing system may offer temporal affordances (as described by Conn [19]) that are fundamental to the way in which the various parts of the system interact, and as such have the potential to dictate, or at least limit the flexibility of, band selection and artefact placement. It is a reasonable starting point to suggest that the affordances offered by a system have to remain invariant over any modification or change in use; thus affordances have the potential to change the "logical" band of an artefact, and the freedom to move artefacts between bands may be restricted if the affordances are to be preserved. It is noted by Burns *et al.* [11] however that in some cases the affordances might be difficult to identify; it appears that their discovery is dependent on the close involvement of the stakeholders, in particular the users.

Depending on the level of detail required it might be possible to have a generic, partial framework that could be applied across a class of systems to address the lower bands. An example here could be the case of a software based real-time embedded systems where there is (currently) a set of bands that relate to the real-time behaviour of the software. These are then *Basic Block*, the *Thread*, and the *Transaction*. The precise characteristics, granularity, precision and range, of these bands vary from system to system — as will the lower boundary, anything finer-grained than this will be considered to be outside the scope of the description, i.e. something in the environment.

As an example using the above conceptual bands, and starting with the lowest:

- The *Basic Block* band could have a granularity of 100 ns, a precision of 10 ns[4] and a scale (or range maximum) of 50 µs;
- The *Thread* band could have a granularity of 10 µs, a precision of 1 µs and a scale of 200 µs;
- The *Transaction* band could have a granularity of 100 µs, a precision of 10 µs and scale of 50 ms.

Given the discussion earlier regarding mapping between artefacts in overlapping bands it would not be possible to have the totality of a thread with a duration of greater than 100 µs mapped as an element (event) in a transaction, and likewise a *Basic Block* of duration 50 µs mapped to an event in the *Thread* band. Interestingly, this example exposes the observation made by Newell regarding the increasing size of the artefacts as the bands are traversed, if, rather than time, the size of each of these is considered (e.g. by considering the number of source lines of code, or the size of the object code) there is an obvious progression in size from *Basic Block*, through *Thread*, to *Transaction*.

By having a partial framework such as this the issue of defining the lowest band is somewhat obviated, it being inherent in the above. An alternative more generic view would be to replace the concrete temporal references with a more abstract approach where the definition of the thread was in terms of units of *Basic Block*; thus the granularity of the thread with respect to the *Basic Block* could be defined would be a number of blocks, and the precision would be a single block. The same approach could then be followed for the granularity of the transactions with respect to the thread. Taking such an approach requires that concrete time is only considered directly at the lowest level, that is not to say however that the higher *Thread* and *Transaction* bands are invariant, this would only be the case where the type of technology is unchanging; thus in the case where the solution goes from a single, or small number of processors to a more highly parallel hardware solution it is likely that the number of threads will increase, and the "size" of each will decrease.

At the other end of the spectrum there is the case where the system is going to be integrated into a new environment, possibly as a component of a system-of-systems. In this case it is the higher bands that will be of interest and the model could be developed in a top-down manner — however the stakeholders in this case are unlikely to be those

[4]In this, the lowest band, the precision does not need to be defined.

related to the development of the original system, but those related to its novel use. In these circumstances there is no guarantee that the stakeholder levels that would have been appropriate to the context of the original system will be appropriate to the new one.

The above example again highlights the issue of the applicability of the model; the timebands model, as a whole, may only be appropriate to a specific set of circumstances, and if those circumstances change then the model may also require change. Certainly it is unlikely that the model in its totality will be appropriate (or of interest) for any particular stakeholder, or group of stakeholders. It is important to note here that different parts of the model will be invariant for different reasons. For example, the lower implementation layers will be invariant if the system is deployed as-is into a different environment; whilst the upper bands can be invariant even if the underlying implementation is changed to address an obsolescence issue.

The upper bounding band of the model depends, as with the lower bounding band, on the motivation for developing the model. As the motivation will be clear prior to the development of the model, it should be possible to define both bounding cases from the beginning; these bands will provide a context for the development of the remaining bands. The ability to set the bounds of the model from the outset for existing systems contrasts with the position for new systems — the diversity associated with these existing systems is the reason for constructing the model.

A final point to consider here is that of any legislative framework in which the system operates. As in the case of the new system this might point to some nominal bands that could be used as constraints on the final solution, but here, as with existing systems, the position is slightly more complex in that the relevant legislation may have changed since the system was originally developed[5] potentially changing fundamental characteristics of the system.

5.3.3 Systems-of-systems

In the two preceding sections the focus has been predominately on the single, unified system, however the application of timebands models also need to be considered in the context of systems-of-systems. The precise definition of a system-of-systems is somewhat "variable" in the literature, Jamshidi [50], and Sage and Cuppan [92] provide a comprehensive background and references on the subject. In this case the term is simply used to describe systems constructed from components that were themselves originally designed as independent systems, and whose use in such larger collaborations was not envisaged at their inception.

[5] Or the system is to be deployed in a different environment, with its own legislative framework.

There appear to be two fundamental approaches to applying the timebands framework to systems-of-systems; these are to either:

- Construct a single model appropriate to the entire system; or,
- Construct a set of models, each of which is appropriate to a component system, and integrate these nominally independent models as necessary.

Given the components of a system-of-systems are not usually developed at the same time as one another or, indeed, the overall system, there is little likelihood of the development of a single top-down, requirements based model of any real value — thus the construction of the components models can follow the approaches described in the previous sections. Furthermore it is fundamental to the nature of a system-of-systems that it evolves; components are added and replaced across a potentially protracted life, with the single original system diversifying to become a set of potentially significantly different systems. To a large degree these evolutions are unpredictable, making the development of a single model, which can address both the long term organisational issues and fragmented technology evolution in a satisfactory manner, somewhat impractical.

Equally, the modelling of each of the component systems independently is not without its problems. Having models of the individual components lacks the overall system context, and it not necessarily the case that a model developed for a particular use of the component system will be entirely applicable to its new use. This is compounded by the issue of relating the models individually to the requirements of the overall system — how can full coverage of the requirements be ensured?

There is a distinction that may be drawn between the higher and lower bands of the model, thus an approach here is to develop a model of the overall system based on the top-level requirements, and on the interactions between the component systems. This model is thus a hybrid driven by both requirements and implementations, though the depth of modelling of the implementation only need be go as far as is necessary to capture the component independent properties, in effect creating a framework. This framework model is then supplemented by models of each of the component systems. Clearly the component models must be compatible with the overall, but there is no absolute need to define the component completely unless it is something being developed as new, or being significantly modified; likewise there is no necessity to have a common set of bands across the detailed component models. An important aspect that will influence the overall model are the affordances inherent in each component — it is critical that these are captured correctly for the model to be useful, and where necessary reflected into the framework.

Once the models have been produced they have to be related to one another. This is not to say that all models have to be related to all others, but more that all of the component models need to be related to the overall; the granularity mappings are the bridge between the models. It is likely that there will be a degree of overlap between the overall model and one of a particular component, thus there might be a number of cross-related bands; it is therefore unlikely that there will be total ordering of the bands between the two models. If a consistent model is to be produced then it is essential that the issues common across a number of models are captured in the central model rather than be repeated in each of the relevant component models.[6] This begs the question: how can the two sets of bands be merged in a systematic and consistent manner?

The answer to this depends to a large extent on the scope of each of the models, if all of the common issues are captured in the central model then it is likely that the models can be brought together;. If however the common issues are repeated across the components, each with different band definitions, then it would seem unlikely that the approach would be successful in generating a useful model — again this is not an indication that the construction of the particular system-of-systems would itself be unsuccessful.

5.4 Band selection and model creation: a unified approach

In the previous sections the issues associated with, and the approaches to, the selection of bands for a model have been suggested and discussed. In simple terms it appears that there is little commonality in the approaches that are applicable to all cases. Clearly it would be useful, if not essential, to have a model that could evolve through the life of the system. However, given the above approaches, it is unlikely that a model developed for a new system prior to the start of development would be the same as, or even largely similar to, a model created for the final system once completed and in service. If there are two different models, then which would be the easier to use in the future modification of the product: the one focused on the requirements, or the one focused on the implementation?

Unfortunately the answer to this probably depends on how the system is to be evolved; if the changes relate to solving obsolescence or modifications to detailed functionality then it seems likely that the implementation based model may be more useful. Alternatively, if the changes relate to the deployment and use of the system, then the high-level requirement based model is more likely to be of use — however such a requirement based model would not capture any affordances inherent in the system. Clearly having a unified model

[6] If the issues are to be captured in the component models then strict relationships and orderings across all of the models are more likely to be necessary.

that addresses both domains equally would be the ideal solution; this leads to the two questions:

- Is it possible to construct such a model?
- Is it possible to define a process for the creation of a model?

These questions are clearly linked: if a process cannot be defined then it is unlikely that it will be possible to answer the construction question for all cases.

The remainder of this section attempts to answer these at first considering only the "simple" case of individual, monolithic systems, then moving on to consider the other issues related to systems-of-systems.

5.4.1 Individual monolithic systems

Previously, in Section 5.3, clear distinctions were drawn between the issues that drive the selection of bands for a new system and those that drive their selection for an existing system, in defining a unified process for creating the model these distinctions are not important in relation to the reasons for creating the model in the first place. A number of steps can be envisaged in the model creation process, the first three of which are:

1. Identify the context of the model — the reason or reasons for the production of the model, and the fundamental performance and deployment requirements;
2. Identify the stakeholders — identifying the issues important to the different elements of the organisation(s) that have an interest in the system;
3. Establish the boundaries of the model — setting the scope in terms of the upper and lower bands of the model.

At first sight the first step of defining the context appears to be relatively straightforward, and in the cases of the new development and the existing system being modelled in isolation that should usually be the case. Perhaps an issue here is the case of the existing system were it is being modelled for a number of diverse reasons, an option would be to create a number of independent models. The issue with truly independent models is that they can be very difficult to relate to one another, whilst in some circumstances this might not be an issue (initially at least) it would be useful to create a common framework for all of the models of that system. This could be achieved by developing each of the independent models from the output of the first three steps of the process, this would

give a number of constraints that should allow the models to be related if necessary — though possibly at the expense of making the resolution of the first steps somewhat more difficult.

Having identified the context of the model, the concerns of the stakeholders need to be considered. In this case it is not simply the identification of the stakeholders that is important, but more their interest in the model that is important — stakeholders in the system who have little or no interest in the model are not important to the construction of that model; it should be possible to relate the stakeholders directly to the context of the model. If stakeholders have diverse interests in the use of the model this could be another reason for building more than one model, but again it would be more useful in the longer term to resolve the issues and create a single model.

The next step, that of defining the bands at the boundaries of the model, sets the initial scope for the remainder of the model; as discussed in the previous sections these will depend on the system context. Selecting the lower band has, in the case of a new system, the potential to influence the lower level design and implementation, however the lower band only needs to be fine-grained enough to allow the initial model to be developed further bands may be added as the development progresses. It is more likely that the upper band can be fixed with some certainty as it only relates to broad characteristics of the system, and is usually firmly wedded to some organisation's behaviours — hence the need to determine the stakeholders prior to setting the scope.

Taken together, these first three steps serve to identify the environment of the system and as such need to identify the temporal influences, both discrete and continuous, inherent in that environment. As noted earlier, in the case of new systems this will result in a fairly complete model (though not fully "banded" at this point) of the environment, whereas for existing systems the model will be more focused. It should be noted that the setting of the context, identifying the stakeholders and defining the systems boundaries is a general 'systems' approach, not simply one particular to timebands, but with timebands the focus has to be on the temporal aspects.

The next steps depend more on the nature of the system, it may be possible in some cases for one or more steps to be omitted; the steps to be followed are:

4. Identify any modal behaviour of the system, if that exists;

5. Identify any tightly-coupled human interaction, if that exists;

6. Identify any bands between the modal and human interaction bands, if necessary;

The fourth step, that of defining the characteristics of a modal band presupposes that there is indeed some clear modal behaviour in the system — this will be the case in the majority of the complex individual systems, though it might not be clearly expressed in the requirements, or immediately apparent in the existing system. The important characteristics with regard to the modes are their persistence and the time taken for the inter-mode transitions. The latter must only appear as events in the modal band thus defining the upper limit on the precision of the fundamental clock of the band. Ideally the maximum persistence of a mode would define the upper bound on the band, however unlike the relationship between the minimum persistence and the clock this is not an absolute requirement, when other bands are defined it may be that there is some overlap between the upper limit of persistence and the clock of some higher band.

In some complex component systems, it may be that there is more than one band in which there is significant modal behaviour — where this is the case a number of separate and independent modal bands could be defined. It is important that the mode models are properly related, and that each model maps to a single band.

In socio-technical systems, tightly-coupled human-computer action will be common, and this is a useful aspect onto which to focus a band. Using Newell's model this would normally equate to the Cognitive and Rational task oriented bands; both of these bands might be present but initially identifying one will suffice. These are the bands where, when modelling existing systems, the temporal affordances are most likely to be identified. As with the modal band(s) it is important to identify the lower bound of the band, the fundamental clock; it is also important to consider it largely in isolation from the modal bands as at this point in building the model the bands are being considered as independent from one another. An obvious approach would be to use Newell's bands directly, this would result in a fixed set of bands in all timebands models of systems with single-human interactions. Whilst in some ways this is an attractive approach, it is also a very constraining one, having these bands in circumstances where there is only a restricted form of human interaction, or the interaction have been designed to give a very specific workload pattern could lead to a distorted or overly complex model, which fails to address the real temporal issues. Contrasting examples here would be the pilot of a high-speed military jet, and the operator of some simple machine — would these be best defined using the same banded structure?

If both the modal and tightly-coupled human interaction bands have been identified, it is worth considering the issues surrounding their integration at this point. It is unlikely that they will be widely separated, modes exhibit rather slow temporal dynamics but

there might be the need to insert a band if the spacing is considerable. More likely is the situation where the bands naturally overlap; in this case there are three possibilities:

- Amalgamate the bands to provide one single band;
- Redefine the bands so that they do not overlap — at least minimising any overlap;
- Leave the bands unchanged, and accept that they are only partially ordered.

Though the last approach is not really appropriate if the modal behaviour has a direct causal affect on the human activity.

The next two steps in the process are those that define the additional bands between the outer boundaries and the human/modal bands in the centre of the model, their order of execution is actually unimportant; these are:

7. Identify the remaining upper bands as driven by the organisation;
8. Identify the remaining lower bands as driven by the technologies employed.

The upper bands will be dominated by the organisations and stakeholders. In Western societies the bands will be based on the normal temporal frames of reference: centuries, decades, years, months, weeks, days and possibly hours; these will be tailored to meet the needs of the particular system. It is interesting to note that these bands can change significantly with different cultures, Levine [54] provides wide-ranging insight into this issue but its further consideration is outside the scope of this thesis.

The lower bands are influenced by the technologies of the implementation. If the system is considered in the broadest sense these lower bands will be influenced by a range of diverse technologies that have temporal dynamics that in some way overlap; in these cases it is important that the bands are defined such that they allow the appropriate interactions to be captured — this does not require that there is a single linear hierarchy, it is quite acceptable for there to be parallel, non-interacting bands in this region.

As the model is developed it is important to ensure that the scale of any particular band is not too small as this could lead to difficulties with the definition of granularities and precisions, and the subsequent allocation of artefacts, and vice versa.

The final step in the process concerns the relationships between the bands:

9. Determine the granularity relationships between the bands, and the precision for each band.

Up to this point the bands have been considered largely independently of one another, granularity mappings and precisions have not been rigorously addressed, only the scales of the bands are understood — these other more detailed aspects now need to be resolved.

The general principles for assigning granularities and precisions are:

- The granularity relationships between bands should be in the range of 0.5 .. 3.0 orders (decimal) of magnitude;
- The precision of a given band should be no finer than the clock (granularity) of the next lower band,[7] and preferably coarser;
- There needs to be a significant ratio between the precision and granularity of a band. The range of values of this ratio is not currently clear and will be investigated in the case study in the next chapter — though, unless there is good reason otherwise, a ratio of less than 10:1 between precision and granularity should be avoided.

The lowest band needs to be defined first, and then the properties of those above can be successively defined, with the upper band being the final one to be defined. This is, in effect, a band refinement process and it may be that this leads to changes in the number and nature of the bands. The lowest band in the system has no true granularity relationship with respect to any lower band, hence its properties are defined entirely with respect to some globally accepted temporal standard — there is no concept of precision for the lowest band. Whilst there is no prerequisite for any band to have particular characteristics, as noted earlier, in the case of systems with a significant element of human interaction it is likely that there will be one or two bands based on Newell's Cognitive and Rational bands; in these cases the granularity and precision can follow directly from his work, though the actual values may be refined. However, also as noted above, if human interaction is only a weak or machine-moderated aspect of the system these bands should not be allowed to dominate or distort the model.

When applying the process to an existing system, the issue of temporal affordances must be considered at every step, though these are more significant in some bands than in others. Whilst recognised as being difficult to identify, it is important that these are reflected in the model in order that any derived system continues to provide similar affordances, or an alternative means of gaining the same behaviour.

The bands of the resultant model do not necessarily have to be totally ordered, it is acceptable to allow bands to overlap. However the bands only being partially ordered

[7] Assuming that there is some relationship between the two bands.

will have an effect on how the activities of one band relate to the events of another, this issue is considered in more detail later in this chapter in the context of artefact allocation.

5.4.2 Systems-of-systems

Having considered the modelling of a simple, or standalone system the focus can now turn to the potentially more complex case of the system-of-systems. Given that a system-of-systems may be viewed as a system constructed from other independent systems, it is not unreasonable to assume that the first step would be to produce a model for each of the component systems in isolation from one another. However, whilst this appears to be an entirely plausible approach it is one that is unlikely to result in an optimal solution — largely because there is no overarching context or constraints to guide the development of the independent models.

A more appropriate approach requires that the context be set first, and the most appropriate way of achieving this is to first outline the overall system-of-systems model. To achieve this the first four, or five, steps of the process outlined earlier for the development of a simple model should be followed; from the identification of the context, through to the identification of the modal behaviour in the case of the four steps, or the tightly-coupled human interaction in the case of the five — there is little to be gained from the fifth step where there is little human interaction. Here, as for the case of the component system models, defining the environment (for the system-of-systems) is crucially important. The outline model sets some fairly firm constraints that can guide the development of the individual component models.

Following on from this initial system-of-systems model, models of the component systems are produced following the full process defined above. These models should be produced to be independent of one another, but must take due regard of the context set by the system-of-systems model. Once the component models are compete the system-of-systems model may be refined and “linked” to the component models — some artefacts in the system-of-systems model may be directly related to a single component, others might be the result of, or emerge from, the interactions between two or more components.

As an example of this process it is useful to consider the simple ATM system described in Chapter 2. This may be simply extended to provide a system-of-systems context through the inclusion of the wider banking network and central computer facilities, resulting in three components — ATM, network and central computers. As a first step a simple time-bands model of the overall system would be created; this would define a, probably small, set of bands in which all of the external interactions and those between the major components may be captured.

After the development of this initial model, models of a similar fidelity to that produced for the ATM in Chapter 2 would be created for each component, their bands being influenced, but not rigidly set, by those of the initial system-of-systems model. The actual fidelity of each of these models would be that necessary for its specific role in the overall system — this could limited to something a simple as interface behaviour. Finally the initial system-of-systems model would be refined, in the light of the component models, to give an overall model that identified the emergent behaviours and their origins.

The key to the development of the system-of-systems model and hence the overall model is identifying the simplest context of the system — i.e. the fundamental reasons for the development of the system-of-systems.

5.4.3 Band selection process summary

The band selection process consists of nine discrete steps that may be viewed in four groups: these are:

- The identification of the overall environment in the widest sense — the reasons for the generations of the model, the stakeholders and their interests and the definition of the temporal bounds of the model:

    1. Identify the context of the model — the reason or reasons for the production of the model, and the fundamental performance and deployment requirements;
    2. Identify the stakeholders — identifying the issues important to the different elements of the organisation(s) that have an interest in the system;
    3. Establish the boundaries of the model — setting the scope in terms of the upper and lower bands of the model.

- The identification and definition of the bands associated with two key influences: modal behaviour and human interactions; and their interactions:

    4. Identify any modal behaviour of the system, if that exists;
    5. Identify any tightly-coupled human interaction, if that exists;
    6. Identify any bands between the modal and human interaction bands, if necessary;

- The identification and definition of the remaining bands, those associated with technology and organisations:

    7. Identify the remaining upper bands as driven by the organisation;

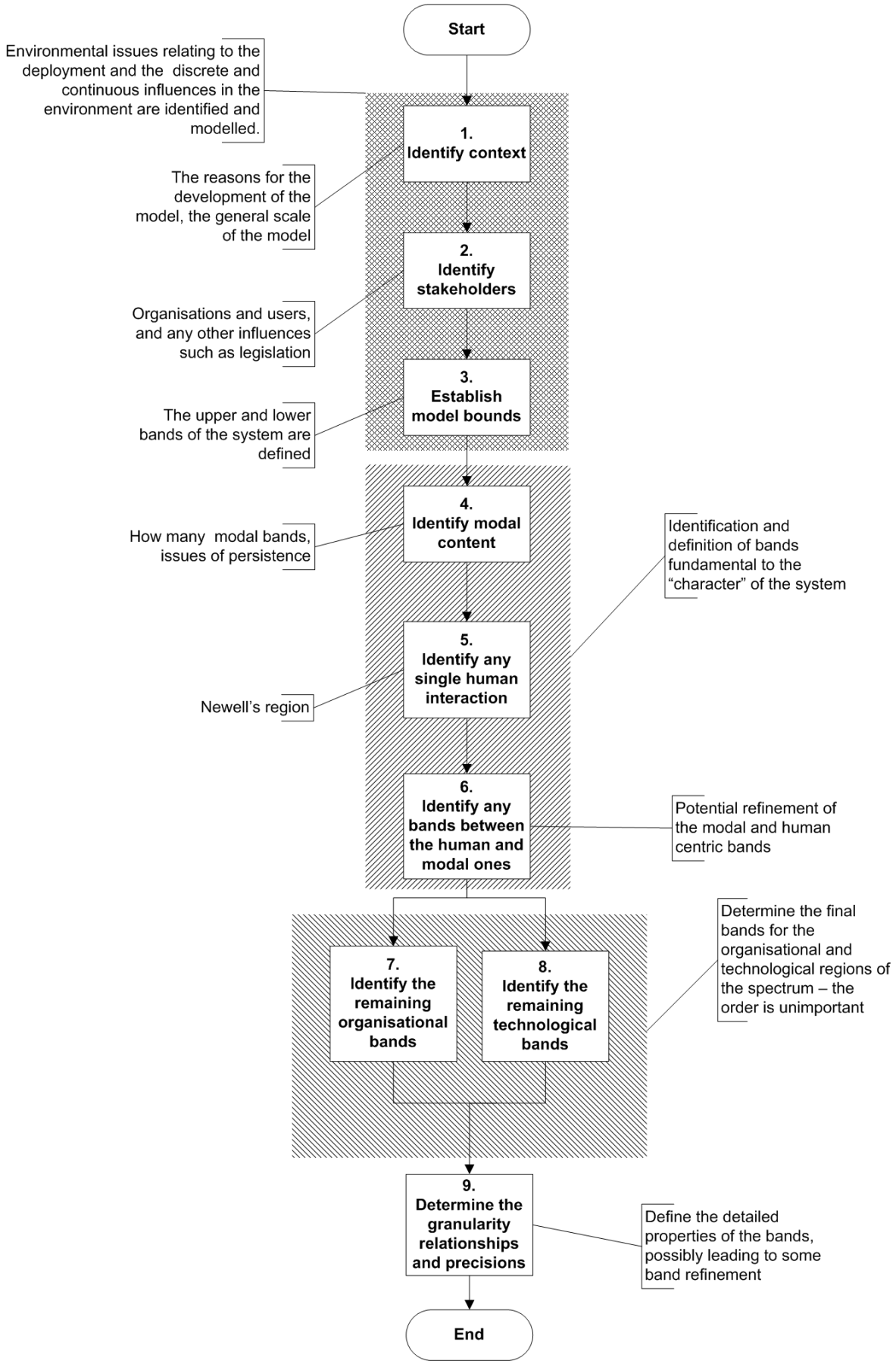


Figure 5.1: Band selection process

8. Identify the remaining lower bands as driven by the technologies employed.

- Cross relating the bands through the definition of the granularity relationships and precisions:

    9. Determine the granularity relationships between the bands, and the precision for each band.

The band selection process for a component models is also summarised in the flowchart given in Figure 5.1; the figure is annotated to indicate the issues to be addressed within each group, and at each individual step.

5.5 Allocation of artefacts to the bands

Band selection is only the first "step" in creating the timebands model. Having defined the bands, and their fundamental inter-relationships, the model can now be populated with the artefacts. Some of the issues regarding the allocation of the artefacts to the bands have already been touched upon in the previous sections; fundamentally there are three related aspects to be addressed, these are the identification of:

- The artefact — these can originate from the either 'requirements' (including the environment) or from design and implementation decisions;
- In which band should they reside; and
- How are they related, both within and between bands.

The identification of the artefacts and, perhaps more importantly, their interactions has the ability to modify the detailed characteristics of the bands in the model. Each of the aspects is covered, in turn, in the sections below. However before addressing these aspects in detail the impact of the partial ordering of bands needs to be considered as this can have a significant affect on the artefacts, their placement and interactions.

5.5.1 Impact of the partial ordering of bands

In the case of simple systems it is likely that the bands will be entirely distinct, and hence totally ordered. In such cases there will be no additional constraints placed on inter-band relationships. However when considering "real" or complex systems such simple examples are likely to be the exception, and it seems probable that it will be common for some of the bands to overlap to some degree and hence become partially ordered. Whilst

these overlaps may introduce some difficulties in the modelling process with respect to the placement of the artefacts, it is important that the bands are defined by the "natural dynamics" of the system, rather than attempt to use the bands to "define" these dynamics.

Such partial ordering requires that each inter-band activity-event mappings is considered individually with regard to their validity; this is because when bands overlap it will not always be the case that an activity in one band can be resolved to an event in the higher band — for the mapping to be valid each activity-event pair must satisfy constraint that the duration of the activity is less than the precision defined for the band containing the event. This makes the validity of the model vulnerable to parametric changes to the activities even if the characteristics of the bands themselves remain unchanged, this is an issue that has to be guarded against.

5.5.2 Artefact identification

At first sight artefact identification might appear to be a relatively straightforward process — and largely that is the case, it is certainly simpler than determining the inter-artefact relationships, or indeed selection of the bands themselves. However it is easier in the case of a system being defined, compared to that of the modelling of an existing system. There are three basic types of artefacts to be allocated, these are:

- The events;
- The activities;
- The modes.

Other elements of the model such as precedence relationships and *Behaviours* are more closely allied to the topic of artefact interaction and inter-relationships, and are considered later.

Clearly the different types of artefacts are not entirely independent of each other. When the case of the mode is considered, it is immediately apparent that events are required to describe the transitions to and from that mode. Likewise an activity is defined by start and stop events. Given this dependence on events, it appears that these are the most appropriate artefact to consider an allocate first. These will be followed, in turn, by the activities and the modes — though the ordering here is less significant, but as the activities are found in all bands, except perhaps the highest, and the modes will be limited to a small number (hopefully no more than one or two) of bands, the activities will be considered prior to the modes. It should be noted that to some degree the modes

have already been identified through the band selection process and that this has also identified, and hence placed, some events — those relating to inter-mode transitions.

Each of the three basic artefact types is now considered in the sections below.

Events

Events are, when considered in terms of the formal definition of the framework, a special case of activities — those with zero duration. However from a system modelling viewpoint they are quite different from activities, the lack of duration make these far more abstract entities. Clearly there are two classes of events: those resultant from other elements of the system; and those that originate in the environment. In fact, it is possible to consider the internal events themselves in two categories: those originating in the requirements; and those that emerge from the design and implementation.[8]

In principle at least, when considering events originating in the environment, an event may be introduced into the system in any band at or below the level of the lowest band in which the event has a direct interaction, either a causal effect or is part of a precedence relationship. This could lead to the temptation to introduce the events into the lowest band of the system and then flow them up to through the bands as necessary; this has the undesirable side effect of associating these events with excessively fine granularities and precisions — resulting in over specification of the system. A better approach is to introduce the event into a band where it has an interaction, thus restricting its precision accordingly. Where environmental events have an impact across a number of bands the event should be defined in each band where it is used — the desirability of mapping these events to one another is unclear and will be investigated as part of the case study.

In the case of the events related to other elements of the system these will have a precision directly related to the activity from which they emanate. Whilst it is not strictly necessary to "flow" the activity through all of the bands, the same issue with over specification of precision may exist if the event is allowed to "skip" bands; however it is equally incorrect to place artefacts in bands in which they have no interactions, or ones whose characteristics are out of step with the temporal dynamics of the artefact. In the case of events one of their characteristics that can indicate in which band they should be placed is their iteration rate, or frequency of occurrence.

Activities

In the context of the allocation process, an activity is nominally a process that has a finite duration. This finiteness of the duration poses a potential problem in terms of allocating

[8]It is recognised that in some cases it may be difficult to separate these fully.

processes that are essentially viewed as being continuous in a particular band; there are two basic categories of such processes:

- Those that operate continuously, monitoring the condition of some aspect of the system — i.e. those which are part of the system;
- Elements of the environment that change on a continuous basis — e.g. the weather, temperature, wind direction and speed etc.

A typical example of such a process, which is part of the system, would be a background health monitoring process that checks for anomalies in the overall system. Whilst such a process may be viewed as being continuous because it operates all the time, in reality it is unlikely that it will continuously produce information, which is in turn used by other parts of the system in some way — it is the way in which this information is produced and used that give an indication as to the effective "duration" of such activities and their frequencies of operation, and hence in which bands these processes should reside. This extreme case gives some useful insight into the definitions of activities in general — it is not so much the temporal behaviour of the activity itself that is important, but more the ways in which it interacts with events emanating from other elements of the system.

The case of continuous elements in the environment is somewhat different, these are truly continuous phenomena. In this case the system has to be able to respond to their changes, detecting variations required to give the correct levels of control in terms of fidelity and responsiveness — this is clearly a control engineering issue. The bandwidths and sampling rates required to give the correct level of control will, in turn, define the dynamics of the activities acting on these environmental stimuli, and hence determine the band in which they should be placed.

Modes

As noted above, the band of a mode is identified when the modal band itself is identified, all that remains in the allocation phase is to complete their definition.

Modes are unlike activities and events in that they do not have any duration as such. Rather than using the concept of duration modes use that of persistence, with the bounds of the persistence of a mode defining the minimum and maximum period that the system may stay in that mode. The inter-mode transitions are considered to be instantaneous in the band in which the associated modes are being modelled, and as such are modelled by events in that band.

The fundamental use of modes is to provide constraints over events and activities; events may, or may not occur in a given mode, and activities may start or end, or be constrained

to execute only in given modes. Clearly it is only possible to relate modes to activities in this way if the temporal relationships allow it.

5.5.3 Allocation to bands

To some extent the issues relating to the allocation of artefacts to bands are inherent in the identification of the artefact, have been consider in the sections above. The allocation of the activities and events to the bands is a relatively straightforward process, the only difficulties arise where the durations of activities in the lower band are close to the precision of the band in which they should be seen as an event, thus leading to the potential for band ambiguity. Where there is the possibility of this ambiguity, its solutions will depend in part at least on the inter-relationships between the artefacts. Where there are no inter-relationships there are fewer constraints on artefact allocation, however there are relationships that do impose constraints on allocation; these are:

- Activities that are mapped to single events in some higher band — the activity duration must be less than the precision of the higher band;
- Modes — Activities that are in some way constrained by a mode must exist in a band in which the precision of the mode timing is compatible with the activity duration. In practice this means that only activities that are in the modal or lower bands can be related in this way.

5.6 Artefact interactions and relationships

The final step in the construction of the model is the definition of the interactions and relationships between artefacts in and across the bands; these may be seen in a number of categories, and are:

- Precedence, activity and event orderings including causality;
- Formation of the *Behaviours*, groupings of activities and events within a band;
- Mapping of modal hierarchy;
- Inter-band mappings between events and activities, and potentially *Behaviours*;
- Modal conditionality of events and activities.

The first two are relatively straightforward in that they only exist within a band; these are the detail of the model rather than the structure and are the last issues to be considered

for each band. The likelihood that the model structure, the bands and their characteristics, will change as a result of the consideration of theses aspects should be low. These can be considered to be the intra-band interactions. The need to form *Behaviours* is optional, and currently the best use of these is unclear. Perhaps these can be used to group related constraints focused on a particular aspect, or stakeholder; thus different *Behaviours*, possibly overlapping, could be used to capture the constraints from different viewpoints.

The mapping of modal hierarchy, if it exists, is the first phase in the creation of the inter-band mappings. In the first instance there are the simple mode to mode model mappings, and this is followed by the mappings between the simple transitional events in the higher bands and the more complex interactions between the model models in the lower bands.

The last two items in the list can be considered to be the remainder of the potential inter-band interactions. In many cases, the definitions of these inter-band relationships will be straightforward; however in some cases where there is a partial ordering of the bands it may be difficult to define the correct relationships due to the limitations in the distinctions of the granularity and precision between the bands. In these, hopefully rare, cases it may be necessary to refine some of the basic characteristics of the bands. It would be wrong, or possibly naive, to assume that the process of creating the model was simply linear; as the bands are defined there will be iterations over the process, though this is not explicitly identified in the process defined earlier.

5.7 Summary

In this chapter, the issues surrounding time band selection, in particular, and model construction or creation, more generally, have been explored in some depth. It is clear that a single simple, linear model, such as the one presented by Newell for human cognitive processes, is unlikely to be adequate for describing multi-faceted complex systems comprised of diverse “technologies”, and even less so when the systems-of-systems are considered. It is therefore likely that there will only be a partial ordering of the bands in models of any significant complexity. The concepts of granularity and precision are the key elements of the timebands framework that need to be considered in construction of the model. The orthogonality between the technical and human aspects of the system make it difficult to define a single approach to developing a model; and the approaches appropriate to modelling an existing system, and a new one appear at first sight to be quite different — though the possibility of using fundamental system modes as a central starting point needs to be further considered and tested as a key element in a common approach.

Having considered the different reasons for constructing a model it appears that, for a given class of system, the shape of the model would be relatively static. The bands might be different in detail, the granularities and precisions are likely to change especially with regard to the technological levels of the model, but the overall set of bands and their ordering would be largely invariant. Thus over time, as experience with the framework and modelling increases, there is the significant possibility of producing generic models that can be tailored to different circumstances — given the observations arising from the development of ADLs it would not be surprising if these models were domain specific.

A general process for the creation of timebands models of single independent systems has been proposed, the basic steps of which are:

1. Define the bands:
   (a) Identify the context of the model, and the reasons for it production;
   (b) Identify the stakeholders — understanding their relative importance;
   (c) Establish the boundaries of the model — the upper and lower bounds;
   (d) Identify any modal behaviour [if any];
   (e) Identify tightly coupled human interaction [if any];
   (f) Identify the modal-human interactions [if any];
   (g) Identify and partition the remaining upper bands driven by the organisation;
   (h) Identify and partition the remaining lower bands driven by the technology;
   (i) Determine the granularity relationships between bands, and precision for each band.
2. Identify the artefacts and allocate these to the appropriate bands;
3. Identify the intra-band artefact relationships;
4. Identify the inter-band artefact relationships.

Although the process is described in a linear fashion there will inevitably be some iteration as the bands are developed and populated.

In the case of models of systems-of-systems the situation is more complex. Here the first step is to create an overall framework, an initial system-of-systems model, by following the first four or five (band definition) steps of the process; this is followed by the creation (if they do not already exist) of appropriate models of the component systems that can be related to the overall framework. The degree of detail captured in the various component

models will depend on the reasons for their creation, it may not be uniform across all of the models or even through all bands of a particular model. Finally the overall system-of-systems model is refined and linked to the component model.

Having developed a process for creating timebands models in a variety of circumstances this will now be used to develop the models required to describe a system in a significant case study described in the next chapter.

Chapter 6

Application of the Timebands Framework: A Case Study

In the previous two chapters the timebands framework has been extended in two ways: firstly through the inclusion of support for the concept of modes; and, secondly with the development of an approach, or process, for creating timebands models. However, in neither case have either of these extensions been applied to anything other than the simplest of examples; clearly this is insufficient to demonstrate that either significantly enhances the framework, or that the extended framework contributes to the overall goal, i.e. the development of better architectures for real-time embedded systems. In order to address these shortcomings in this chapter a more significant case study is developed; taking a large industrial (i.e. real), complex system a set of timebands models are constructed.

As is clear from the previous chapter the circumstances in which a timebands model might be developed can vary considerably; hence in order to evaluate fully the extended framework any case study has to encompass a wide cross section of these circumstances. This could, undeniably, be achieved through the development of a number of independent case studies, where each would address a particular set of circumstances, but this approach would still leave the broader issues regarding the integration of systems-of-systems largely unaddressed. By careful selection of the case study subject matter however it is possible to develop a set of models that address the three fundamental circumstances under which a model might be developed; namely:

1. An essentially new system developed to fulfil a specific role;
2. An existing system being extended or deployed in a different environment;
3. The integration of a number of independent component systems in a systems-of-systems context.

In order to address all of these circumstances the subject matter for the case study has to be quite diverse, this is achieved by developing it around a relatively large system consisting of a number of complex component systems and selecting the appropriate parts to be modelled.

In the next section, an overview of the system and it components is provided, setting the scene against which the models will be developed, after which the remainder of this chapter is divided to:

- Describe the approach taken in developing this case study — the “research method”;
- Develop an outline, initial system-of-systems model to set the context for the individual component models of interest;
- Develop the models for two of the individual component systems;
- Refine the initial system-of-systems model in the light of the component models to “provide” the final systems-of-systems model.

Issues regarding the effectiveness, or otherwise, of the framework in particular, and the strengths and weaknesses of the models created for this case study more generally, are considered in the next, evaluation, chapter.

6.1 System description

The subject matter of this case study is derived from a system consisting of a number of significant components each of which could be considered a system in its own right; these component systems are described in varying levels of detail in the sections below. As the case study focuses on two key component systems, namely the local command post and the payload delivery vehicle, these are described in some detail; the descriptions of the other elements, which may be considered part of the environment, are relatively brief. For the purposes of this case study there has been some simplification of the system in that functions that do not introduce additional timing interactions are not considered — thus the case study is only complete in relation to the fundamental timing properties of the system. Finally in this section, both the stakeholders and the environment are identified, for these, as noted in the previous chapter, are fundamental to the correct modelling of any system.

The system described here is based on a snapshot taken early in the formative stages of its development. It is common for such systems to evolve considerably over the course

of this stage of their development, as such there is divergence (substantial in places) between the description provided here and that of the system currently in development. The definition of system in the context of this case study is limited to the basic equipment and does not include the wider military context.

Before describing the component systems in detail an overview of each of the systems, together with its roles and environment, is provided.

6.1.1 General background and overall system

The system on which this case study is focused, is a new low-cost[1] military payload delivery (air-vehicle) system that allows for remote planning and management, with (more) local control of the particular mission. The system is described as, or designated, a *Loitering munition* (LM). Currently, in early 2009, the system is in the early stages of development, and further background information is given by Ghulam [35]. In common with the majority of complex military products this system is expected to have a prolonged life and, whilst the full extent of this is not yet defined, a period of up to forty years is not uncommon.

The primary purpose of the system is to allow the delivery of payloads, usually (but potentially not exclusively) explosive munitions, to precise targets in the battlespace — the precise delivery location will not generally be known at the time of launch, but it will be in an area designated a safe target zone — an area where there is zero possibility of collateral damage. The delivery of the payload has to be accurate both in terms of geographic position and time as the targets are often time sensitive.[2] In general the system will be operated in the field by relatively non-specialist military personnel, and will be handled in a similar manner to other currently deployed equipment. It is clear from this that there are diverse stakeholders even from the operation viewpoint, these include those directly involved with the use of the system, the operators, local commanders and armourers; and those in less directly connected such as the logistics and training organisations.

Whilst it is difficult, in terms of the physical and functional elements, to precisely define the bounds of this system, when considered from the perspective of creating the time-bands models the system can be seen to be comprised of three fundamental component systems; these are:

[1] Low cost in this context refers to total through life cost, and not simply that of the initial development, or unit purchase price.

[2] The target may be mobile and hence only be “available”, or “safe” to engage at a particular time, or in a particular area (the safe target zone).

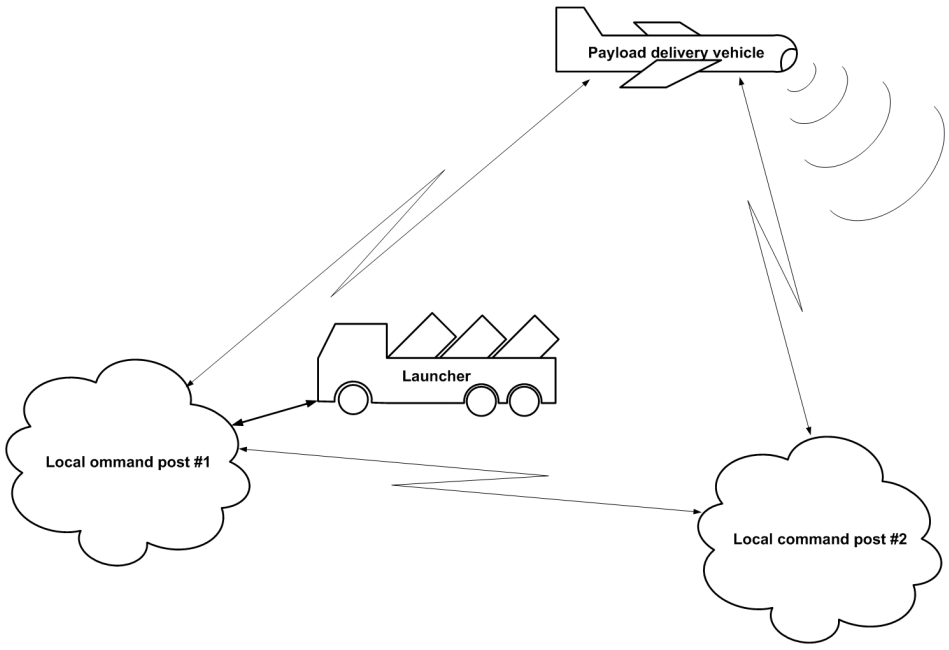


Figure 6.1: Schematic for overall system

- A payload delivery vehicle — a “simple”, and hence low-cost, air launched munition;
- A vehicle launch system — this is (based on) an existing facility, thus the vehicle needs to be designed to meet the requirements and constraints of this system;
- Local command post(s) — providing local control of the vehicle mission from launch through to mission completion, multiple such systems may be required to interact over the course of a mission.

An overall schematic of the system, showing its major components and their primary interactions, is given in Figure 6.1. In addition to the fundamental interactions between the components, the overall system has also to interact with, and use services provided by, other “systems”, These include systems such as airborne control centres; for the purpose of this case study these are assumed to part of the “environment” and are not considered (or modelled) in any detail.

6.1.2 The component systems

Each of the three fundamental component systems is described in more detail in the sections below, starting with the payload delivery vehicle

6.1.2.1 Payload delivery vehicle

The payload delivery vehicle is a simple air launched munition; its primary goal being to maximise the potential flight time and active payload capacity — clearly these are

immediately in some conflict as flight time is affected by payload mass. As a result of these goals the capacity (volume, mass, electrical power etc.) available for any sensing and processing systems is severely limited — resulting in the potential need for the vehicle and ground systems to collaborate on some of the slower-rate or latency-tolerant data processing tasks. The environment in which the vehicle is required to operate is not particularly harsh by military standards, where possible unmodified commercial grade equipment will be used in the final system. The basic payload delivery vehicle is the only component which is developed specifically as part of this system; the other components, the launcher and local command post, can be viewed either as existing systems or as modifications to existing systems — the payload delivery vehicle has to be designed to meet the constraints imposed by these systems.

The duration of the payload delivery vehicle's mission is up to ten hours, and for most of this period the vehicle is effectively autonomous, however it is required to report status information regularly to the local command post. The fundamental principles of operation are that the vehicle is pre-programmed with an initial flight plan and then launched; it then flies, on a course defined by a pre-loaded flight plan, to some holding point in the general target area where it "loiters". The vehicle is neither fast nor highly manoeuvrable, thus the transit from launch site to loitering area can take a considerable time.[3] When targets become available the local command post operators upload detailed target data and the vehicle proceeds to its target — there is the possibility of limited real-time operator interaction with the vehicle immediately prior to target engagement.[4] If no targets become available the munition will self-destruct (in the safe target zone) at the end of its mission time, or when so directed by the local command post operators.

Roles

Whilst the primary role of the vehicle is to deliver an explosive payload other potential future roles are:

- Acting as a communications relay between the local command post and another vehicle;
- Acting as a voice communications relay between base commanders and personnel in a forward area;

[3] A high degree of manoeuvrability is not seen as being particularly desirable as its exploitation will often result in inefficient fuel usage.

[4] This is the only time-critical point, any interaction prior to this does not have to be hard real-time — but temporal references between components have to be consistent.

- Acting as a battlefield sensor providing, for instance, real-time feedback on the effectiveness of the missions of other vehicles to the local command post operators, and strategic planners.

In each of these cases a different payload would be used. Clearly there is also the possibility of providing multiple roles, assuming sufficient payload capacity; for example, initially the vehicle may act as a battlefield sensor, before being directed to a specific target as an active munition. Whilst the desire is to minimise the cost of the system this has to be balanced by the cost of providing other discrete facilities — a single airframe undertaking multiple roles will be considerably more cost effective than having a unique solution to each problem, not to mention the simplification a single solution can give to the logistical aspects; however in order to accommodate such flexibility it is recognised that its initial development might be more costly.

Components

The vehicle itself consists of a number of components:

- An airframe with the necessary flight control surfaces and their actuation systems;
- A propulsion system — there are two parts: a launch booster and an in-flight motor;
- A flight control system to maintain stable and efficient flight, efficient flight allows for a greater active payload (*c.f.* fuel);
- A power system — batteries to provide initial power, and a generator for in-flight power;
- A navigation and guidance system — providing accurate positional information and producing directional demands for the control systems;
- Inertial measurement instruments — to provide vehicle attitude information used in the control and stabilisation of the airframe, and for use in position assessment for guidance and navigation;
- Other instruments to support the flight control system, e.g. airspeed indicator, altimeter etc;
- A computer system to host functions such as the guidance, navigation and flight control systems — the Weapon Controller;

- A two-way radio link to the local command post system, primarily supporting the exchange of mission control and status information, but also potentially to provide relayed communications between personnel located in a forward area and the battlefield commanders;
- The active payload — e.g. a warhead;
- Sensor(s) to provide local awareness and operator feedback — a video camera and associated local processing being a primary example.

Clearly in order for the vehicle to be successfully flown these all have to work together, in some cases, e.g. inertial instruments and the navigation function, the relationships (timing and accuracy) between the elements are crucial. Generally these components will be off-the-shelf items,[5] the key elements being developed are the computers and software that integrate the information from the different areas, coordinating all of the components parts to deliver the overall performance.

The vehicle is delivered in a container similar in size to those already in use for other munitions. The wings are folded when in the container, and are deployed as the vehicle leaves the container when launched. The vehicle is designed to be treated as a simple munition with no need, or indeed any capability, for periodic preventive maintenance; and it uses a number of parts that are one-shot devices that have a limited storage life.[6] The vehicle is designed to have a maintenance free service-life of ten years,[7] though this can be extended through periodic (or mid-life) updates and refurbishment; this life is, however, notably shorter than that for the overall system, i.e. approximately forty years.

The mission

Pre-launch, the vehicle is powered-up from external supplies and performs some simple functional go/no-go, tests of its systems; as it contains a number of one-shot components and is required to have simple handling extensive pre-flight testing is precluded. It is then loaded with a flight plan; in its simplest form this would be the route (a set of way-points) to its loitering area, and the flight pattern it is to adopt once it enters that area. The vehicle is then launched from the canister and it proceeds largely autonomously in accordance with this loaded flight plan.

The mission can be updated, using the radio link, at certain points in flight, thus it is possible for the vehicle to be re-routed from one loitering point to another. The concept

[5] Or closely derived from such items.

[6] Life is limited in-part to ensure that system availability and reliability targets can be met.

[7] For initial variants this will be less than ten years, but this is of little importance to the models produced in this case study.

of operation does not require that the target be pre-programmed prior to launch as the intended targets are generally not amenable to this type of approach — the vehicle is however pre-programmed with a safe target zone, limiting where it may engage targets. When a target is identified the operators upload target information to the vehicle, again using the radio link, after which the vehicle proceeds to engage the target. Video feedback is provided to facilitate human arbitration of the final phase of the engagement; this is required in order to ensure that either the target is correctly identified, or that previous engagements have not already destroyed the target.

Finally, if no target is provided the operators can cause the vehicle to self-destruct in safe target zone; the vehicle will also self-destruct if it loses communications with, or receives invalid data from, the ground system. The vehicle is not intended to be recovered should the mission be aborted, this ability would make the overall system considerably more complex, increasing the cost, and require operators with a wider set of skills.

6.1.2.2 Vehicle launch system

The vehicle launch system is a relatively simple system, its main functions are to provide the control, power and safety interlocks for pre-launch operation and the launch sequence; it also enables the local command post system to interact with the vehicle whilst on the ground either to upload the mission data, or to initiate pre-launch confidence tests.

In principle there can be several different variants of launcher but in general these will be based on existing systems albeit requiring modifications for this role. Expected applications of the system to allow for maritime launch could introduce considerable difference between launchers, all of which have to be accommodated by the payload delivery vehicle and local command post systems.

6.1.2.3 Local command post system

The local command post system is responsible for both the management of the mission for each vehicle, and providing a degree of coordination between missions — it is perhaps the single most complex component of the overall system. The key elements of this component are:

- The operators — management of the munitions and the launcher, and control of the mission;
- The communications links — control and status to and from the vehicle, and information to and from other facilities, e.g. remote and strategic planning systems,

other local command post systems (peer systems), and observation facilities (radars, unmanned air vehicles etc.), higher military command and control systems;

- The controlling computer system — coordinating and presenting the information to the operators, fusing data from different sources, and managing the use of the communications links.

The system has to be mobile and may be deployed in relatively environmentally testing conditions. As a final complication, in order to minimise cost and maximise the use of existing facilities, it is planned that it will ultimately be capable of being hosted on existing facilities; though for the initial deployment a more compact and portable facility will be developed — there is clearly a need for the temporal dynamics of this new facility to be "compatible" with those inherent in the exiting facilities.

The operators have a number of specific roles, the subset of which are of interest here are:

- Planning the vehicle route to the loitering area;
- Preparing the vehicle for launch, loading the mission data;
- Managing the vehicle mission, potentially re-routing the vehicle;
- Uploading final target information and arbitrating the final engagement, possibly through the use of video feedback.

The ultimate intent is that an operator can control a number of missions simultaneously, and there may be a number of operators in each facility — the workloads on the operators will clearly vary over the duration of a single vehicle mission. During the transit and loiter phases the vehicle is highly autonomous requiring little operator intervention, the only task being the monitoring of the vehicles' status; however in the pre-launch and (especially) the final stages of a mission the operator will have a number of tasks to perform.

Whilst the local command post systems are largely independent of one another there is the possibility of the controlling authority for a vehicle being passed from one local command post system to another. Possible reasons for this are: communications being restricted in some way in the target area; or the availability of vehicles at different launch sites — the local command post system does not have to be geographically associated with a vehicle launch site.

6.1.3 Major stakeholders and the environment

The stakeholders and the environment are the context of the system, as such they are crucially important considerations when creating the model. Whilst they are closely linked it is useful to consider then independently at this stage.

6.1.3.1 Stakeholders

There are a significant number of stakeholders for this system, more often these are organisations rather than particular individuals or small groups of individuals; broadly they can be divided into two categories: those with an interest in the development of the system; and those with an interest in its use. In some cases the stakeholder is interested in both aspects, but for distinctly different reasons. Here the interest is in the stakeholders who have an impact on, or interest in, the architecture of the system — thus those concerned with the financial aspects or similar are of lesser interest. Given the complexity of the military organisation the view of stakeholders here has been somewhat simplified.

Each of these two broad categories is considered in turn in the sections below.

Development stakeholders

It can be difficult to distinguish between a stakeholder's interest in the development of the system and their interest in its use. In this case the issues of development are considered to be those related to the financial and programme aspects: how much will it cost to develop and use the system, how long will it take, and when will it be available for use in service. Clearly there are two sides to this: the procurers and the providers.

The procurers, and their political masters, are clearly interested in budgets both short and long term (though generally less so the latter), and the time to having the capability in-service. Whilst there is some interest in the detailed capabilities of the system, and perhaps its flexibility to accommodate different roles in the future, these are not a primary consideration; hence there is little real direct interest in either the functional architecture of the system, or its detailed temporal properties.

The providers can be seen in two areas: the contractors developing the system, and the elements of the user organisation that have to deploy and manage the system once in-service — strategists, tacticians and those responsible for doctrinal issues around the system, i.e. those providing a capability. Whilst the developers are clearly not interested in the details of how the system is to be used, indeed this would be difficult to achieve for a system whose usage is expected to evolve over its life, they are undoubtedly interested in the general constraints that are inherent in its use. Equally the users are only interested

in the architecture in so far as it has an impact on the way in which they perceive the system will or can be used, or how it limits the use of the system. The two groups also have quite different temporal concerns; the developers are concerned about the low-level timing issues, e.g. rates of control laws, communications latencies etc. Contrastingly the users are interested in much higher level issues such as the time to set-up the system, the time to plan a mission, latencies in "effect" etc.

Usage stakeholders

The set of stakeholders concerned with the physical deployment and use of the system is more diverse than that for the development of the system; the key parties here are:

- The logistics organisation, responsible for the strategic deployment, storage, disposal and general management of the systems — temporal concerns relating to maintenance and re-life schedules, and storage life;
- The battle-field commanders, responsible for the strategic use of the system — time taken for the overall act of planning and executing a mission;
- Local commanders, responsible for the tactical deployment of the system, and its management in the field — mission durations, shift-patterns, firing frequency;
- Local command post operators, preparation of the individual missions, hands-on control of the mission — responsiveness and workload;
- Vehicle armourers, preparing the vehicles and the launcher, and handling misfires etc — set-up and disassembly times, failure recovery; for example how long the system needs to be left after a misfire.
- Service providers, responsible for delivering the communications and transport infrastructure upon which the system depends — communications bandwidth usage and loading patterns.

This is clearly a very diverse group, and one for which it seems unlikely that any single model could address all of their interests. In order to limit the impact of this diversity, as the model is developed the views and concerns of these stakeholders will be prioritised — the focus being on those who have a significant interest in the parts of the system being modelled in detail.

6.1.3.2 Operating environment

The operating environment of the system is the overall "military machine"; this is a rather large and complex structure that spans many "domains", which include both political and

industrial elements in addition to the "simple" armed forces. An important aspect of the environment is the legislative framework under which the system is used, this is defined in the "The Joint Services Manual of the Law of Armed Conflict" [64]. The environment will inevitably evolve over the life of the product, from development and manufacture through to entry into service and beyond; additionally the change in the environment can be rather suddenly, e.g. from peacetime, to a time of conflict.

Considering each of the development, manufacture and in-service phases different elements of the environment can be identified; briefly these are:

Development and manufacture

During these phases the environment is largely dictated by the demands of the stakeholder, but there are other elements that can be identified. Aspects such as the availability of parts and technologies have political and economic dimensions and, more technically, the ability to test the performance of the system is constrained by the availability of suitable sites and the rules governing their use, and even more mundane aspects such as the weather.[8]

In-service

Here the environment is characterised by the other aspects of the military environment with which this system interacts, this is clearly quite diverse/multi-dimensional. It ranges from elements such as the transportation and other logistical systems, when the system is passive, through to systems such as airborne control systems when the system is active. Clearly the overall political and military situation is a fundamental part of the in-service environment.

6.2 Constructing the case study — the research method

Having provided a basic description of the overall system the approach to constructing the case study may now be considered. Whilst the system is still in development it is far enough advanced such that the initial stages, so often characterised by rapid flux in concepts, requirements and solutions, are over and the systems concept is essentially stable; however this case study is based on a snapshot of the development and, as noted earlier, divergence between the "design" used for this case study and that of the system under development is inevitable. As is clear from the description the scope of the full system is quite broad with many complex interactions, many of which are non-technical;

[8]Tests and trials at the system level are expensive, thus there is a desire to gather "good" data — bad weather works against this goal.

here it is simplified to consider in essence just two of the components parts, the vehicle and local command post system, each of which is a system in it own right. Together these two components give good coverage of the issues of creating detailed timebands models.

Data used to construct the case study has been obtained from a number of sources, in decreasing order of importance these are:

- Informal discussions with key participants involved in the formulation of the technical solution, i.e. architects, designers and implementers;
- Existing design artefacts supplied in a variety of notations, e.g. UML and MASCOT (real-time network) diagrams:
- Informal discussions with other "non-design" staff who had a good understanding of the customer and end-user expectations.[9]

These were not used independently but in combination, with the design artefacts being used to help direct the discussions with the designers. The use of informal discussions rather than some more formal approach allowed the issues to be explored as they were encountered; it also helped in getting a better "explanation" and context for the design artefacts. The fact that there were only a few key participants helped make this approach an effective one.

Perhaps surprisingly the timing-related requirements of the system do not explicitly feature in the above list; this is not unusual for systems similar to this one and there two reasons for this. Firstly there are, in fact, very few timing related requirements due to the customer; and secondly, the performance of the system, to which any timing-related requirements are strongly coupled, is tradeable — requirements may be relaxed in order to allow the system to be available in-service earlier. As a result of this, it is difficult to identify a stable set of detailed time-related requirements early in development, what information there is has been obtained in the discussions with the designers. At the lower-levels there is a contrastingly greater quantity of timing related information. In the case of the vehicle there are many timing diagram describing the detailed interaction between the various elements of this component, but little of this is easily traceable to the higher level requirements. It is not unusual for the timing information for such a system to be "organised" in this way, with there being little abstraction between the information if the high level requirements and that from the lower-level design detail — this, of course, is a significant part of the problem that timebands seek to address.

[9]It was not possible to discuss the system directly with either customers or end-users.

6.3 Initial overall system-of-systems model

Given the earlier outline description of the various components of the system the process for creating a timebands model, developed in Section 5.4.2, can now be applied. The first step in the process is to construct an initial overall system-of-systems model, thus setting the context for the component models that will be developed subsequently. The role of this initial model is to capture the fundamental temporal attributes of the system and its environment, and use these to identify the natural bands of the system. This outline model may then be used to guide the development of the timebands models for each of the component systems. By its very nature this initial model can be viewed as a "first approximation" to defining the final bands for the system, and certainly there is nothing in this structure which should be considered immutable during the development of either the component, or the final system-of-systems models.

This initial model can be seen as being informal for a number of reasons:

- The granularities and precisions for each band are only "weakly" developed;
- Within the bands the formal precedence relationships are incomplete;
- The inter-band relationships are not fully developed, only addressing the essential activity-event and event-event mappings — the inter-band granularity mappings are not considered in any detail;
- The mode models are not fully developed, the issue of persistence is not considered.

6.3.1 Developing the model

The process of building this model follows that defined in the previous chapter for an initial system-of-systems model. The steps in the process are:

1. Define the overall context;
2. Identify the stakeholders and their specific interests;
3. Establish the temporal bounds of the model;
4. Identify any modal behaviour;
5. Identify tightly coupled human interactions.

The optional fifth step is included for this system as it is clear that there is a high degree of human interaction with the system.

On the whole both the overall context of the system and a range of the stakeholders have been defined and identified in the earlier section. The following sections will refine the description and develop the model by addressing steps 1. through 5. above.

Step 1 — Defining the overall context

Whilst the overall system has been outlined in the earlier sections a particular context has to be established for this case study. As noted earlier, the model will be limited to consider only the vehicle and local command post components, and will focus on the operational, in-service aspects of the system; thus other aspect such as the design, development and manufacture will be ignored. Also, in order to keep things relatively simple, the model will be limited to address only the case of a single mission per operator, and will thus discount the possibility of mission handover between local command post systems. Whilst these might appear to be rather significant limitations they do not detract from the overall value of the case study, the components remain complex, all that has been lost are some more complex interactions, a significant number of which would have been within the "environment" given how this system has been scoped.

There are several different aspects to the temporal context of the system and these align to the interests of the different stakeholders. As noted earlier the service life of the system is expected to be in the order of forty years. This actually means that the actual life of the system, from the start of development through to final disposal, is somewhat longer, perhaps extending the life to a figure nearer to fifty years; however for the purposes of this case study only the true service life will be considered.

Whilst the system has a service life of forty years it is not the case that it is simply built and then used with no account of time. A number of events can and will cause changes to the system over its life, a number of the contributors are:

- Lifed components require periodic replacement — this approach is often adopted for non-testable items such as one-shot devices (e.g. the wing deployment mechanism) in order to guarantee reliability and availability;
- Parts of the system, which are shared with other facilities, can be replaced or updated independently of this system;
- The role of the system will inevitably evolve — however this will be driven by circumstances which cannot be planned for *a priori*, this may require changes to functionality and potentially operational doctrine, and may result from so-called "Urgent Operational Requirements";

- Simple component obsolescence will require that the payload delivery vehicle be updated in some way — traditionally this has been achieved through a series of so-called mid-life updates, which are planned to occur at approximately ten to fifteen year intervals, though parts of the ground equipment are expected to be replaced more frequently, possibly over a five year cycle.

For convenience the point at which these events are managed will be combined — thus all of the lifed components will be replaced as part of one of the "planned" mid-life update events.

Considering the use of the system operationally, across the life of the system a number of deployments into the operational arena can be expected;[10] the durations of these can range from a few weeks to a number of years. Actually very short deployments are also possible for exercise and training purposes, but these are ignored here as they add very little of interest to the model, and it is unlikely that they strongly influence the dynamics of the system for normal operation. Within each deployment there can be a number of missions, each of which can take a number of hours, generally these will be within one operator "shift" period.

The final aspect of interest at this point is that of the operator's response to the different types of data being passed from the payload delivery vehicle. These are all task oriented, thus the times range from a few seconds to a small number of minutes, generally less than ten.

From the above it is clear that the temporal context of the model is quite diverse, spanning a considerable range from a few seconds, up to many years. Limiting the system to its operational environment simplifies the context somewhat, but not to the point that the system becomes in any way trivial or artificial.

Step 2 — Identifying the stakeholders

In keeping with the context of the system being limited to its operational, in-service use, and its scope being limited to the payload delivery vehicle and the local command post system, the stakeholders are a restricted subset taken from the group identified earlier. In this case, those of interest are the system operators and their local managers. Clearly the other stakeholders have an impact on the system, but given the context these may be viewed as part of the "environment" of the system.

[10] These are called "Battlefield Missions" by the end user.

Step 3 — Establishing the temporal bounds of the model

The upper bound follows directly from the context developed in **Step 1** above; the life of the system is expected to be forty years and thus this becomes the upper bound of the model. The upper bound could be extended to accommodate a longer life than originally anticipated, not an uncommon occurrence for military systems, but for the purposes of this initial model that is unlikely to be of any benefit, and as any alternative value chosen would almost certainly be arbitrary.

Identifying the lower bound is somewhat more problematic; here there is no particular value that presents itself as being the true, or obvious, lower bound for the system. In fact, in order to achieve the fidelity of modelling required for this initial phase it is unnecessary to find the true lower bound of the system, all that is required is that the bound be low enough to both set an adequate context for the subsequent modelling of the components, and capture correctly any timing aspects in the environment that act upon the overall model. As such the bound can again follow from the context described previously and is thus determined by the "issues" where there are interactions between the operator and the vehicle, especially those at the end of the mission. This leads to the lower bound being of the order of one second.

Undoubtedly there will be some faster equipment-equipment interactions that are not arbitrated by humans, and that these would drive this lower bound down further, however these are not easily identified without making assumptions regarding implementation, and are probably best left to be considered during the final refinement of the system-of-systems model, which follows from the development of the component models.

In summary: the upper bound of the model is forty years and the lower bound, the finest granularity, is one second. This will ultimately result in an upper time band for the system — the SYSTEM-LIFE band. At this point in the process it is not possible to identify the detailed characteristics of this band, these will emerge once the subsequent steps of the process have been actioned and characteristics have been defined for some of the lower bands.

Step 4 — Identifying fundamental modal behaviours

Identifying the fundamental modal behaviour of a system can, as the modes can identify and focus the temporal dynamics of the system, give a strong pointer to one, or perhaps more, time bands of the system. For the system-of-systems model the modal behaviours of interest are limited to those apparent across the overall system, rather than those associated with a specific component. It is possible, indeed quite likely, that the components

will have their own modal behaviours but these are not of interest at this stage in the process.

The highest modal behaviour of the overall system is quite simple and relates to the system being deployed, i.e. it is in use rather than being stored. There are only a limited number of fundamental modes of the system when deployed; these are:

Quiescent The system is setup and ready for use, but currently there are no missions in progress;

Active There are missions in progress.

There is also a passive "Not-deployed" mode from which the system comes into being and returns to after deployment, but that is of no interest here. The fundamental difference between the **Quiescent** and **Active** modes is that it is only in the latter there is the need for human "support". Figure 6.2 gives an overall view of this simple mode model — the internal rounded-rectangles are the modes of the model, and the labeled arcs are the transitions, with the initial transition being the arc from the solid circle to the **Quiescent** mode.

If these modes are to be useful in defining the time bands then they must both reside in the same band; however before considering the definition of a band it is necessary to understand how the modes interact; the transitions between these modes are:

Configure The setting up of the system and the establishment of the inter-system communication paths;

Start-missions The start of a new set/sequence of missions, causes the transition for **Quiescent** to **Active**;

Complete-missions The completion of the last mission in a set/sequence, causes the transition from **Active** to **Quiescent**;

The transitions between the **Quiescent** and **Active** modes do not denote actual missions but, in the case of the *Start-missions* event, simply the start of a mission when none is already in progress, i.e. the beginning of a new sequence; and, in the case of the *Complete-missions* event, the completion of the "final" mission of a sequence. Of course in the case of a single mission being performed the *Start-missions* and the *Complete-missions* events would relate to the same mission.

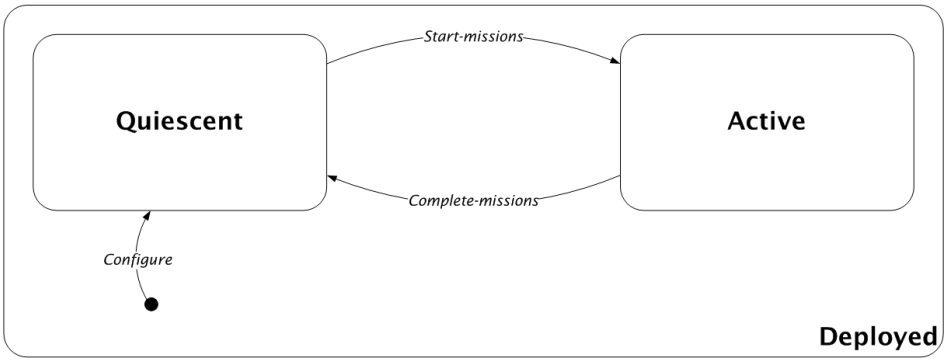


Figure 6.2: Modes for a deployment

The time band for these modes is defined by the bounds on deployment durations. The upper bound for the deployment can only relate to the intended operational use of the system, which is currently not fully developed; however the upper bound will be no more than the duration of a tour of duty, which is usually six months.[11] The lower bound will be significantly less, deployments for training exercises could last as little as a few days.

Within the **Active** mode one of the significant events relates to conducting a mission. Whilst the maximum flight time of the vehicle is intended to be of the order of ten hours, the mission also includes a preamble for planning and preparation; this results in an extreme upper bound of something in the order of eleven or twelve hours. At the other extreme the shortest mission is one that is aborted (or fails) immediately after launch, in this case the lower bound is dominated by the planning and preparation preamble, this results in a time between one and two hours. Assuming for now that conducting a mission is an event in this time band then the precision of the band can be no less than twelve hours. However if, rather than being described as a single event, the missions were to be deconstructed into a sequence of events then the precision could be refined to a lower figure. This latter approach is the one adopted here; the precision is defined to be four hours, as this both allows the mission to be decomposed into its fundamental phases, and provides a clear discrimination between the events and activities within the band. The granularity of this band is not particularly important for now but a value of two days would be appropriate for capturing the bound of the deployment — it does not unduly limit the minimum duration of a deployment. Given the context for this case study, there is little of interest that occurs whilst in the **Quiescent** mode.

As the mode extensions to the framework require that transitions are mapped to events in the modal band all of these must ultimately map to lower-level activities that have an overall duration of less than the precision of the band. Thus the transitional events

[11] Extended tours of duty can occur but these will change little in the model.

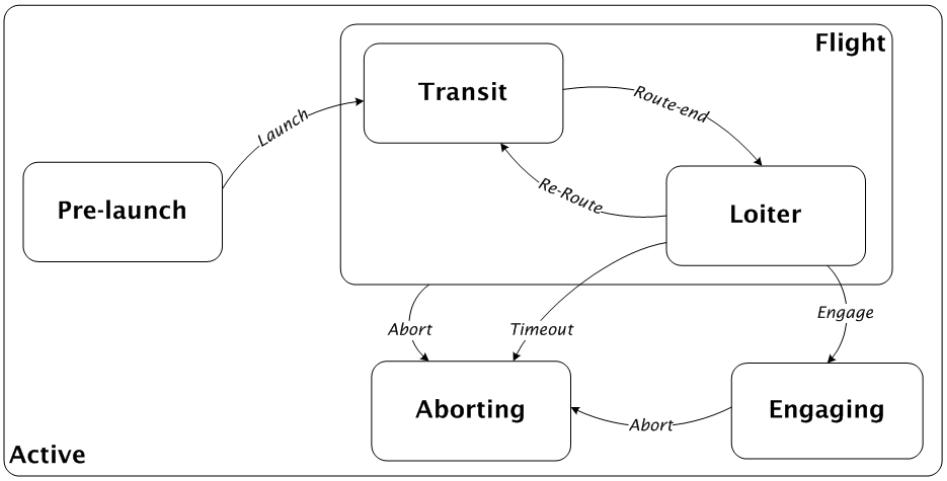


Figure 6.3: Modes for the mission under **Active**

identified above must each map to activities having a duration of less than four hours. It is interesting to note that precision is of greater importance than granularity in the definition of a modal band due to the need to be able to identify these transitional events.

Having defined one modal band it is worth considering whether there is any modal hierarchy under the either of the **Active** or **Quiescent** modes that can help in the identification and definition of another band. The **Quiescent** mode does not have any fundamental underlying structure that is of interest in this case study, however in the case of the **Active** mode some structure can be identified. The **Active** mode relates to the progress of a number of missions, therefore it is not unreasonable to define a mode model for a mission as a child of the **Active** mode. This model is somewhat more complex than that for the deployment, the overall interactions between its modes and the associated transitions is shown in Figure 6.3.

In more detail the model consists of the following modes:

Pre-launch Initial power-up checks and mission loading;

Flight A collection of modes consisting of:

- **Transit** Flying the pre-programmed route to the loitering region;
- **Loiter** Maintaining the position in the loitering region.

Engaging Moving from the loitering region to engage with the target;

Aborting Moving to the abort/self-destruct zone.

With the transitions between these modes being:

Launch The operator initiating a launch sequence;

Re-Route The development of a new route and its transmission to the vehicle;

Route-end Reaching the end of the route to the loitering zone;

Engage The operator providing a target and commanding the vehicle to engage with that target;

Timeout The lack of a target being provided to the vehicle within a specified time of entering the loitering area;

Abort The operator commanding the mission to be aborted.

To a degree this mode model has been simplified ignoring the possibility of reversion from the **Aborting** mode, or a *Disengage* transition back to the **Loiter** mode.

At first sight, this model might appear to be rather vehicle centric, and hence somewhat out of place in the overall system-of-systems context; however this is not the case. This mode model is appropriate to the system-of-systems model because the essence of it relates to the dynamics of the overall system; it may indeed appear to focus largely on the vehicle but that is only an acknowledgment of the clear, central role that the vehicle plays in the mission. The actual viewpoint, as indicated by most of the transitions is that of the interaction between the vehicle and the local command post system — thus this is a valid mode model in a band of the system-of-systems model. The upper bound of this band is twelve hours, the maximum mission time including planning and vehicle preparation. The lower bound is five minutes, a value sufficient to accommodate the description of the activities of this band without allowing too much detail from lower levels to creep in; a precision of 30 seconds is appropriate in that is provides a significant distinction between the activities and events — the modal changes have to occur within this time. It is interesting to observe that this lower band is driven to some degree by the dynamics of the vehicle — a faster vehicle with a shorter mission duration would in all probability require a finer granularity, or possibly a significantly different banded structure for the model.

Perhaps a more general observation here is that it is the nature of the transitions, which directly map to events in their modal band, that determines the appropriateness or otherwise of a band to a given mode model; if the transitions map to events that represent the interactions between component systems then the band, and hence mode model, are certainly part of the system-of-systems model. This observation can be extended somewhat to include transitions due to events originating in the environment, if these affect more

than one component system then this would also imply a system-of-systems context. Alternatively, if the events predominately relate to the interactions, or abstract views of activities within a particular component system, or between a single component system and the environment, then the band and hence mode model should probably be considered to be an element of the model of the particular component system, rather than part of the system-of-systems model.

Having defined two mode models in apparently adjacent bands it is not unreasonable to ask the question(s): could these models usefully be combined into a single model, and hence be mapped to a single band?

The simple answer to both parts of the question is no; there are two points to consider. The dynamics of the **Quiescent** to **Active** transition, in the higher-level model, are very different from those of the transition between, say, **Pre-launch** and **Transit**, in the lower-level model; the first is measured in tens of minutes, with the second in a small number of seconds. A second aspect to consider is that of the overall range of the band: as a single band the range would be from a number of years down to five minutes, a range of greater than 10^5. This is certainly much greater than that used by Newell for his bands, and thus does not appear to be necessarily appropriate — later when the system-of-systems bands are refined in the light of the developed component models the possibility of a reduction to a single modal band will be reconsidered.

In summary: there are two adjacent modal bands:

DEPLOYMENT With an upper bound of six months, a granularity of two days and a precision of four hours; and

MISSION With an upper bound of twelve hours, a granularity of five minutes and a precision of 30 seconds.

Step 5 — Identifying tightly coupled human interactions

The process defines this step as being optional, it is only required where there is a high degree of human interaction, which for this system is the case — hence the step is followed.

There are two aspects of tightly coupled human interaction in this system: the local command post operators managing the vehicle missions; and the armourers preparing the vehicles for launch. Both of these relate to sequences of relatively short duration activities, ranging from a few seconds up to around fifteen minutes.

In the first group, the activities consists of mission planning, and the detailed control of a mission — the analysis of live feedback and taking confirm/abort decisions during the course of the mission. In the second group, all of the tasks relate to the preparation of the vehicles: fuelling, loading and pre-flight testing. The range of times varies between the small number of seconds required to make the commit or abort decision, to the much longer time required to prepare the vehicles, in this latter cases the times are probably measured in minutes.

This leads to a band with a lower bound of a second, thus matching the lower bound of the system, and an upper bound of a quarter of an hour, which overlaps with the lower bound of the MISSION modal band. It should be noted here that this band only considers deliberate human actions, the reaction to events is not considered at this point, but will be addressed when the model for the local command post system is developed.

Recalling Newell's bands from earlier it is worth noting that the characteristics defined for this band do not directly map to any of his bands. This band overlaps with the lower end of the Rational band and the upper end of the Cognitive band, notably it has a range of the same order as do Newell's bands — 900 *cf.* 10^3. There are three related questions arising from this observation; these are:

- Is it acceptable for this band, which relates to tightly coupled human interactions, to be different from the model provided by Newell?
- Should this band be modified to align with one of Newell's bands, and if so which one?
- Should the range of the "band" be expanded, with the resulting band being split into two, which in turn map directly to Newell's Cognitive and Rational bands?

The answers to these, as with the distinction between the two modal bands earlier, are to be found through consideration of the context of this model. This is a system-of-systems model and as such the temporal dynamics emerge from the interactions between the systems rather than being those internal to the systems themselves, hence it is perhaps inappropriate to follow Newell's model dogmatically in the system-of-systems context. Later, when the local command post system is modelled in isolation, Newell's bands may indeed prove to be an appropriate structure within which to capture the human dynamics. In summary: there is a single tightly-coupled human interaction band, with an upper bound of fifteen minutes, and a granularity of one second — the OPERATOR-INTERACTION band. With this being the lowest band of the system (for now) no explicit precision is defined.

6.3.2 The initial model

Having followed the steps of the band selection process, as described in the sections above, four time bands have been identified. It might be possible to identify more bands, e.g. through the splitting of the human interaction band to map precisely to Newell's bands, however at this stage it is perhaps best to keep the model simple and allow it to be refined by the influences of the component models rather than introduce additional bands that constrain these other models artificially. The four bands, starting from the highest, are:

1. The SYSTEM-LIFE band;
2. The DEPLOYMENT band;
3. The MISSION band.
4. The OPERATOR-INTERACTION band.

The names given to these bands might in some cases appear rather functional (*cf.* temporal) in nature, however each relates to a specific aspect of temporal dynamics of the system — if a more directly, time-related view were to be taken then they could be considered to be: Year, Month, Minutes, and Seconds bands (not a particularly helpful viewpoint?).

The initial characteristics of the lower three bands have already been defined in the sections above; in the light of these characteristics the definition of the SYSTEM-LIFE band may now be completed: it granularity being one year and its precision being one month. The characteristics of all of the bands are summarised in Table 6.1.

Band	Upper bound/Scale	Granularity	Precision
SYSTEM-LIFE	40 years	1 year	1 month
DEPLOYMENT	2 years	2 days	4 hours
MISSION	12 hours	5 minute	30 seconds
OPERATOR-INTERACTION	15 minutes	1 second	—

Table 6.1: Band Characteristics

It should be remembered that the precisions allocated at this stage are only indicative and that the requirement that the duration of an event being less than the precision of its bands is not rigorously enforced (though in general it is followed). The precision is not defined for the lowest band — however as this is a system-of-systems model, which will ultimately be viewed in the context of those of its components, this value will be

defined when the model is refined later in this case study. A final point to note is that the inter-band granularity mappings have not been considered at this point.

The process of defining the clocks and granularities poses a question regarding the ordering of the bands: should the bands, at this stage, be totally or partially ordered? As the intent, in subsequent stages, is to refine the model then it seems appropriate that this initial model is suitable for such refinement. This in turn would seem to preclude the totally ordered model as any refinement would also have to be totally ordered — unless of course additional bands were to be introduced. The alternative of having a partially ordered model is a less deterministic solution, and as such affords more scope for options at the component models, and hence for the later refinement of this model.

However the process cannot be seen as simply being one of model refinement, as the component models are developed they will influence the final system-of-systems model, and depending on the degree of this influence the process could move from one focused on refinement to one more concerned with reformulation. In this case, as can be seen in Table 6.1, the bands are partially ordered with overlap between the OPERATOR-INTERACTION and MISSION bands, and between the DEPLOYMENT and SYSTEM-LIFE bands. It is interesting to consider what the state would have been had more bands been defined for this model: would the resultant model been more strongly ordered? This is a question that will be returned to later in this case study.

6.3.3 Populating the model — artefact identification, placement and relationships

The initial model focuses on the events in each band, activities are also defined where they can be seen to unambiguously reside within that band. When the model is refined in the light of the component models it is likely that additional events and activities will be identified; it is also possible that some others will move between bands.

A number of fundamental events and activities have been identified; these are described in the context of their bands in the sections below, commencing with the highest band — that of SYSTEM-LIFE. It should be noted that the values of the activity duration in the sections that follow are given in 'ticks' in the granularity of their band.

SYSTEM-LIFE

At this stage only a few events and activities are identified in the SYSTEM-LIFE band; these together with the band characteristics are detailed in Table 6.2. Additional activities could be defined if the scope of the model were to be widened beyond that of the execution of a deployment and associated missions, e.g. activities related to disposal.

Band name:	SYSTEM-LIFE	
Granularity:	1 year	
Precision:	1 month	
Events:	*Mid-life-update*	A major upgrade to, or refurbishment of the system, these always occur in the context of the in-service activity
	End-of-development	The start of the In-Service activity
	Deploy	A number of these could be expected over the life of the system, these can only occur after the end of development event, these relate to the deployment of the system into an operational zone and the issues about inserting and removing the system to and from the zone, and not about the use for missions
	Disposal	The disposal of part of the system, clearly there can be subdivisions of this to allow for the disposal of the independent components in the individual component models.
Activities:	`In-service` 40	A long activity relating to the existence of the system in a usable state

Table 6.2: SYSTEM-LIFE band definition

There are a number of precedence relationships that may be stated: *End-of-development* precedes the start of the `In-service` activity, also the *Mid-life-update* and *Deploy* events must occur within the bounds of the this activity; finally the *Disposal* event may occur up to the end of `In-service`. Thus:

$$\textit{End-of-development} \prec \uparrow\texttt{In-service} \tag{6.1}$$

$$\uparrow\texttt{In-service} \prec \textit{Mid-life-update} \prec \downarrow\texttt{In-service} \tag{6.2}$$

$$\uparrow\texttt{In-service} \prec \textit{Deploy} \prec \downarrow\texttt{In-service} \tag{6.3}$$

$$\textit{Disposal} \preceq \downarrow\texttt{In-service} \tag{6.4}$$

DEPLOYMENT

The DEPLOYMENT band covers both the deployment of the system in the field and its use for active operations. Clearly there are elements in this band that can be related to the disposal events in the higher band — however these are not developed further in this case study. The activities and events associated with this band together with the fundamental band characteristics are detailed in Table 6.3.

There are some basic precedence relationships between the events within this band. All occurrences of the *Conduct-mission* event must occur within the bounds of an occurrence of the `Deploy-system` activity; additionally no mission can occur until the system has

Band name:	DEPLOYMENT	
Granularity:	2 days	
Precision:	4 hours	
Events:	*Start-missions*	Starting a new sequence of missions
	Complete-missions	The completion of a sequence of missions
	Configure	The setting up of the system
	Mission-received	A demand to perform an individual mission
	Conduct-mission	Performing a complete individual mission, from preparation through to termination — a number of these would be expected in a single deployment
Activities:	`Deploy-system` 8 .. 90	The system is deployed in the field *cf.* being stored

Table 6.3: DEPLOYMENT band definition

been configured thus the following precedence relationships apply:

$$\uparrow\texttt{Deploy-system} \prec \textit{Configure} \prec \textit{Start-missions} \tag{6.5}$$

$$\textit{Start-missions} \preceq \textit{Conduct-mission} \preceq \textit{Complete-missions} \tag{6.6}$$

$$\textit{Complete-missions} \preceq \downarrow\texttt{Deploy-system} \tag{6.7}$$

The *Mission-received* event is the initiator for the *Conduct-mission* event; hence the following relationships apply:

$$\textit{Mission-received} \preceq \textit{Conduct-mission} \tag{6.8}$$

Considering the constraints defined by the mode model for this band, the *Conduct-mission* event may only occur whilst in the **Active** mode, i.e.

$$\{\textbf{Active}\} \overset{inv}{\Rrightarrow} \{\textit{Conduct-mission}\} \tag{6.9}$$

The key relationships between the activities and events in this band are shown in Figure 6.4.

MISSION

The MISSION band relates to the temporal dynamics of a single local command post facility controlling a number of vehicles. The activities and events associated with this band together with the fundamental band characteristics are detailed in Table 6.4.

All activity in this band follows a *Mission-received* event, thus:

$$\textit{Mission-received} \prec \uparrow\texttt{Configure-vehicle} \tag{6.10}$$

$$\textit{Mission-received} \prec \textit{Plan-available} \tag{6.11}$$

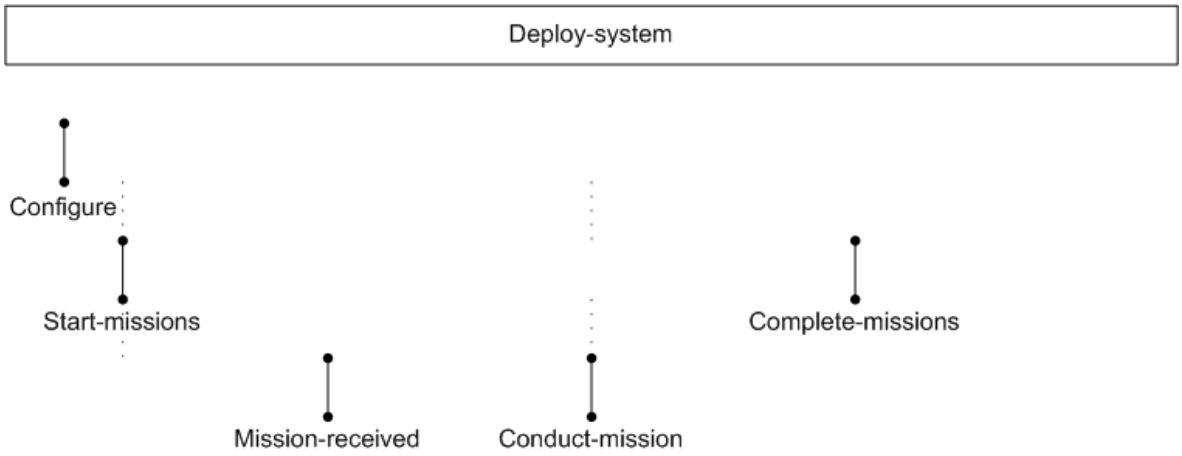


Figure 6.4: Deployment band — activities and events

Band name:	MISSION		
Granularity:	5 minutes		
Precision:	30 seconds		
Events:	*Launch*		Launching the vehicle
	Abort		Communication of an abort demand to the vehicle
	Timeout		Vehicle autonomously re-directs to abort zone having lost communications with the local command post system
	Engage		The operator commands the vehicle to engage with the target
	Re-route		New route received
	Route-end		End of route reached — commences loitering
	Prelaunch-tests		Pre-launch tests performed
	Load-mission		Initial mission loaded
	Mission-received		Demand to conduct a mission
	Plan-available		The initial mission plan or an update is ready to be loaded
	End-of-mission		Completion of the mission
Activities:	`Configure-vehicle`	9..15	Load the canister on the launcher, supply fuel and connect cables
	`Monitor-vehicle`	2..120	Continuous monitoring of the state of the vehicle or the target

Table 6.4: MISSION band definition

In order that the mission may be loaded into the vehicle both the vehicle and plan must be “ready”, thus:

$$\downarrow\texttt{Configure-vehicle} \prec \textit{Power-up} \prec \textit{Prelaunch-tests} \prec \textit{Load-mission} \tag{6.12}$$

$$\textit{Plan-available} \prec \textit{Load-mission} \tag{6.13}$$

Once the mission has been loaded the vehicle may be launched:

$$\textit{Load-mission} \preceq \textit{Launch} \tag{6.14}$$

At some point after launch the vehicle will have reached the end of its route to the loitering area; only after the vehicle has reached this point it will respond to re-route commands, and this behaviour will only persist until the vehicle is committed to a target or has its mission aborted, i.e.

$$\textit{Route-end} \preceq \textit{Re-route} \tag{6.15}$$

$$\textit{Re-route} \prec \textit{Abort} \tag{6.16}$$

$$\textit{Re-route} \prec \textit{Engage} \tag{6.17}$$

$$\textit{Re-route} \prec \textit{Timeout} \tag{6.18}$$

Throughout the mission, from the point at which the vehicle is powered to the end of the mission, the vehicle is monitored periodically, i.e.

$$\textit{Power-up} \preceq \uparrow\texttt{Monitor-vehicle} \tag{6.19}$$

$$\downarrow\texttt{Monitor-vehicle} \equiv \textit{End-of-mission} \tag{6.20}$$

The end of the mission is related to the *Abort*, *Timeout* and *Engage* events, thus:

$$\textit{Abort} \equiv \textit{End-of-mission} \tag{6.21}$$

$$\textit{Timeout} \equiv \textit{End-of-mission} \tag{6.22}$$

$$\textit{Engage} \prec \textit{End-of-mission} \tag{6.23}$$

Using the mode model for the **Active** mode in the MISSION band the following constraints can be stated regarding a number of events only occurring within the **Pre-launch** mode:

$$\begin{aligned}\{\textbf{Pre-launch}\} \overset{inv}{\Rrightarrow} \{&\uparrow\texttt{Configure-vehicle}, \downarrow\texttt{Configure-vehicle},\\ &\textit{Power-up}, \textit{Prelaunch-tests}, \textit{Plan-available},\\ &\uparrow\texttt{Monitor-vehicle}\}\end{aligned} \tag{6.24}$$

The basic relationships between the activities and events in this band are shown in Figure 6.5.

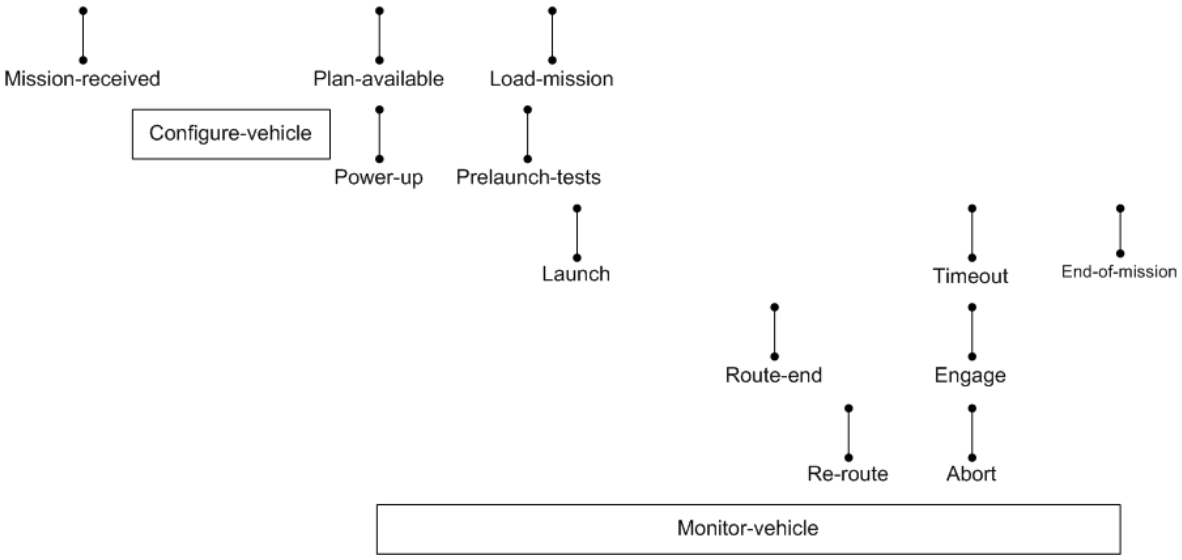


Figure 6.5: Mission band — activities and events

OPERATOR-INTERACTION

The OPERATOR-INTERACTION band is the lowest band defined for the initial system-of-systems model. The activities and events associated with this band together with the fundamental band characteristics are detailed in Table 6.5. The events and activities defined for this band in the system-of-systems context are abstract is so far that they do not presume how they will ultimately be implemented. As an example the event *Abort-mission* may ultimately consist of a number of keystrokes on a keyboard, or alternatively there might be a dedicated button for the function. Ultimately when the component model for the local command post system is developed these abstract descriptions will be refined.

The basic relationships between the activities and events are shown in Figure 6.6, and are more formally defined below. In common with the MISSION band the initiator for any sequence of activities in this band is the *Mission-received* event, planning can commence once this demand to conduct a mission has been received; hence the following can be stated:

$$\textit{Mission-received} \preceq \uparrow\texttt{Plan-route} \tag{6.25}$$

The start of the `Prelaunch-runup` activity depends on the vehicle's readiness, and its end depends on the plan being available, hence the following can be stated:

$$\textit{Vehicle-ready} \prec \uparrow\texttt{Prelaunch-runup} \tag{6.26}$$

$$\downarrow\texttt{Plan-route} \prec \downarrow\texttt{Prelaunch-runup} \tag{6.27}$$

Band name:	OPERATOR-INTERACTION		
Granularity:	1 second		
Precision:	—		
Events:	*Mission-received*		Demand to conduct a mission
	Image-received		An image from the vehicle's camera is received
	Fire		The act of initiating the launching the vehicle
	Vehicle-ready		Vehicle loaded on the launcher and ready for use
	Vehicle-status		Status update received from the vehicle
	Load-mission		Send an original plan to the vehicle
	Send-update		Send an updated plan to the vehicle
	Engage-target		Command the vehicle to leave the loitering pattern and start the engagement with the target
	Abort-mission		Command the vehicle to abort the mission
	Final-commit		Final confirmation to engage target sent to the vehicle
Activities:	`Plan-route`	12 .. 900	Planning a initial or replacement route
	`Assess-target`	5 .. 15	Assess the state of the target from images received
	`Assess-vehicle`	5 .. 10	Assess the state of the vehicle from date received
	`Prelaunch-runup`	450 .. 600	Power the vehicle, load the initial plan and initiate pre-launch tests
	`Replan`	300 .. 600	The updating of the mission plan during the course of a mission

Table 6.5: OPERATOR-INTERACTION band definition

Once the vehicle has completed the pre-launch activities it may be launched, thus:

$$\downarrow\texttt{Prelaunch-runup} \prec \textit{Fire} \tag{6.28}$$

The status of the vehicle is assessed periodically before and during flight, thus:

$$\textit{Vehicle-ready} \prec \textit{Vehicle-status} \tag{6.29}$$

$$\textit{Vehicle-status} \preceq \uparrow\texttt{Assess-vehicle} \tag{6.30}$$

The events related to the aborting of the mission or engaging the target are clearly only relevant once the vehicle is in flight, thus the following can be stated:

$$\textit{Fire} \prec \textit{Abort-mission} \tag{6.31}$$

$$\textit{Fire} \prec \textit{Engage-target} \tag{6.32}$$

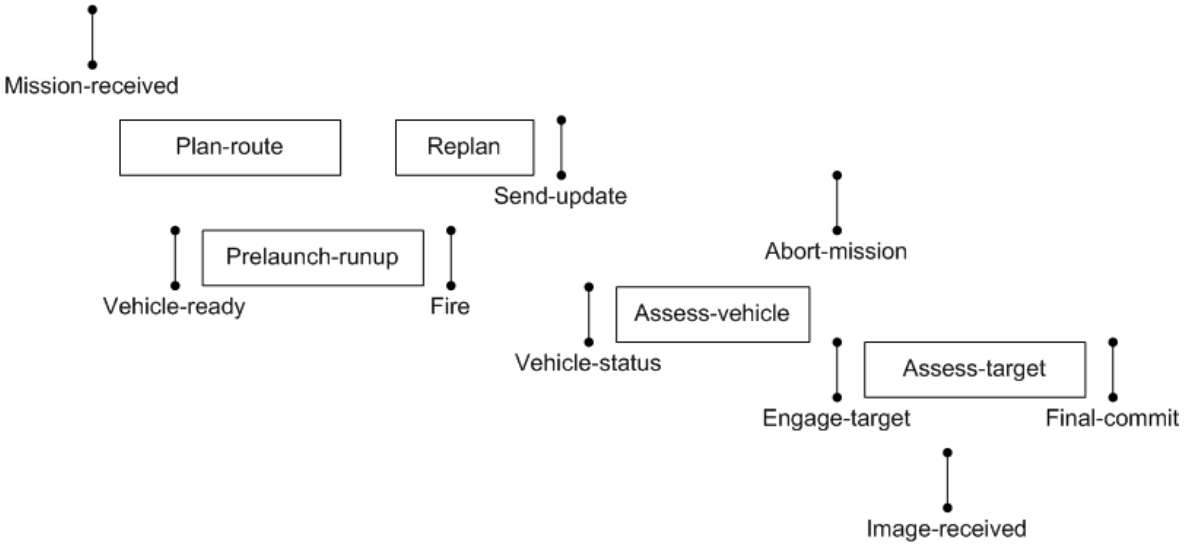


Figure 6.6: Operator-Interaction band — activities and events

Assuming that the operator is always makes the final decision to engage with the target then the final commit event is only appropriate after the target has been assessed, hence:

$$\textit{Engage-target} \prec \uparrow\texttt{Assess-target} \tag{6.33}$$

$$\textit{Image-received} \prec \downarrow\texttt{Assess-target} \tag{6.34}$$

$$\downarrow\texttt{Assess-target} \preceq \textit{Final-commit} \tag{6.35}$$

Re-planning a mission is only dependent upon the completion of the initial plan, and once re-planed an update can be sent, though the vehicle may reject or not respond to the command if it is sent whilst in an incorrect mode; hence

$$\downarrow\texttt{Plan-route} \prec \uparrow\texttt{Replan} \tag{6.36}$$

$$\downarrow\texttt{Replan} \prec \textit{Send-update} \tag{6.37}$$

This may be strengthened by the modal constraint stating that the *Send-update* event may only occur in the **Loiter** mode:

$$\{\textbf{Loiter}\} \overset{inv}{\Rrightarrow} \textit{Send-update} \tag{6.38}$$

6.3.4 Relationships between bands

Whilst this initial model is not fully developed there are a number of significant inter-band relationships that can be identified, and will be used later in the development of the component models. These relationships are presented in Figure 6.7.

The relationships can be seen to form two distinct sets: one form those that flow down through the model (requirement driven); and another form those that propagate up

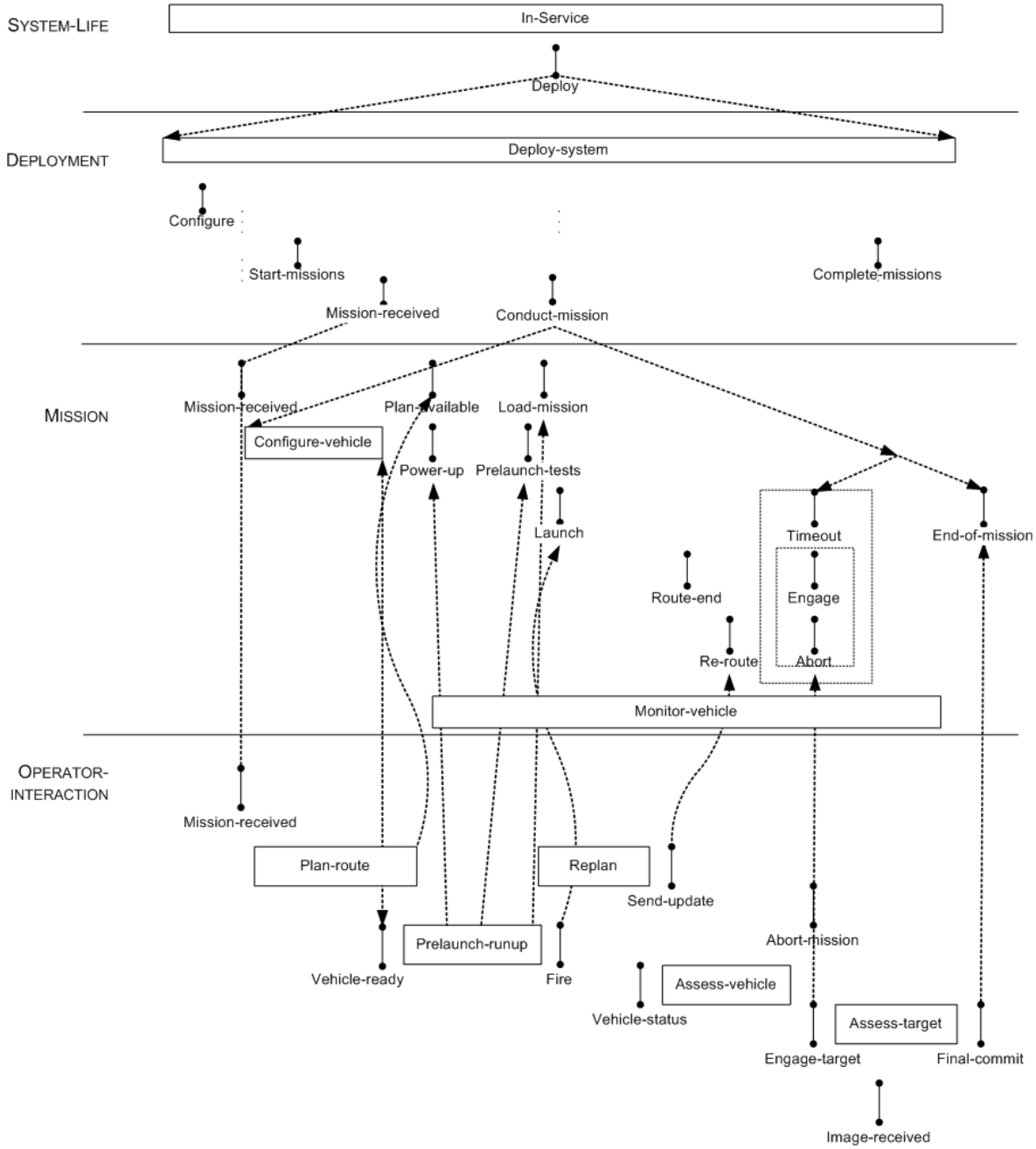


Figure 6.7: System-of-systems inter-band relationships

through the model (implementation inspired). The key requirement-based relationships can be seen as being those between: the *Deploy* event in the SYSTEM-LIFE band and the `Deploy-system` activity in the DEPLOYMENT band; and the *Conduct-mission* event in the DEPLOYMENT band, and the period between the ↑`Configure-vehicle` event up until any of the *End-of-mission*, *Timeout* or *Abort* events, all of which are in the MISSION band.

The implementation inspired mapping are more numerous; these all originate in the OPERATOR-INTERACTION band and map to events in the MISSION band. These are:

$$\downarrow\texttt{Plan-route} \longrightarrow \textit{Plan-available} \tag{6.39}$$

$$\textit{Fire} \longrightarrow \textit{Launch} \tag{6.40}$$

$$\textit{Send-update} \longrightarrow \textit{Re-route} \tag{6.41}$$

$$\textit{Engage-target} \longrightarrow \textit{Engage} \tag{6.42}$$

$$\textit{Abort-mission} \longrightarrow \textit{Abort} \tag{6.43}$$

$$\textit{Final-commit} \longrightarrow \textit{End-of-mission} \tag{6.44}$$

In addition, each of the *Power-up*, *Prelaunch-tests* and *Load-mission* events in the MISSION band originate from critical points in the `Prelaunch-runup` activity in the OPERATOR-INTERACTION band. At this stage these relationships are not well defined, but will become clearer as the models are developed.

A final point to note is the presence of the *Mission-received* event in each of the DEPLOYMENT, MISSION and OPERATOR-INTERACTION bands; whilst these events are currently indistinguishable they can also be expected to evolve as the models develop. The case of the relationship between *Vehicle-ready* in the OPERATOR-INTERACTION band and ↓`Configure-vehicle` in the MISSION band it is not entirely clear at this point whether this is one that is requirement driven, or implementation inspired — this will also become clearer as the models are developed.

6.3.5 Observations on the initial system-of-systems model

Looking at the definitions of the bands presented above it is apparent that there is some repetition across the bands, e.g. the *Mission-received* event; the question is: is this repetition necessary, and if so why? In order to answer this the drivers for the level of detail in the model need to be considered. Fundamentally there are two factors:

- The need to ensure that the model is adequate to act as the framework to support the combination of the models of the components of the system — the model will be flawed if necessary detail is omitted;

- Stakeholder perception — for the stakeholders to be bought into the model it is essential that it captures all of their issues in the context in which they expect to see them; in some cases these may be more than those simply necessary to capture the fundamental adequacy requirement.

Thus, where the stakeholders are a diverse group, as they are in the case of this system, it seems inevitable that the models will contain some "redundancy", and the inclusion of similar (or the same) artefacts in adjacent bands is not particularly surprising. This is especially the case where the ordering of the bands is weak, or the model is not fully developed. As the relationships between the bands are well defined it is unlikely that the repetition will lead to an ambiguous and hence flawed model, however any such redundancy should be limited to events, preferably those originating in the environment — it would be meaningless (and incorrect) to describe the same activity in a number of different bands.[12] The alternative approach of strict adherence to the principle of allocating each artefact to a unique band is more likely to lead to a model that has a low level of credibility with the stakeholders, and will hence be of limited use. As this is the initial version of the system-of-systems model it is possible that some of the "repeated" artefacts will be eliminated when the model is refined in the light of the component models.

A second observation is that there is a potential difficulty with the definition of the bands in this model in that they "overlap"; this can often make it difficult to determine the appropriate band for an activity or event. As an example: consider the case of the `Plan-route` activity, its duration is such that it could be accommodated in either the OPERATOR-INTERACTION or MISSION bands — in this case the activity has been placed in the lower of the two bands largely because the planning operation is operator-centric. An alternative approach to resolving this difficulty would be to combine the two bands, the disadvantage here would be the creation, as mentioned earlier, of a very wide band, and crucially one having no particular temporal focus. As with the point above regarding the repeated artefacts, it is expected that the refinement of this system-of-systems model will clarify in which band these artefacts should be located.

6.4 Component models

The next two sections move on to develop the component timebands models for the payload delivery vehicle and the local command post systems. These systems are fundamentally different: the first may be viewed as a "typical" real-time embedded system where meeting a set of fixed and hard deadlines is fundamental to system performance.

[12] It would also be completely counter to the fundamental principles of the timebands approach.

The second is a system with a significant degree of human interaction; here the deadlines are not absolute when viewed in isolation, however, in order to obtain acceptable performance the number of such deadline "misses" must be minimised — in general the quality of service provided by the system increases as more deadlines are met.[13] Taking these contrasting system types allows a broad evaluation of the process defined in the previous chapter.

In creating and developing these models the outline model developed for the overall system is used to guide the selection of the bands and identify the system level events that have to be "satisfied" by these models, though it should be noted that it is not necessary, or indeed likely, that all of the system-of-systems level events will be reflected in the components models; there are two reasons for this:

1. Some of the system-of-systems level events will emerge due to the interactions between the components;

2. As not all of the components are being modelled it is entirely possible that some events may be relate to components that are not being modelled.

In each case the models are developed using the process defined in the previous chapter, the resultant model is then described formally.

6.5 Payload delivery vehicle model

An outline description of the payload delivery vehicle was provided earlier in this chapter in Section 6.1.2.1; here the vehicle is considered in more detail, and specifically with respect to its temporal (*cf.* functional) dynamics. Figure 6.8 gives the context of the various elements of the vehicle — the central processing element, the "Weapon Controller", will be the focus for this model as this is the element of the system primarily responsible for "defining" the dynamic performance.

6.5.1 Identifying the bands

The following sections follow the process defined in the previous chapter for constructing a timebands model, starting with the identification of the temporal context. In this case the full process will be used in contrast to the development of the system-of-systems model earlier in this chapter where only the limited process was employed. Throughout

[13] Clearly there is some minimum acceptable level for this quality of service, but this is not considered in this case study.

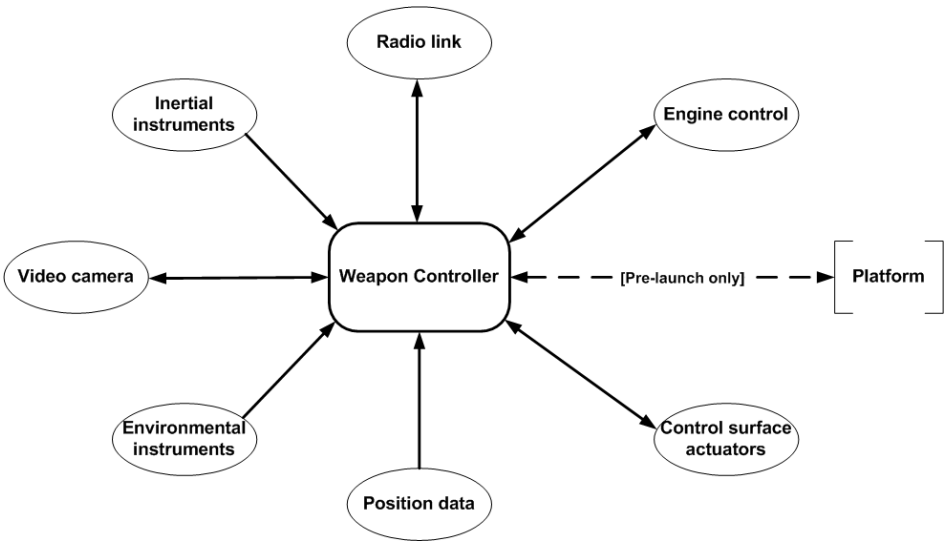


Figure 6.8: Context of the central processing element of the vehicle

these sections the description and discussion provided in the development of the system-of-systems model will assumed, and not repeated unless there is specific issue of interest relating to the vehicle context.

Step 1 — Identifying the context of the model.

As with the overall system-of-systems model there are many temporal aspects associated with the vehicle. Here again however, and in common with the overall model, the focus is on the in-service use of the vehicle; given this context the temporal issues associated with other phases of its life-cycle such as development, manufacture and disposal are not considered.

From an in-service viewpoint there are two fundamental facets that have to be considered; these are:

- The internal behaviour of the vehicle; and
- The interactions between the vehicle and the other parts of the system (see Figure 6.8).

These are not totally separable. Clearly there is coupling both where an interaction modifies an internal behaviour, and where an internal behaviour leads to some system-level interaction; however where possible these will be considered independently.

The contextual temporal boundaries have already been established in the overall system-of-systems model; these include mission duration, vehicle life etc. These will be used to guide the selection and definition of timebands later in the process.

Step 2 — Identifying the stakeholders.

The stakeholders for the vehicle model are clearly a subset of those for the overall system-of-systems model; the important ones here, together with their particular interests being:

- Planners — vehicle performance;
- Local commanders — all aspects of a mission;
- Operators — the dynamics of the interactive control of the mission.

The last of these, the operators, may actually be better considered to be part of the system rather than a stakeholder. They are in part accountable for the dynamics of the system through their interactions with the vehicle throughout the mission and are possibly, as noted by Booch [8], less interested in details of the model or architecture — i.e. not really a stakeholder. In this case however their interactions with the vehicle are moderated by the local command post, and thus they are also not actually "part" of the (vehicle) system; in this case is perhaps best to view the operators as stakeholders, albeit minor ones. This perhaps leads to a more fundamental question: are the stakeholders part of the system or part of the environment, or something else? This will be returned to in the evaluation of the models and process in the next chapter.

In addition to the above stakeholders the model is also influenced by the desire to explore the utility of the timebands framework more generally, hence the depth to which the bands are developed may be more extensive than would be strictly necessary for the purely operational context.

Step 3 — Identifying the temporal boundaries.

The upper bound for this model is easily identified as being that of the life of the vehicle with no "re-life" maintenance, however the lower bound is somewhat more difficult to determine. Taking a focused approach, and considering only the direct interactions with the other parts of the system and the "observable" behaviours of the vehicle, the lower bound could be based on its mechanical characteristics, i.e. its natural resonant frequencies, as these ultimately dictate (limit) the performance. In this case, where the resonant frequency is of the order of 50Hz, this would lead to a lower bound in the order of 20 milliseconds — thus all finer grained detail would be represented by events in the lowest band.

This choice would be perfectly acceptable if the process were being applied simply to create a useful model for the operational use across the in-service life, here however part of the goal of this case study is to explore the utility of the framework more generally

and limiting the model to this lower bound does not allow the issues of the application of the bands to implementation technologies to be explored in any detail. In order to allow these issues to be explored the lower bound is set, admittedly somewhat arbitrarily, to a value of 100 nanoseconds; this will allow the consideration of all aspects of the system implemented in software, without the need to address the even more fine-grained detail associated with the underlying electronic hardware.

In summary, the upper bound of the model is defined by the life of the vehicle, which is ten years; the lower bound of the model, the finest granularity is 100 nanoseconds.

Step 4 — Identifying any modal behaviour.

As noted in the development of the system-of-systems model, the mode model for the higher level mode **Active** defined for the MISSION band appeared to be somewhat vehicle centric. Whilst it was justifiably argued that this considered more that simply the behaviour of the vehicle it is also undoubtedly largely applicable to the vehicle itself — though perhaps the characteristics of the band will be worthy of reconsideration; in this case the band will be called MISSION^{V} in order to distinguish it from the similar one in the system-of-systems model. Before attempting to refine the characteristics of this band it is important to consider whether or not there is any modal hierarchy, or possibly some independent lower-level mode model, which is vehicle specific.

In attempting to identify lower-level modal behaviours it is important not to confuse transient actions with persistent behaviours; it would be quite easy to decompose a mode of the **Active** mode model to a state machine describing a sequence of transient actions, but this would not be a modal model. In the decomposition of the mode model for the system-of-systems model the original **Deployed** model was simple enough that a direct decomposition of the **Active** mode was possible, especially given that the **Inactive** mode was of no real interest; in this case however the situation is somewhat different as all of the modes in the **Active** model are of interest. There are two approaches that can be taken to any modal decomposition, either:

- Produce a separate model for each mode of **Active**; or
- Produce a new single model, which is entirely vehicle specific, providing the required additional behavioural detail.

It is not immediately clear which of these approaches is the more appropriate.

Recalling the development of the framework to support the hierarchical mode model given earlier in Section 4.4, the models will only be truly hierarchical if each mode of

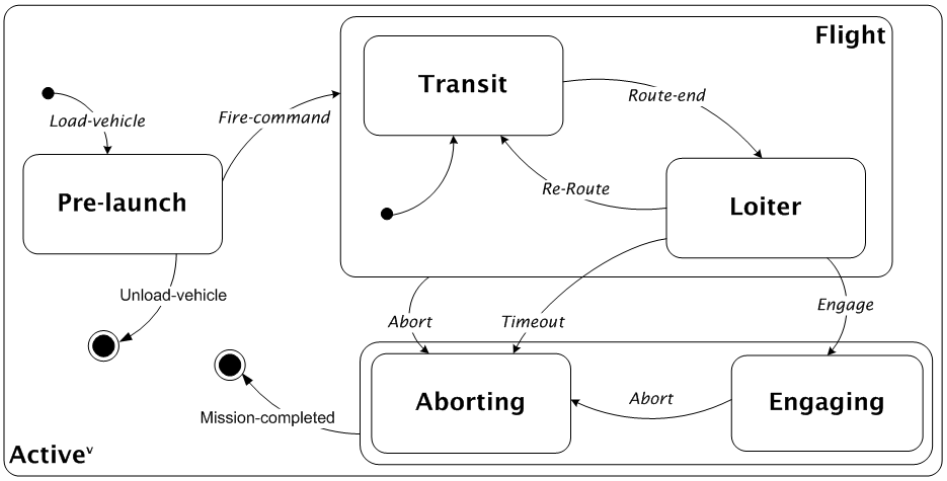


Figure 6.9: **Active^{v}** mode model — the vehicle context of **Active**

the model in a high level band maps to a complete mode model in a lower band (or there is no decomposition); this would seem to favour the first of the two approaches above. However, remembering that the original **Active** model is a system-of-systems model rather than one related purely to the vehicle it might be more appropriate simply to develop a new model in a lower band and not try to imply any formal modal hierarchy. The real disadvantage of adopting the second approach is that all mappings between the elements of the two models would have to be explicitly stated, whereas in the first approach the mode model to mode relationship described in Section 4.4.4 ensures an appropriate mapping. This is the approach that will be adopted in the development of the lower level models of the component modes of the **Active** mode; this results in the need for a second modal band, which will be called FLIGHT-PHASES — the characteristics of the band will be defined later after the corresponding mode models have been developed.

Before defining the lower levels of the mode model the **Active** model has to be made more relevant to the vehicle context. To that end Figure 6.9 provides a slightly changed view of the original **Active** mode model identifying the initialisation of the mode as being the completion of the loading of the vehicle on to the launcher, with the termination of the model being its unloading, or the completion of the mission — this is now identified as **Active^{v}**.

The decompositions of each of the modes of the **Active^{v}** model, other than **Loiter** and **Aborting**, are described in the sections below; the **Loiter** and **Aborting** modes are not decomposed further as they do not exhibit any more detailed modal behaviours.

Pre-launch

The **Pre-launch** mode model is given in Figure 6.10, most human interaction occurs in this mode. Whilst this appears to be a simple sequence and thus inappropriate for a

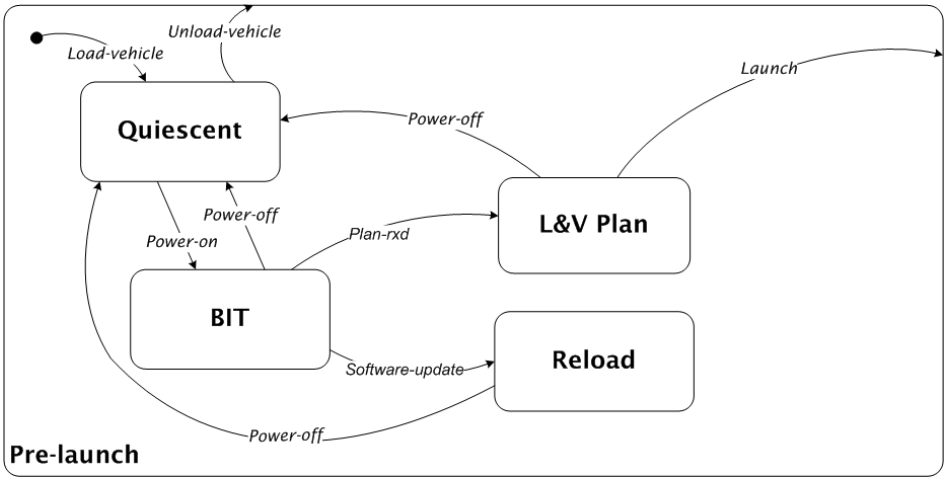


Figure 6.10: **Pre-launch** mode model

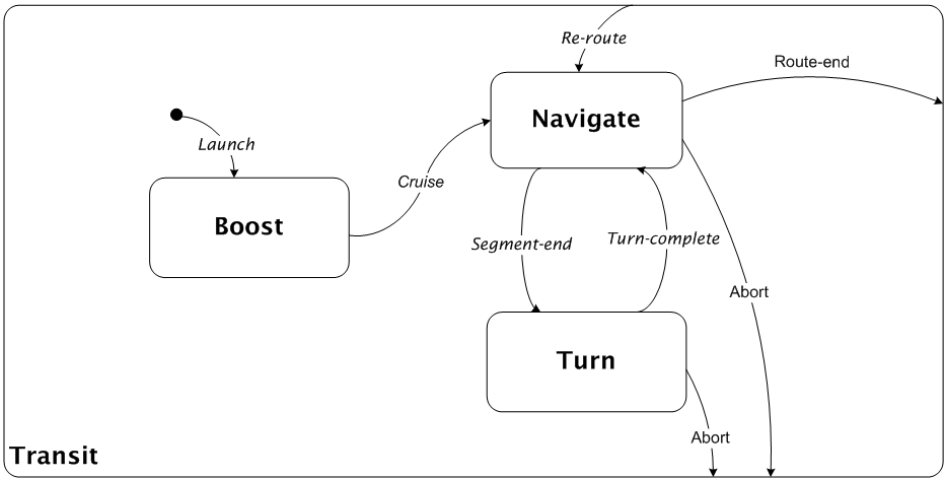


Figure 6.11: **Transit** mode model

mode model, each of the modes, **BIT**, **L&V Plan** and **Reload** all have significant duration and result in quite varied interactions with the other components of the system and thus are appropriately described as modes. In the **BIT** mode the system performs a number of self-tests and reports the results to the launch equipment, it is for the operators to assess the results and decide on the viability of the vehicle for the mission. In the **L&V Plan** mode the mission is loaded and verified, the verification status is reported to the operators who again assess the viability of the vehicle for the mission. In both of the preceding cases the operators can terminate the mission pre-launch by simply removing power.

The **Reload** mode is used to reprogram the software embedded in the vehicle; in this case this applies solely to that software resident in the Weapon Controller. This mode is terminated by the removal of power.

Transit

The **Transit** mode model is given in Figure 6.11. The **Transit** mode covers the period of flight from launch through to reaching the loitering point, and it consists of three modes: **Boost**, **Navigate** and **Turn**. Initially the vehicle enters the **Boost** mode where it is launched using a solid fuel rocket in order to achieve the airspeed required to sustain flight; this mode might be better treated as part of the "entry" event, but its persistence, of between ten and twenty seconds, is too long relative to the precision of the band for it to be viewed as an event. Once the airspeed is sufficient and the flight can be sustained the vehicle then alternates between **Navigate** and **Turn**. In the first of these the vehicle navigates using a variety of instruments, to the next way-point on its route to the loitering location; in the second it performs the fundamental manoeuvre required to set up for the next way-point. At anytime during the **Navigate** and **Turn** modes the vehicle can be commanded to abort via a radio message.

The **Transit** mode is normally exited once the end of the route, the loitering "location", is reached. Whilst it is not possible to load a new route whilst in the **Transit** mode it is possible to do so when in `Loitering`. The loading of a new route causes the vehicle to re-enter the **Transit** mode through the *Re-route* event.

Engaging

The **Engaging** mode model is given in Figure 6.12. The **Engaging** mode relates to the interactions between the operator and the vehicle in the final stages of the mission; it consists of two modes: **Find-target** and **Await-confirmation**. The first of these is a combination of flying to the general location of the target and then performing a search; the second involves the provision of target feedback to the operators and awaiting confirmation or rejection of the proposed target.

In summary: there are two adjacent modal bands, these are:

MISSION^{V} With an upper bound of twelve hours, a granularity of ten minutes and a precision of 60 seconds — these represent small changes to those defined for the MISSION band in the initial system-of-systems model, largely to allow more flexibility in defining the transitional events with respect to the lower level model.

FLIGHT-PHASES With an upper bound of one hour, a granularity of ten seconds and a precision of one second.

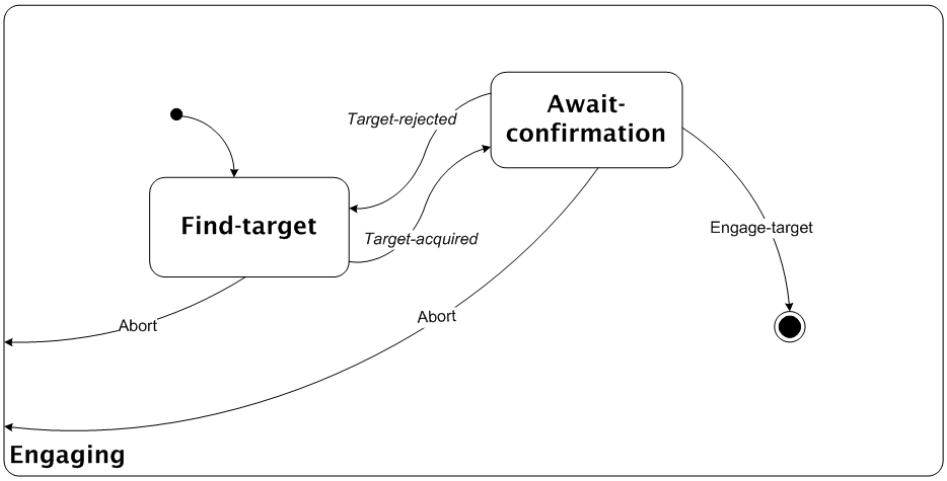


Figure 6.12: **Engaging** mode model

The detailed definition of the modes is provided formally in the context of the description of the MISSION^{V} and FLIGHT-PHASES bands in a later section.

Step 5 — Identifying any tightly coupled human interactions.
When considering the vehicle alone there are no tightly coupled human interactions. Clearly the operators have to interact with the vehicle in flight, but that is through the local command post system — thus these interactions will be considered in the context of that model. The other interactions are with the armourers who handle the vehicle and prepare it for launch, these interactions are relatively slow and can be simply accommodated within the previously defined modal bands, i.e. MISSION^{V} and FLIGHT-PHASES.

Step 6 — Identifying any modal-human interactions.
This step is redundant for this model as the previous step identified that there was little human interaction in this system, and that all such interactions could be captured in the modal bands defined in Step 4 above.

Step 7 — Identifying and partitioning the remaining organisational bands.
It is difficult to identify anything in terms of an organisational band structure to decompose the temporal space between the MISSION^{V} band an the upper bound of the model. This is largely due to the fact that the vehicle is being viewed from a relatively narrow operational in-service perspective, resulting in the issues regarding development, manufacture and upgrading being ignored. A possible structure would be to include the DEPLOYMENT band from the system-of-systems model but, when considering a single vehicle, there is little (nothing?) of interest that would appear in this band.

The model is thus restricted to having a single organisational band, that of the VEHICLE-LIFE; it has an upper bound of ten years, a granularity of one month and a precision of one day.

Step 8 — Identifying and partitioning the remaining technology bands.

The final bands to be allocated are the lower-level technology related bands are driven by the implementation, hence they are not specified in a top-down manner but emerge as the implementation decisions are made. The overall design of the vehicle software follows a real-time network approach, a simplified version of which is given in Figure 6.13; this overall architecture is further decomposed in Figures 6.14 through 6.18.

In the case of the vehicle all of the lower-level timing dynamics relate to the periods of either the control loops within the system, or those of the associated external data sources; there are several of these with the major ones being:

- Position update rate;
- Video frame rate;
- Radio link message rates (up- and down-links);
- Airframe stability maintenance loop;
- Engine control;
- Navigation;
- Guidance;
- Image processing loop.

In reality for the majority of the duration of the mission there are two fundamental loops: one controlling of the engine and maintaining the stability of the airframe, and the other related to the achievement of the flight plan, i.e. navigation and guidance. In the "end game" the dynamics of the video frame rate, image processing (acquisition and tracking) and the radio link message rates gain in significance.

Generally, in this kind of vehicle, the stability loop operates at a significantly higher rate than does the other loop; in this particular case there is a ratio of approximately 100:1 between the two loops. The rate of the stability control loop is governed by the mechanical behaviour and fundamental resonant frequencies of the vehicle, whereas the other loop is more related to the speed of the vehicle and the desired overall performance of the system. The activities of the stability control loop are generally of short and of more-or-less constant duration whereas those for the other loops are not only of longer duration but are often more variable, the variability being in-part due to the different rates associated

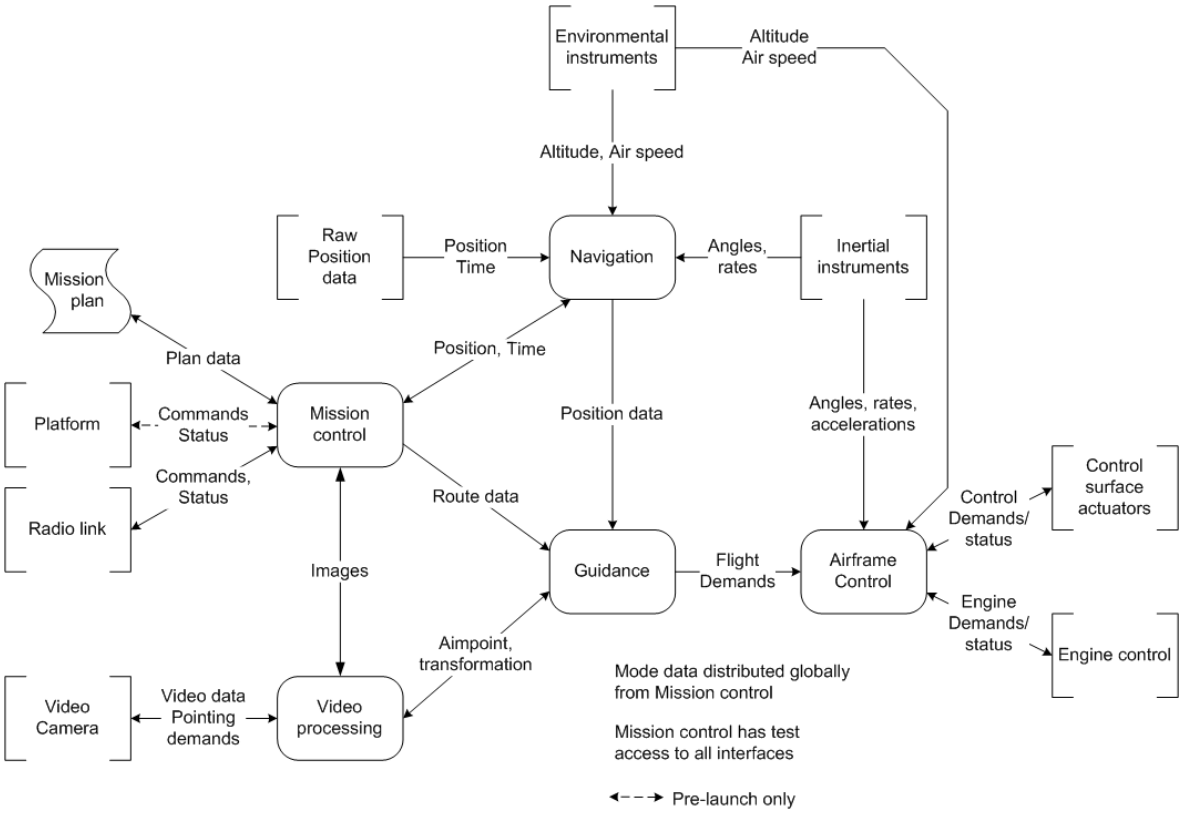


Figure 6.13: Overall real-time network design

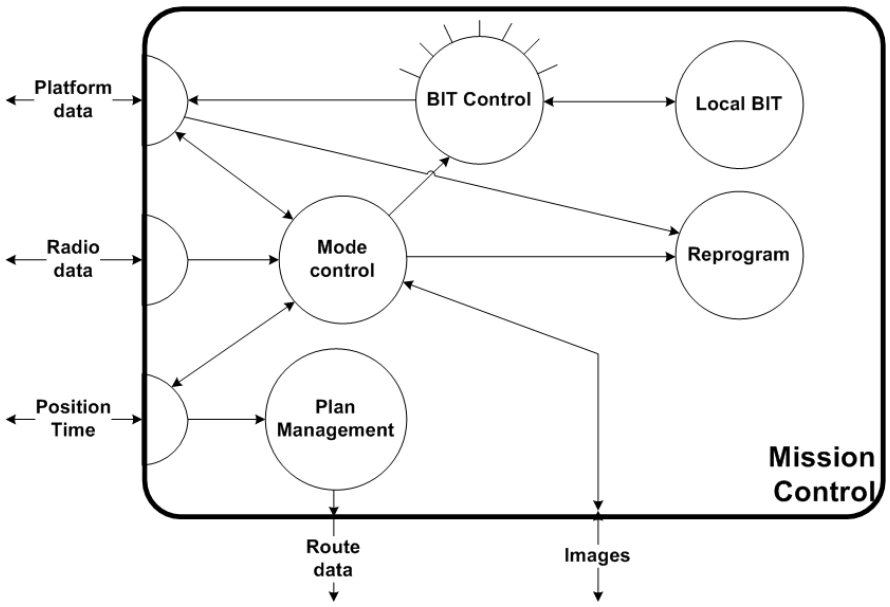


Figure 6.14: Mission control

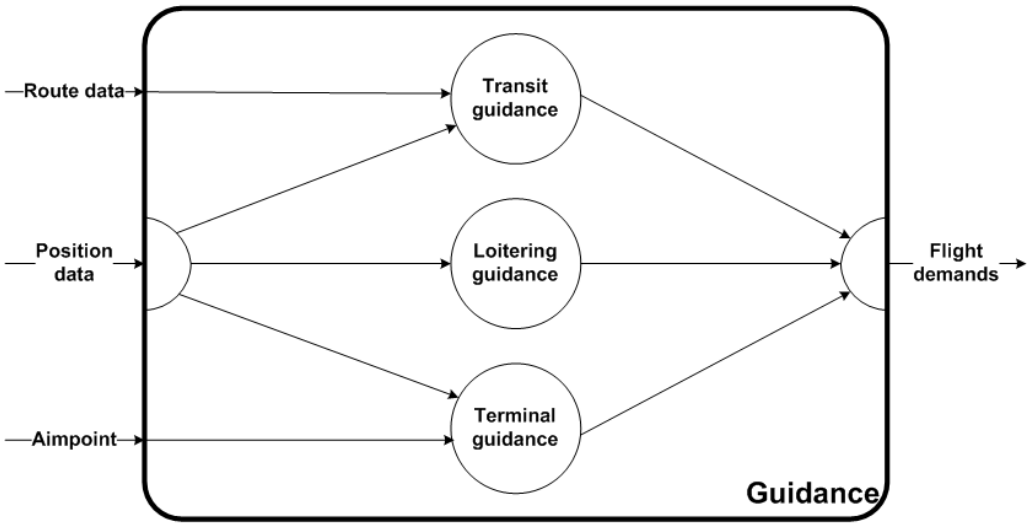


Figure 6.15: Guidance

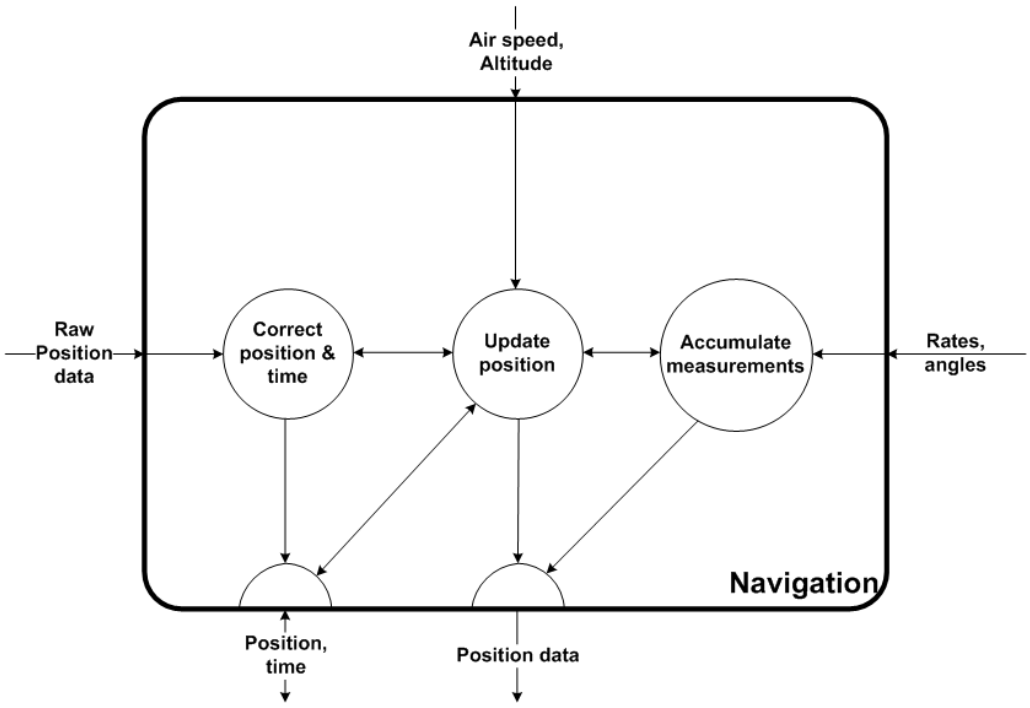


Figure 6.16: Navigation

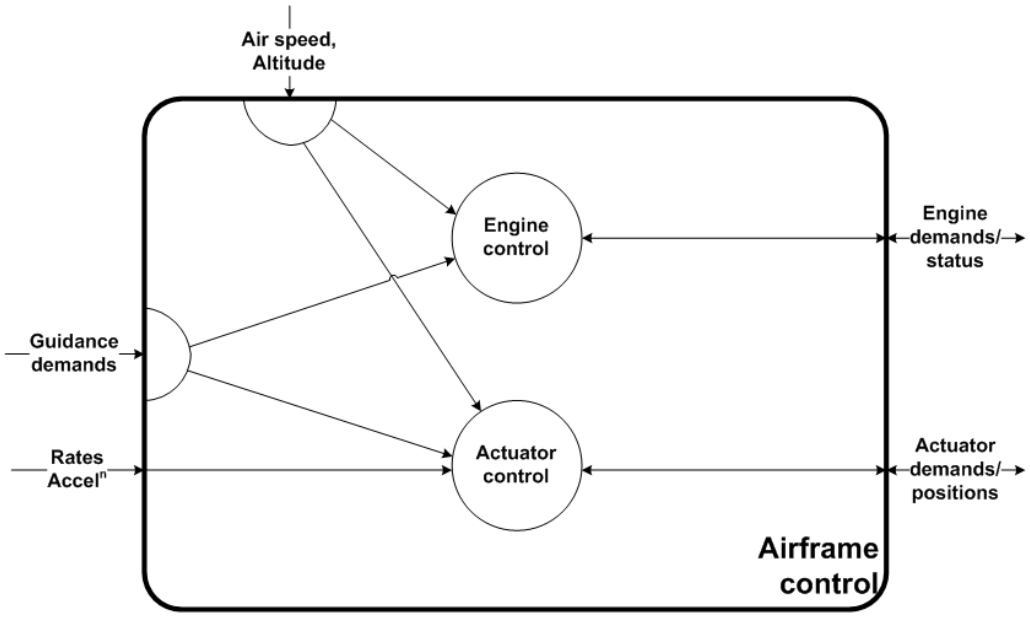


Figure 6.17: Airframe control

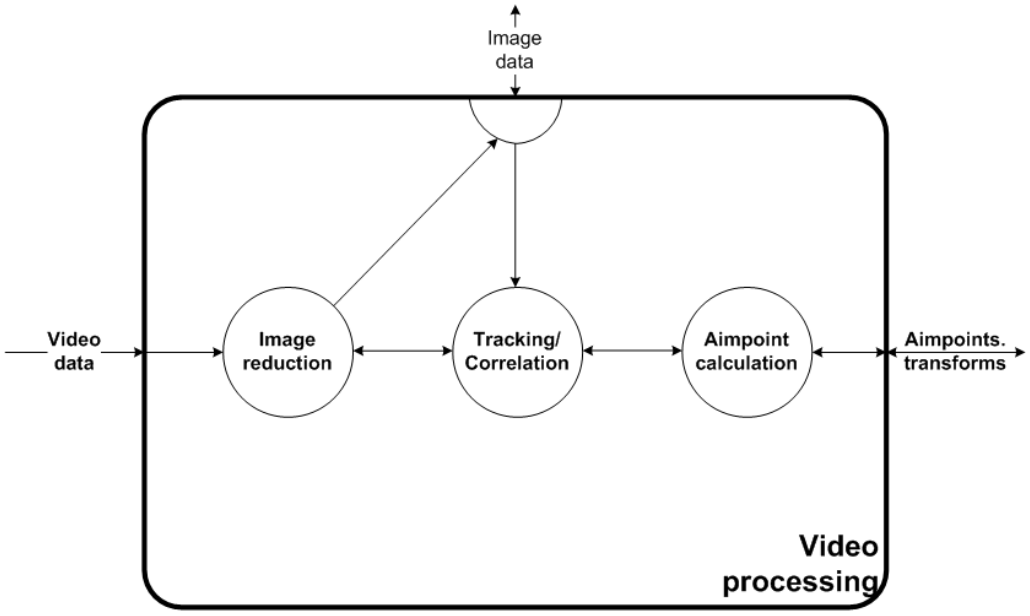


Figure 6.18: Video processing

with the various data sources, the variability of processing of the data sets, and the inherent asynchrony of the loops. Thus these two aspects have quite different characteristics and temporal dynamics and should thus be considered in separate bands.

Below these bands there are additional levels relating to more detailed characteristics of the software and, at an even lower-levels, the behaviours of the hardware. Given the modelling of the system is seen as not including the hardware platform, whose behaviours and dynamics may be taken as a given, the only possibility for an additional band would be that of the detailed characteristics of the lower-level software elements, i.e. the procedures and basic code blocks. The characteristics of these would clearly be of particularly significant interest for WCET determination and as the underpinning for any schedulability analysis. Whilst no such analyses will be considered in this case study an additional band will be defined and partially populated for this lower-level; this will allow the time-bands framework to be explored in this region, and provide better understand of the relationships between artefacts in both adjacent and non-adjacent bands.

In summary there are three technological bands, in descending order these are:

CAPABILITY For the slower rate control loops and the capabilities these provide. This has an upper bound of ten seconds, with a granularity of 200 milliseconds and a precision of 20 milliseconds;

CONTROL For the faster rate control loops that provide the inherent stability of the system. This has an upper bound of 100 milliseconds, with a granularity of two milliseconds and a precision of 200 microseconds; and

BASIC-BLOCK To capture the characteristics of the basic code blocks. This has an upper bound of ten milliseconds, a granularity of 100 nanoseconds; as with all component models the precision of the lowest band is left undefined — thus in this band individual processor instructions could be treated as events if so required.

It should be noted that the relationship between the CAPABILITY and CONTROL bands is set by the fundamental 100:1 ratio between the major control loops, whereas the characteristics of the BASIC-BLOCK band, and its relationships with the other bands are determined by the characteristics of the underlying technology and the kinds of analysis that might be undertaken.

Step 9 — Determine the granularity relationships between bands and the precision of each band.

The timebands model for the vehicle consists of six bands; in order, and starting with the highest, these are:

- VEHICLE-LIFE;
- MISSION^{V};
- FLIGHT-PHASES;
- CAPABILITY;
- CONTROL;
- BASIC-BLOCK.

As suggested above additional lower-level bands could be defined to address the detailed dynamics of the hardware platform, but these are not developed for this model. The characteristics of the bands identified in the sections above are summarised in Table 6.6, and the inter-band granularity mappings are given in Table 6.7.

Band	Upper bound/Scale	Granularity	Precision
VEHICLE-LIFE	10 years	1 month	1 day
MISSION^{V}	12 hours	10 minutes	60 seconds
FLIGHT-PHASES	1 hour	10 seconds	1 second
CAPABILITY	10 seconds	200 ms	20 ms
CONTROL	100 ms	2 ms	200 μs
BASIC-BLOCK	10 ms	100 ns	–

Table 6.6: Vehicle model band characteristics

From	To				
	VEHICLE-LIFE	MISSION^{V}	FLIGHT-PHASES	CAPABILITY	CONTROL
MISSION^{V}	4320	–	–	–	–
FLIGHT-PHASES	2.59×10^5	60	–	–	–
CAPABILITY	1.30×10^7	3000	50	–	–
CONTROL	1.30×10^9	3.0×10^5	5000	100	–
BASIC-BLOCK	2.59×10^{13}	6.0×10^9	1.0×10^8	2.0×10^6	20000

Table 6.7: Vehicle inter-band granularity mappings

There are two aspects of the band spacing that are perhaps interesting to note. Firstly the spacing between the inner four bands is largely regular with ratios of between 50:1 and

100:1; this is a somewhat narrower spacing that that suggested by Newell for his major bands and leads to the question: does this suggest that there should only be three inner bands rather than the four currently defined? The second point is that of the spacing between this inner group and the top and bottom bands; here the ratios are much greater at 4320:1 and 20000:1 respectively. These spacings are clearly much greater than those defined by Newell: does this suggest that additional bands are required in these regions?

Both of these points, and the questions they pose, will be returned to in the evaluation of the model presented in the next chapter.

6.5.2 Populating the model — artefact identification, placement and relationships

The next step in the process is to map the artefacts to the bands and define their relationships considering each band independently. The sections below provides a description of each band, including formal definitions of elements where appropriate — the formal definitions of elements of the model which span multiple bands are provided in a later section.

It should be noted that the band definitions are incomplete; this is especially true for the lower bands where only a limited selection of the possible events and activities have been defined. Those included have been selected to ensure a wide coverage of the different aspects of the framework, whilst avoiding too much unnecessary detail. In all cases the duration of the activities and the persistence of each mode is defined in terms of clock-ticks of the band in which it resides.

VEHICLE-LIFE

The VEHICLE-LIFE band is the highest band in this model and is related to the life of a single vehicle. Only a small number of events and a single activity have been defined for this band, if the scope of the model were to be widened then additional artefacts related to other aspects of the system could also be identified — e.g. those relating to development. The events and activities, together with the general characteristics of the band are defined in Table 6.8.

There are a few precedence relationships that can be stated: the *End-of-development* event must precede the start of the `Vehicle-available` activity, and the end of this activity precedes the *Disposal* event. All occurrences of the *Mission-aborted* and *Mission-*

Band name:	VEHICLE-LIFE	
Granularity:	1 month	
Precision:	1 day	
Events:	*Mid-life-update*	The completion of a mid-life update, one of the possible start and end events of the `Available` activity
	End-of-development	The end of system development — another of the possible start events of the `Available` activity
	Mission-aborted	An aborted mission — a number of these could be expected over the life of the vehicle; these can only occur in the context of the `Available` activity
	Mission-completed	A successfully completed mission — only one of these events would be expected over the life of the vehicle; these can only occur in the context of the `Available` activity
	Software-update	Updating the vehicle software
	Disposal	Another of the possible end events of the `Available` activity .
Activities:	`Vehicle-available`	80..120 A long activity relating to the existence of the vehicle in a usable state

Table 6.8: VEHICLE-LIFE band definition

completed events must occur in the context of the `Vehicle-available` activity. More formally these may be expressed as:

$$\textit{End-of-development} \prec \uparrow\texttt{Vehicle-available} \tag{6.45}$$

$$\downarrow\texttt{Vehicle-available} \prec \textit{Disposal} \tag{6.46}$$

$$\uparrow\texttt{Vehicle-available} \preceq \textit{Mission-aborted} \preceq \downarrow\texttt{Vehicle-available} \tag{6.47}$$

$$\uparrow\texttt{Vehicle-available} \preceq \textit{Mission-completed} \preceq \downarrow\texttt{Vehicle-available} \tag{6.48}$$

An additional capability is that of being able to change the vehicle software during it life thus there is also the relationship:

$$\uparrow\texttt{Vehicle-available} \prec \textit{Software-update} \prec \downarrow\texttt{Vehicle-available} \tag{6.49}$$

The relationships between the `Vehicle-available` activity and the *Mid-life-update* event are more complex. In the case of occurrences of the *Mid-life-update* event these must be preceded by an occurrence of the $\downarrow$`Vehicle-available` event; likewise all occurrences of the $\uparrow$`Vehicle-available` event must be preceded by an occurrence of the *Mid-life-*

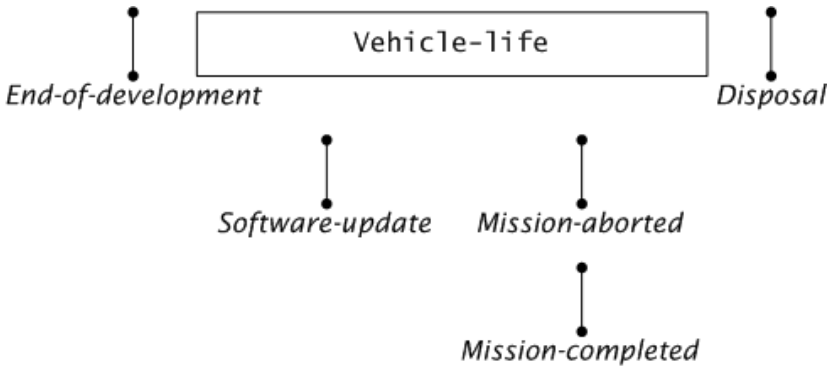


Figure 6.19: VEHICLE-LIFE band — activities and events

update event. Using superscripts to represent occurrences, more formally these may be expressed:

$$\downarrow\texttt{Vehicle-available}^{n} \preceq \textit{Mid-life-update}^{n} \tag{6.50}$$

$$\textit{Mid-life-update}^{n} \preceq \uparrow\texttt{Vehicle-available}^{n+1} \tag{6.51}$$

These basic relationships for this band are also shown in Figure 6.19.

MISSION^{V}

The MISSION^{V} band is the highest level modal band; it captures the dynamics of the overall mission, from the preparation of the vehicle through to the end of the flight. The events and activities, together with the fundamental characteristics are defined in Table 6.9 — a significant number of the events relate to the transitions of the **Active^{v}** mode model.

The two fundamental activities in this band are those representing the two types of mission that can occur: the successful case $\texttt{Mission}^{C}$ and the aborted case $\texttt{Mission}^{C}$. Both of these activities start at the point the vehicle is loaded, and they complete when the mission is completed or the vehicle is unloaded respectively. Hence the following relationships can be stated:

$$\textit{Load-vehicle} \equiv \uparrow\texttt{Mission}^{C} \tag{6.52}$$

$$\textit{Load-vehicle} \equiv \uparrow\texttt{Mission}^{A} \tag{6.53}$$

$$\textit{Mission-completed} \equiv \downarrow\texttt{Mission}^{C} \tag{6.54}$$

$$\textit{Unload-vehicle} \equiv \downarrow\texttt{Mission}^{A} \tag{6.55}$$

Considering the case of the an operational scenario where it is the intention that the vehicle be launched for a mission, there are a considerable number of precedence relationships which may be defined for this band. These relationships can be considered in

Band name:	MISSION^{V}		
Granularity:	10 minutes		
Precision:	60 seconds		
Events:	*Load-vehicle*		Load vehicle on the launcher
	Unload-vehicle		Remove vehicle from the launcher
	Fuel-vehicle		Fuel the vehicle
	Power-on		Application of power to the vehicle, all systems become operational
	Power-off		Deliberate removal of power
	Engine-start		Starting the vehicle engine
	Plan-rxd		Mission plan received and loaded
	Vehicle-ready		Vehicle ready to be launched
	Fire-command		Initiate the launching of the vehicle, igniting the booster
	Built-in-test		Execute BIT tests
	Abort		Aborting of the mission
	Re-route		New route received
	Route-end		End of programmed route reached — enter loitering mode.
	Timeout		No command received when expected or low fuel level
	Engage		Command to proceed to engage a designated target
	Mission-completed		The end of the flight and mission
Activities:	`Prepare-vehicle`	2 .. 3	Activities relating to the loading of the vehicle on the launcher up to the point of launch
	`Vehicle-flight`	1 .. 60	All aspects of the vehicle flight
	`Mission`^{A}	1 .. 12	A mission which is aborted prior to launch
	`Mission`^{C}	12 .. 72	A mission which proceeds to some form of completion
	`Update-software`	2 .. 3	Reprogramming the vehicle software
	`Status-monitoring`	1 .. 60	A “continuous” activity to monitor vehicle status in the immediate pre-launch period and through flight

Table 6.9: MISSION^{V} band definition

two groups: pre-launch, i.e. those occurring before the *Fire-command* event; and post-launch, i.e. those occurring after that event. In the first group the initial event *Load-vehicle* signifies the completion the actions relating to the fitting of the vehicle on to the launcher and hence precedes the start of the `Prepare-vehicle` activity. All of the run-up to the launch of the vehicle occurs within the context of this activity; this run-up can be seen as a sequence of events resulting in either the *Vehicle-ready* or *Unload-vehicle* events; the *Unload-vehicle* event signifies the start of the activities associated of the removal of the vehicle from the launcher.

$$\textit{Load-vehicle} \preceq \uparrow\texttt{Prepare-vehicle} \tag{6.56}$$

$$\downarrow\texttt{Prepare-vehicle} \preceq \textit{Unload-vehicle} \tag{6.57}$$

$$\downarrow\texttt{Prepare-vehicle} \equiv \textit{Vehicle-ready} \tag{6.58}$$

In more detail the sequence of events through the `Prepare-vehicle` activity for the case resulting in *Vehicle-ready* is:

$$\begin{aligned}\uparrow\texttt{Prepare-vehicle} \preceq \textit{Fuel-vehicle} \prec \textit{Power-on} \prec \textit{Built-in-test} \prec \textit{Plan-rxd} \prec \\ \textit{Engine-start} \prec \textit{Vehicle-ready} \equiv \downarrow\texttt{Prepare-vehicle}\end{aligned} \tag{6.59}$$

In the alternative case, where the vehicle is ultimately unloaded, the sequence is dependent on the reason for the unloading, amongst the possibilities here are:

- A failure in the built-in tests;
- The engine not operating correctly;
- The mission being aborted prior to launch

These are not described here as they add nothing of significance to this case study.

Once the vehicle is ready it may then be launched, and the `Vehicle-flight` activity starts. Within the context of this activity all of the other, flight-related events occur. Thus the remainder of the precedence relationships for this band are:

$$\textit{Fire-command} \prec \uparrow\texttt{Vehicle-flight} \tag{6.60}$$

$$\uparrow\texttt{Vehicle-flight} \prec \textit{Route-end} \prec \downarrow\texttt{Vehicle-flight} \tag{6.61}$$

$$\uparrow\texttt{Vehicle-flight} \prec \textit{Timeout} \prec \downarrow\texttt{Vehicle-flight} \tag{6.62}$$

$$\uparrow\texttt{Vehicle-flight} \prec \textit{Abort} \prec \downarrow\texttt{Vehicle-flight} \tag{6.63}$$

$$\textit{Route-end} \prec \textit{Engage} \prec \downarrow\texttt{Vehicle-flight} \tag{6.64}$$

$$\textit{Route-end} \prec \textit{Re-route} \prec \downarrow\texttt{Vehicle-flight} \tag{6.65}$$

$$\textit{Abort} \prec \downarrow\texttt{Vehicle-flight} \tag{6.66}$$

$$\textit{Abort} \prec \downarrow\texttt{Vehicle-flight} \tag{6.67}$$

$$\textit{Timeout} \prec \downarrow\texttt{Vehicle-flight} \tag{6.68}$$

$$\textit{Mission-completed} \equiv \downarrow\texttt{Vehicle-flight} \tag{6.69}$$

An alternative scenario is that of the vehicle being reprogrammed; in this case there is a relatively simple set of relationships. The `Update-software` activity starts with the application of power and completes with its removal, in between these two events BIT is performed and the software is updated. Hence the relationships for this scenario are:

$$\textit{Power-on} \equiv \uparrow\texttt{Update-software} \tag{6.70}$$

$$\uparrow\texttt{Update-software} \prec \textit{Built-in-test} \prec \textit{Load-software} \prec \downarrow\texttt{Update-software} \tag{6.71}$$

$$\textit{Power-off} \equiv \downarrow\texttt{Update-software} \tag{6.72}$$

The final aspect to consider is that of the `Status-monitoring` activity, this starts once the *Built-in-test* event has occurred; hence the precedence relationship is:

$$\textit{Built-in-test} \preceq \uparrow\texttt{Status-monitoring} \tag{6.73}$$

The precedence relationships for this band are also shown in Figure 6.20.

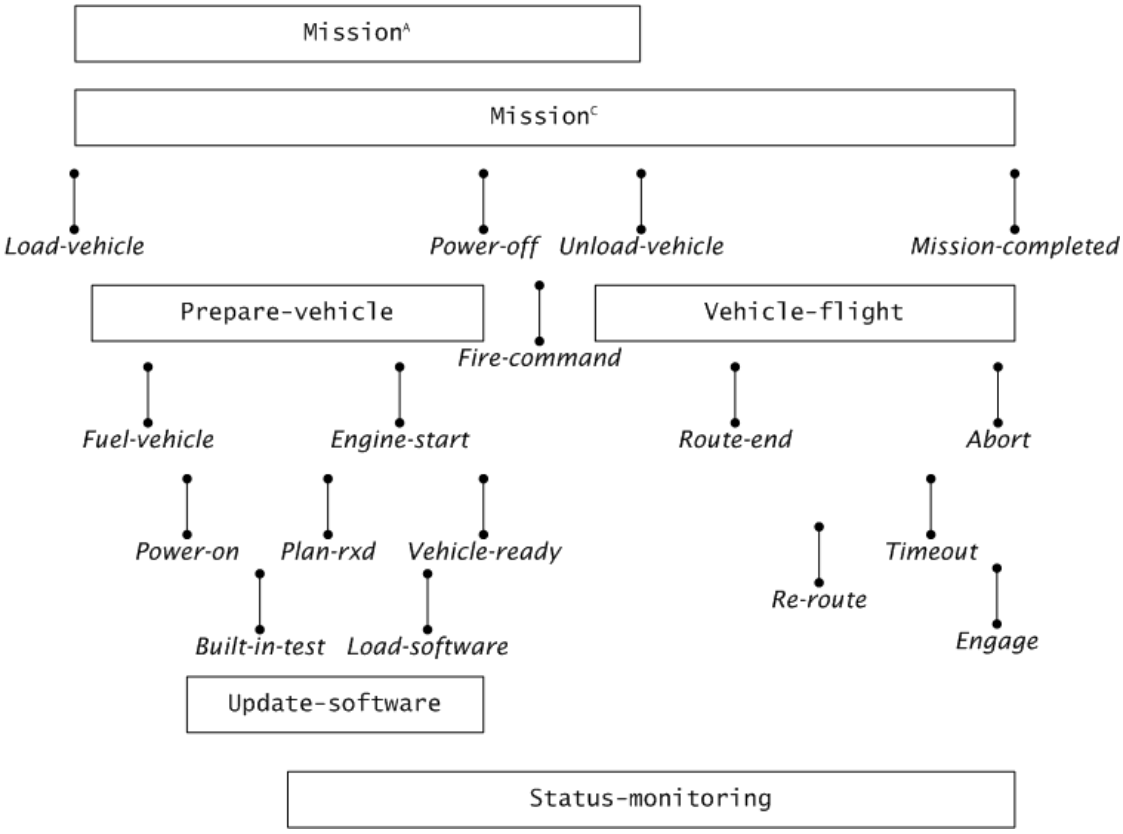


Figure 6.20: MISSION^{V} band — activities and events

In addition to the precedence relationships there are also a number of modal constraints relating to the **Active^{v}** mode model defined in this band. All of the events up to the *Fire-command* must occur in the **Pre-launch** mode:

$$\{\textbf{Pre-launch}\} \Rrightarrow \{\textit{Fuel-vehicle}, \textit{Power-on}, \textit{Built-in-test}, \textit{Plan-rxd}, \textit{Engine-start}, \textit{Vehicle-ready}\} \tag{6.74}$$

$$\{\textbf{Pre-launch}\} \overset{inv}{\Rrightarrow} \{\texttt{Prepare-vehicle}\} \tag{6.75}$$

The `Status-monitoring` activity starts in the **Pre-launch** mode and continues through to the end of the mission; hence the following pre-condition applies:

$$\{\textbf{Pre-launch}\} \overset{pre}{\Rrightarrow} \{\texttt{Status-monitoring}\} \tag{6.76}$$

The other events, which relate to the vehicle whilst in-flight, are all inter-mode transitions, and as such do not occur in any mode at this level, though the *Route-end* and *Re-route* could be said to occur in the **Flight** mode which acts as a "container" for the **Transit** and **Loiter** modes. However as the **Flight** mode only exists as a diagrammatic convenience, and is not formally modelled, there are no formal modal constraints that may be stated with regard to the **Transit** and **Loiter** modes.

The formal definition for the **Active^{v}** mode model is defined in terms of three elements: the modes; the transitions; and the overall model. Taking each of these in turn; the set of modes for this model are:

$$\mathit{ActiveModes} == \{\textbf{Pre-launch}, \textbf{Transit}, \textbf{Loiter}, \textbf{Aborting}, \textbf{Engaging}\}$$

With the values of persistence in each case being:

$$\mathit{persistence}(\textbf{Pre-launch}) == 1 \,..\, 12$$
$$\mathit{persistence}(\textbf{Transit}) == 1 \,..\, 54$$
$$\mathit{persistence}(\textbf{Loiter}) == 1 \,..\, 54$$
$$\mathit{persistence}(\textbf{Aborting}) == 1 \,..\, 2$$
$$\mathit{persistence}(\textbf{Engaging}) == 1 \,..\, 2$$

The transitions are:

$$
\begin{aligned}
&\mathit{ActiveTransitions} == \{ \\
&\quad (\mathit{Fire\text{-}command}, \textbf{Pre-launch}) \mapsto \textbf{Transit}, \\
&\quad (\mathit{Route\text{-}end}, \textbf{Transit}) \mapsto \textbf{Loiter}, \\
&\quad (\mathit{Re\text{-}route}, \textbf{Loiter}) \mapsto \textbf{Transit}, \\
&\quad (\mathit{Abort}, \textbf{Transit}) \mapsto \textbf{Aborting}, \\
&\quad (\mathit{Abort}, \textbf{Loiter}) \mapsto \textbf{Aborting}, \\
&\quad (\mathit{Abort}, \textbf{Engaging}) \mapsto \textbf{Aborting}, \\
&\quad (\mathit{Timeout}, \textbf{Loiter}) \mapsto \textbf{Aborting}, \\
&\quad (\mathit{Engage}, \textbf{Loiter}) \mapsto \textbf{Engaging}\}
\end{aligned}
$$

Finally the binding for the overall mode model for $\textbf{Active}^{\mathrm{V}}$ may be defined:

$$
\begin{aligned}
&\mathit{AciveModel} == (\!| \\
&\quad \mathit{modes} \rightsquigarrow \mathit{ActiveModes}, \\
&\quad \mathit{band} \rightsquigarrow \textsc{Mission}^{\mathrm{V}}, \\
&\quad \mathit{transitions} \rightsquigarrow \mathit{ActiveTransitons}, \\
&\quad \mathit{current} \rightsquigarrow \emptyset, \\
&\quad \mathit{init} \rightsquigarrow \{\mathit{Load\text{-}vehicle} \mapsto \textbf{Pre-launch}\} \\
&\quad \mathit{term} \rightsquigarrow \{\mathit{Unload\text{-}vehicle}, \mathit{Mission\text{-}completed}\} |\!)
\end{aligned}
$$

FLIGHT-PHASES

The FLIGHT-PHASES band is the second band strongly related to the modes of the system. It is focused largely on the key events in flight and the detailed interactions with the local command post system, which occur towards the end of the mission; it also accommodates the long duration activities performed pre-launch. As with the $\textsc{Mission}^{\mathrm{V}}$ band a significant number of the events here relate to the inter-mode transitions for the lower-level mode models. The events and activities of this band, together with its fundamental characteristics are defined in Table 6.10.

In common with the $\textsc{Mission}^{\mathrm{V}}$ band the activities and events may be divided into two groups: those occurring pre-launch; and those occurring post-launch. The relationships within the former group are associated with two quite separate conditions as, in addition to the operational mission, there is an "off-line" ability to change the software in the vehicle. In this latter case the vehicle needs to be powered, and to have completed built-in tests before any reprogramming can take place. Once the software has been updated

Band name:	FLIGHT-PHASES		
Granularity:	10 secs		
Precision:	1 sec		
Events:	*Power-on*		Application of ground power to the vehicle
	Power-off		Removal of ground power
	Launch		Command to start launch sequence
	Cruise		End of boost, enter navigate mode
	Abort		Aborting of the mission
	Timeout		No command received when expected or low fuel level
	Segment-end		End of flight segment, start turn
	Turn-complete		End of turn, enter next segment
	Re-route		New route received
	Route-end		End of programmed route reached
	Local-BIT		BIT completed for Weapon Controller
	Target-acquired		Target acquired and reported
	Target-rejected		Target rejected by operator via the local command post
	Mission-completed		End of the flight and mission
	Engage-target		Target accepted for engagement
Activities:	`Perform-vehicle-BIT`	2 .. 3	Management of the pre-launch test of the vehicle
	`Load-plan`	1 .. 2	Loading/reloading the missions plan
	`Reprogram-software`	6 .. 10	Updating the vehicle software

Table 6.10: FLIGHT-PHASES band definition

power is removed to complete the process; thus the precedence relationships here are:

$$\textit{Power-on} \prec \uparrow\texttt{Perform-vehicle-BIT} \tag{6.77}$$

$$\downarrow\texttt{Perform-vehicle-BIT} \preceq \uparrow\texttt{Reprogram-software} \tag{6.78}$$

$$\downarrow\texttt{Reprogram-software} \equiv \textit{Power-off} \tag{6.79}$$

Clearly both the `Perform-vehicle-BIT` and `Reprogram-software` activities and the power related events can only occur in the **Pre-launch** mode, hence the following modal invariant can be stated:

$$\{\textbf{Pre-launch}\} \overset{inv}{\Rrightarrow} \{\texttt{Perform-vehicle-BIT}, \texttt{Reprogram-software}\} \tag{6.80}$$

$$\{\textbf{Pre-launch}\} \Rrightarrow \{\textit{Power-on}, \textit{Power-off}\} \tag{6.81}$$

Additionally there is a modal invariant regarding the `Reprogram-software` activity as it can only occur in the **Reload** mode:

$$\{\textbf{Reload}\} \overset{inv}{\Rrightarrow} \{\texttt{Reprogram-software}\} \tag{6.82}$$

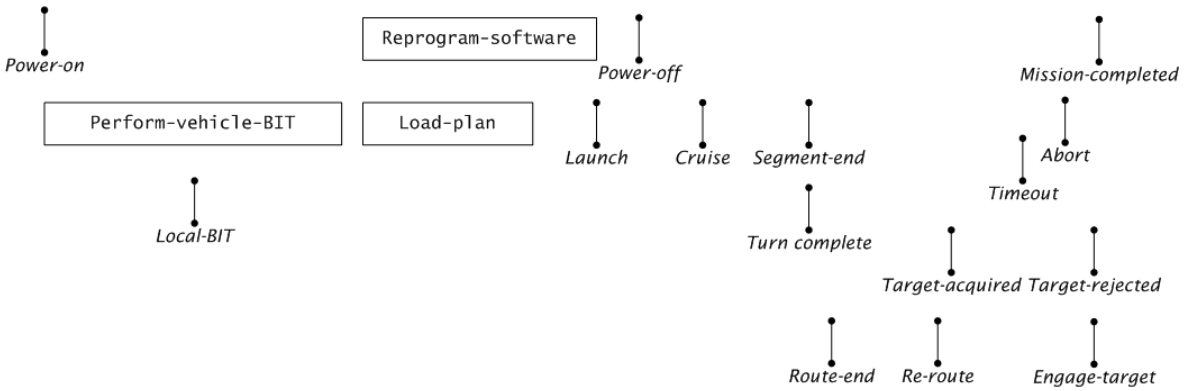


Figure 6.21: FLIGHT-PHASES band — activities and events

Moving on to consider the normal operational case up to the point of launch, the main differences are that the `Load-plan` activity follows `Perform-vehicle-BIT`, and power remains applied; hence the precedence relationships are:

$$\textit{Power-on} \prec \uparrow\texttt{Perform-vehicle-BIT} \tag{6.83}$$

$$\downarrow\texttt{Perform-vehicle-BIT} \prec \uparrow\texttt{Load-plan} \tag{6.84}$$

The following additional modal invariant applies to the `Load-plan` activity:

$$\{\textbf{Pre-launch}\} \overset{inv}{\Rrightarrow} \{\texttt{Load-plan}\} \tag{6.85}$$

Moving beyond the **Pre-launch** mode, once the plan has been loaded then the vehicle may be launched:

$$\downarrow\texttt{Load-plan} \prec \textit{Launch} \tag{6.86}$$

The other post-launch events relate to either inter-mode transitions, or the completion of the mission, some of the precedence relationships between these events are:

$$\textit{Launch} \prec \textit{Cruise} \prec \textit{Segment-end} \prec \textit{Turn-complete} \tag{6.87}$$

$$\textit{Segment-end} \prec \textit{Route-end} \tag{6.88}$$

$$\textit{Route-end} \prec \textit{Engage} \prec \textit{Target-acquired} \prec \textit{Engage-target} \tag{6.89}$$

$$\textit{Route-end} \prec \textit{Re-route} \tag{6.90}$$

$$\textit{Route-end} \prec \textit{Abort} \tag{6.91}$$

The relationships between the events and activities in this band are shown in Figure 6.21.

An additional modal constraint that can be stated for the *Target-acquired*, *Target-rejected* and *Engage-target* events is that these can only occur in the **Engaging** mode, i.e.:

$$\{\textbf{Engaging}\} \Rrightarrow \{\textit{Target-acquired}, \textit{Target-rejected}, \textit{Engage-target}\} \tag{6.92}$$

As for the **Active**$^{\mathbf{V}}$ mode model defined of the MISSION$^{\mathrm{V}}$ band the formal definitions for the lower-level mode models for this band are defined in terms of three elements:

- The modes;
- The transitions; and
- The overall model

Rather than treat each of the models separately there are combined; taking each of the elements in turn, the sets of modes for this model are:

$$PrelaunchModes == \{\textbf{Quiescent}, \textbf{BIT}, \textbf{L\&V Plan}, \textbf{Reload}\}$$
$$TransitModes == \{\textbf{Boost}, \textbf{Navigate}, \textbf{Turn}\}$$
$$LoiteringModes == \{\textbf{Loiter}\}$$
$$AbortingModes == \{\textbf{Aborting}\}$$
$$EngagingModes == \{\textbf{Find-target}, \textbf{Await-confirmation}\}$$

With the values of persistence in each case being:

$$persistence(\textbf{Quiescent}) == 6 \,..\, 360$$
$$persistence(\textbf{BIT}) == 2 \,..\, 3$$
$$persistence(\textbf{L\&V Plan}) == 2 \,..\, 6$$
$$persistence(\textbf{Reload}) == 30 \,..\, 120$$
$$persistence(\textbf{Boost}) == 1 \,..\, 2$$
$$persistence(\textbf{Navigate}) == 6 \,..\, 360$$
$$persistence(\textbf{Turn}) == 6 \,..\, 12$$
$$persistence(\textbf{Loiter}) == 6 \,..\, 3240$$
$$persistence(\textbf{Aborting}) == 60 \,..\, 360$$
$$persistence(\textbf{Find-target}) == 6 \,..\, 180$$
$$persistence(\textbf{Await-confirmation}) == 1 \,..\, 3$$

The transitions for the decomposed modes are:

$$\begin{aligned} PrelaunchTransitions == \{ & \\ & (\textit{Power-on}, \textbf{Quiescent}) \mapsto \textbf{BIT}, \\ & (\textit{Power-off}, \textbf{BIT}) \mapsto \textbf{Quiescent}, \\ & (\textit{Power-off}, \textbf{Reload}) \mapsto \textbf{Quiescent}, \\ & (\textit{Power-off}, \textbf{L\&V Plan}) \mapsto \textbf{Quiescent}, \\ & (\textit{Software-update}, \textbf{BIT}) \mapsto \textbf{Reload}, \\ & (\textit{Plan-rxd}, \textbf{BIT}) \mapsto \textbf{L\&V Plan}\} \end{aligned}$$

$$
\begin{array}{l}
\mathit{TransitTransitions} == \{ \\
\quad (\mathit{Cruise}, \textbf{Boost}) \mapsto \textbf{Navigate}, \\
\quad (\mathit{Segment\text{-}end}, \textbf{Navigate}) \mapsto \textbf{Turn}, \\
\quad (\mathit{Turn\text{-}complete}, \textbf{Turn}) \mapsto \textbf{Navigate}\}
\end{array}
$$

$$
\begin{array}{l}
\mathit{EngagingTransitions} == \{ \\
\quad (\mathit{Target\text{-}acquired}, \textbf{Find-target}) \mapsto \textbf{Await-confirmation}, \\
\quad (\mathit{Target\text{-}rejected}, \textbf{Await-confirmation}) \mapsto \textbf{Find-target}\}
\end{array}
$$

Finally the overall set of mode models for the FLIGHT-PHASES band is:

$$
\begin{array}{l}
\mathit{PrelaunchModel} == (\!| \\
\quad \mathit{modes} \rightsquigarrow \mathit{PrelaunchModes}, \\
\quad \mathit{band} \rightsquigarrow \text{FLIGHT-PHASES}, \\
\quad \mathit{transitions} \rightsquigarrow \mathit{PrelaunchTransitions}, \\
\quad \mathit{current} \rightsquigarrow \varnothing, \\
\quad \mathit{init} \rightsquigarrow \{\mathit{Load\text{-}vehicle} \mapsto \textbf{Quiescent}\}, \\
\quad \mathit{term} \rightsquigarrow \{\mathit{Launch}, \mathit{Unload\text{-}vehicle}\} |\!)
\end{array}
$$

$$
\begin{array}{l}
\mathit{TransitModel} == (\!| \\
\quad \mathit{modes} \rightsquigarrow \mathit{TransitModes}, \\
\quad \mathit{band} \rightsquigarrow \text{FLIGHT-PHASES}, \\
\quad \mathit{transitions} \rightsquigarrow \mathit{TransitTransitions}, \\
\quad \mathit{current} \rightsquigarrow \varnothing, \\
\quad \mathit{init} \rightsquigarrow \{\mathit{Launch} \mapsto \textbf{Boost}, \mathit{Re\text{-}route} \mapsto \textbf{Navigate}\} \\
\quad \mathit{term} \rightsquigarrow \{\mathit{Route\text{-}end}, \mathit{Abort}\} |\!)
\end{array}
$$

$$
\begin{array}{l}
\mathit{LoiterModel} == (\!| \\
\quad \mathit{modes} \rightsquigarrow \mathit{LoiteringModes}, \\
\quad \mathit{band} \rightsquigarrow \text{FLIGHT-PHASES}, \\
\quad \mathit{transitions} \rightsquigarrow \varnothing, \\
\quad \mathit{current} \rightsquigarrow \varnothing, \\
\quad \mathit{init} \rightsquigarrow \{\mathit{Route\text{-}end} \mapsto \textbf{Loiter}\}, \\
\quad \mathit{term} \rightsquigarrow \{\mathit{Abort}, \mathit{Timeout}, \mathit{Engage}\} |\!)
\end{array}
$$

$$
\begin{aligned}
&\mathit{AbortingModel} == (\!| \\
&\quad \mathit{modes} \rightsquigarrow \mathit{AbortingModes}, \\
&\quad \mathit{band} \rightsquigarrow \text{FLIGHT-PHASES}, \\
&\quad \mathit{transitions} \rightsquigarrow \varnothing, \\
&\quad \mathit{current} \rightsquigarrow \varnothing, \\
&\quad \mathit{init} \rightsquigarrow \{\mathit{Abort} \mapsto \textbf{Aborting}, \mathit{Timeout} \mapsto \textbf{Aborting}\}, \\
&\quad \mathit{term} \rightsquigarrow \{\mathit{Mission\text{-}completed}\} |\!)
\end{aligned}
$$

$$
\begin{aligned}
&\mathit{EngagingModel} == (\!| \\
&\quad \mathit{modes} \rightsquigarrow \mathit{EngagingModes}, \\
&\quad \mathit{band} \rightsquigarrow \text{FLIGHT-PHASES}, \\
&\quad \mathit{transitions} \rightsquigarrow \mathit{EngagingTransitions}, \\
&\quad \mathit{current} \rightsquigarrow \varnothing, \\
&\quad \mathit{init} \rightsquigarrow \{\mathit{Engage} \mapsto \textbf{Find-target}\} \\
&\quad \mathit{term} \rightsquigarrow \{\mathit{Engage\text{-}target}, \mathit{Abort}\} |\!)
\end{aligned}
$$

CAPABILITY

The CAPABILITY band is the first of the so-called technological bands, the ones that are heavily implementation dependent and have no direct human interaction. The activities and events in this band relate to durations and periods of the processes performed at the slower control loop rates. The events and activities, together with the fundamental characteristics of the band are defined in Table 6.11.

The precedence relationships between the artefacts in this band are associated with two aspects of the system; the initial start-up of the Weapon Controller; and the processing of video data. In the first case the relationships are:

$$\mathit{Power\text{-}ready} \equiv \uparrow \texttt{Init} \tag{6.93}$$

$$\downarrow \texttt{Init} \preceq \uparrow \texttt{Perform-local-BIT} \tag{6.94}$$

In the second case, that of processing the video data the `Process-image` activity can only commence once both the *Video-image* and *Axes-available* events have occurred; the guidance update event is dependent on the update of the position of the vehicle and the completion of the processing of the image. More formally these relationships are:

$$\mathit{Video\text{-}image} \prec \uparrow \texttt{Process-image} \tag{6.95}$$

$$\mathit{Axes\text{-}available} \prec \uparrow \texttt{Process-image} \tag{6.96}$$

$$\mathit{Position\text{-}updated} \preceq \mathit{Guidance\text{-}update} \tag{6.97}$$

$$\downarrow \texttt{Process-image} \preceq \mathit{Guidance\text{-}update} \tag{6.98}$$

Band name:	CAPABILITY	
Granularity:	200 ms	
Precision:	20 ms	
Events:	*Power-ready*	Power supplies stable — reset removed
	Axes-available	Axes transforms available
	Video-image	New video image available
	Guidance-update	Updated guidance input data produced
	Message-transmitted	Message sent to local command post
	Message-received	Message received from the local command post
	Position-updated	Position updated
Activities:	`Perform-local-BIT`	3 . . 5 Perform the local BIT tests
	`Init`	3 Initialisation of the system after application of power
	`Process-image`	1 . . 3 Complete processing of a single video image
	`Update-route`	1 . . 2 Update route data

Table 6.11: CAPABILITIES band definition

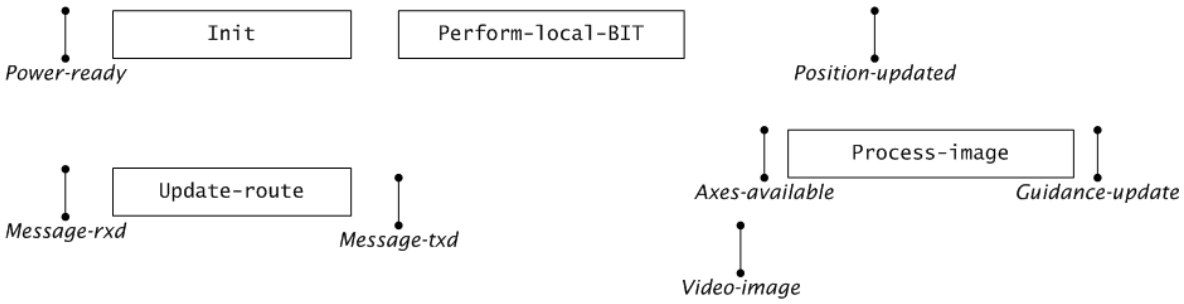


Figure 6.22: CAPABILITY band — activities and events

These relationships are also shown in Figure 6.22, no other events in this band have precedence relationships.

Whilst there are no modes defined for this band there are some modal constraints relating to the modes in the higher bands. The *Power-ready* event and the `Init` activity may only occur in the **Pre-launch** modes; the `Perform-local-BIT` activity is more tightly constrained to only occur within the **BIT** mode; hence:

$$\{\textbf{Pre-launch}\} \overset{inv}{\Rrightarrow} \{\texttt{Init}\} \tag{6.99}$$

$$\{\textbf{Pre-launch}\} \Rrightarrow \{\textit{Power-ready}\} \tag{6.100}$$

$$\{\textbf{BIT}\} \overset{inv}{\Rrightarrow} \{\texttt{Perform-local-BIT}\} \tag{6.101}$$

Equally the other events defined for this band may never occur in the **Pre-launch** mode, hence:

$$\{\textbf{Pre-launch}\} \not\Rrightarrow \{\textit{Position-updated}, \textit{Video-image}, \textit{Message-received}, \textit{Message-transmitted}, \textit{Guidance-update}, \textit{Axes-available}, \uparrow\texttt{Process-image}, \downarrow\texttt{Process-image}, \uparrow\texttt{Update-route}.\downarrow\texttt{Update-route}\} \tag{6.102}$$

The negated invariant above captures the fact that its associated set of events can only occur whilst the vehicle is in flight, however it is possible to strengthen this with the following constraints over the *Video-image* event and the `Process-image` activity:

$$\{\textbf{Engaging}\} \Rrightarrow \{\textit{Video-image}\} \tag{6.103}$$

$$\{\textbf{Engaging}\} \overset{inv}{\Rrightarrow} \{\texttt{Process-image}\} \tag{6.104}$$

CONTROL

The CONTROL band is the middle of the three technology related, implementation dependent bands; in this case the dynamics relate to iteration rates required for the faster, stabilisation related control loops. The events and activities for this band, together with its fundamental characteristics are defined in Table 6.12.

The fundamental precedence relationships in this band are those associated with the two control loops for the airframe and the engine; in both cases the updates are dependent on the availability of data from instruments within the vehicle, the relationships are:

$$\textit{Engine-status} \prec \textit{Engine-update} \tag{6.105}$$

$$\textit{Air-data} \prec \textit{Engine-update} \tag{6.106}$$

$$\textit{Air-data} \prec \textit{Stabilise} \tag{6.107}$$

$$\textit{Inertial-measurement} \prec \textit{Stabilise} \tag{6.108}$$

The following activities relating to some of the slower rate processes appear in this band: `Predict-position`, part of the overall navigation function; and `Image-extraction`, `Correlation`,`Tracking` and `Aimpoint-extraction`, the CPU intensive parts of the image processing function. These are all related to both the occurrence of the *Capability-clock* event, and the availability of the required information, i.e. image data and axes transforms, the latter of which is in-turn dependent on the inertial measurements. The precedence relationships for the start of the `Predict-position` activity are:

$$\textit{Capability-clock} \preceq \uparrow\texttt{Predict-position} \tag{6.109}$$

$$\textit{Inertial-measurement} \prec \textit{Axes-available} \preceq \uparrow\texttt{Predict-position} \tag{6.110}$$

Band name:	CONTROL		
Granularity:	2 ms		
Precision:	200 μs		
Events:	*Axes-available*		Update alignment of vehicle axes sets (transforms available)
	Stabilise		Airframe stabilisation update
	Engine-update		Engine management systems update
	Inertial-measurement		INS update available
	Engine-status		Engine status update available
	Air-data		Air data update available
	Image-received		Image data received from the local command post
	Video-image		Video image available
	Capability-clock		Clock for the CAPABILITY band
Activities:	`Predict-position`	8 .. 12	Update the predicted position
	`Image-extraction`	10 .. 60	Extraction of raw data from a video image
	`Correlation`	10 .. 50	Correlation of raw image data with the provided image
	`Tracking`	10 .. 50	Tracking of elements of interest from image to image
	`Aimpoint-extraction`	30 .. 50	Production of the aimpoint for guidance from the image data

Table 6.12: CONTROL band definition

For the image processing related activities the precedence relationships are:

$$\textit{Capability-clock} \preceq \uparrow\texttt{Image-extraction} \tag{6.111}$$

$$\textit{Video-image} \preceq \uparrow\texttt{Image-extraction} \tag{6.112}$$

$$\textit{Axes-available} \preceq \uparrow\texttt{Image-extraction} \tag{6.113}$$

$$\downarrow\texttt{Image-extraction} \preceq \uparrow\texttt{Correlation} \tag{6.114}$$

$$\textit{Image-received} \preceq \uparrow\texttt{Correlation} \tag{6.115}$$

$$\downarrow\texttt{Image-extraction} \preceq \uparrow\texttt{Tracking} \tag{6.116}$$

The start of the `Aimpoint-extraction` activity is dependent on either the `Correlation` or the `Tracking` activities completing:

$$\downarrow\texttt{Correlation} \preceq \uparrow\texttt{Aimpoint-extraction} \tag{6.117}$$

$$\downarrow\texttt{Tracking} \preceq \uparrow\texttt{Aimpoint-extraction} \tag{6.118}$$

The above relationships are also shown in Figure 6.23.

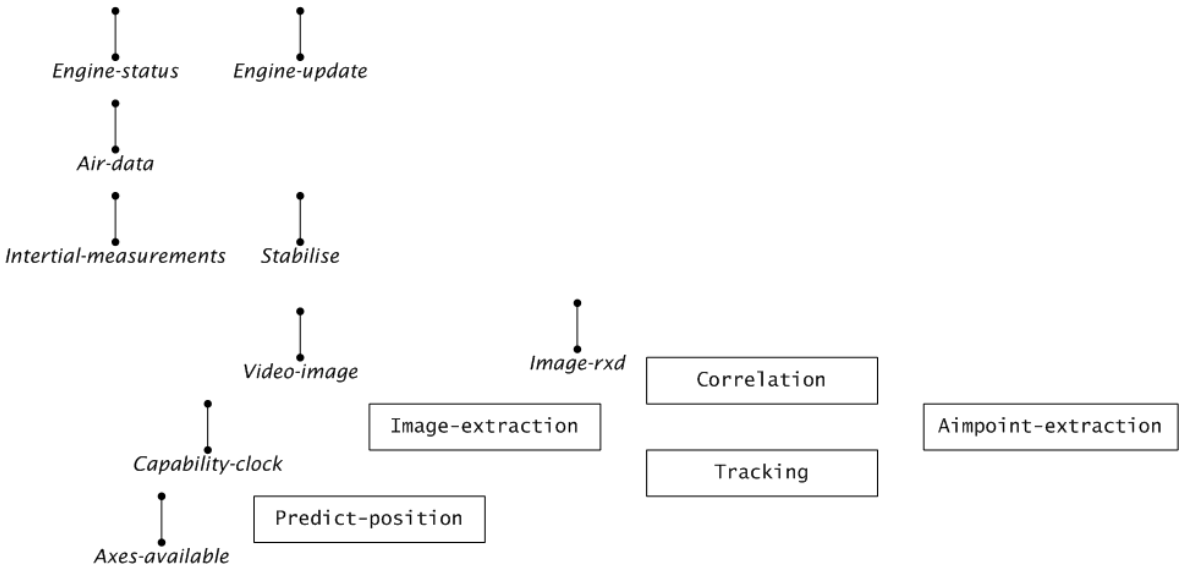


Figure 6.23: CONTROL band — activities and events

As was the case with the CAPABILITY band there are no modes defined within this band, but again there are some constraints relating to the modes defined in the higher level bands. The external subsystems of the vehicle are largely unaware of the mode of the system other than possibly when they are commanded to perform a particular function, for example built-in test, as such the event due to these systems can be considered to be of no interest with respect to mode. As this band is primarily associated with the stability of the vehicle in flight is not surprising that the fundamental constraint is that the activities do not occur in the pre-launch mode, hence:

$$\{\textbf{Pre-launch}\} \not\Rightarrow \{\texttt{Predict-position}, \texttt{Image-extraction}, \texttt{Correlation}, \texttt{Tracking}, \texttt{Aimpoint-extraction}\} \tag{6.119}$$

$$\{\textbf{Pre-launch}\} \not\Rightarrow \{\textit{Axes-available}, \textit{Stabilise}\} \tag{6.120}$$

As with the CAPABILITY band the image processing related activities are also constrained to occur only in the **Engaging** mode; hence the modal constraints are:

$$\{\textbf{Engaging}\} \overset{inv}{\Rrightarrow} \{\texttt{Image-extraction}, \texttt{Correlation}, \texttt{Tracking}, \texttt{Aimpoint-extraction}\} \tag{6.121}$$

$$\{\textbf{Engaging}\} \Rrightarrow \{\textit{Image-received}, \textit{Video-image}\} \tag{6.122}$$

A final modal constraint in this band relates to the `Predict-position` activity, it does not occur in the **Engaging** mode as the guidance at this point is governed by the output of the image processing and makes no reference the absolute position of the vehicle; hence the following constraint applies:

$$\{\textbf{Engaging}\} \not\Rightarrow \{\texttt{Predict-position}\} \tag{6.123}$$

BASIC-BLOCK

The BASIC-BLOCK band is the third and lowest of the technology related, implementation dependent bands, it is also the lowest band defined for this model. The dynamics of this band are such that it captures the detailed timing characteristics of low-level code fragments, information that would typically be needed for schedulability and similar analyses. The events and activities of this band, together with its fundamental characteristics are defined in Table 6.13.

The precedence relationships for this band are quite diverse, some are unique to this band, other are a refinement of those already present in the higher bands. In the latter category there are the stabilisation loops for the airframe and the engine previously described in the CONTROL band, the relationships here are:

$$\textit{Control-clock} \prec \textit{Inertial-measurement} \prec \texttt{Stabilise-airframe} \tag{6.124}$$

$$\textit{Air-data} \prec \texttt{Stabilise-airframe} \tag{6.125}$$

$$\textit{Air-data} \prec \texttt{Manage-engine} \tag{6.126}$$

$$\textit{Control-clock} \prec \textit{Engine-status} \prec \texttt{Manage-engine} \tag{6.127}$$

In the unique category are the relationships between the vehicle and the local command post system (via the radio link), the relationships here are:

$$\textit{Message-received} \prec \uparrow\texttt{Decode-message} \tag{6.128}$$

$$\textit{Capability-clock} \prec \uparrow\texttt{Status-collection} \tag{6.129}$$

$$\textit{Capability-clock} \prec \uparrow\texttt{Mode-processing} \tag{6.130}$$

$$\downarrow\texttt{Status-collection} \prec \uparrow\texttt{Encode-message} \tag{6.131}$$

$$\downarrow\texttt{Mode-processing} \prec \uparrow\texttt{Encode-message} \tag{6.132}$$

$$\textit{Reduced-video-available} \prec \uparrow\texttt{Encode-message} \tag{6.133}$$

The is the simple relationship between the availability of position data and the update of the position in navigation:

$$\textit{Position-update} \prec \uparrow\texttt{Update-position} \tag{6.134}$$

Band name:	BASIC-BLOCK		
Granularity:	100 ns		
Precision:	–		
Events:	*Inertial-measurement*	INS update available	
	Engine-status	Engine status update available	
	Air-data	Air data update available	
	Position-update	Position data received	
	Message-received	Radio message received	
	Reduced-video-available	Video feedback to operators ready for transmission	
	Target-data	Target data derived from image available	
	New-waypoint	Next waypoint in flight plan provided for guidance	
	Capability-clock	Clock for the CAPABILITY band	
	Control-clock	Clock for the CONTROL band	
Activities:	`Stabilise-airframe`	900 .. 1100	Control algorithms to stabilise the airframe in the roll, pitch and yaw planes whilst integrating any new guidance demands
	`Decode-message`	100 .. 300	Decode a radio link message
	`Encode-message`	100 .. 300	Encode a radio link message
	`Manage-engine`	300 .. 500	Control the engines speed to meet airspeed demands
	`Accumulate-meas`	200 .. 250	Accumulates measurements from the IMU
	`Update-axes`	200 .. 250	Update the system axes sets
	`Update-position`	200 .. 250	Update the position of the vehicle based on new position data
	`Turning-guidance`	1000 .. 10000	Calculate the guidance demands for a turn
	`Terminal-guidance`	500 .. 900	Calculate the guidance demands to intercept the target
	`Intercept-guidance`	5000 .. 10000	Calculate the guidance demands to achieve the location of the next waypoint
	`Status-collection`	1000 .. 1000	Collection of vehicle status information for reporting to operators
	`Mode-processing`	400 .. 600	Maintenance of vehicle mode information

Table 6.13: BASIC-BLOCK band definition

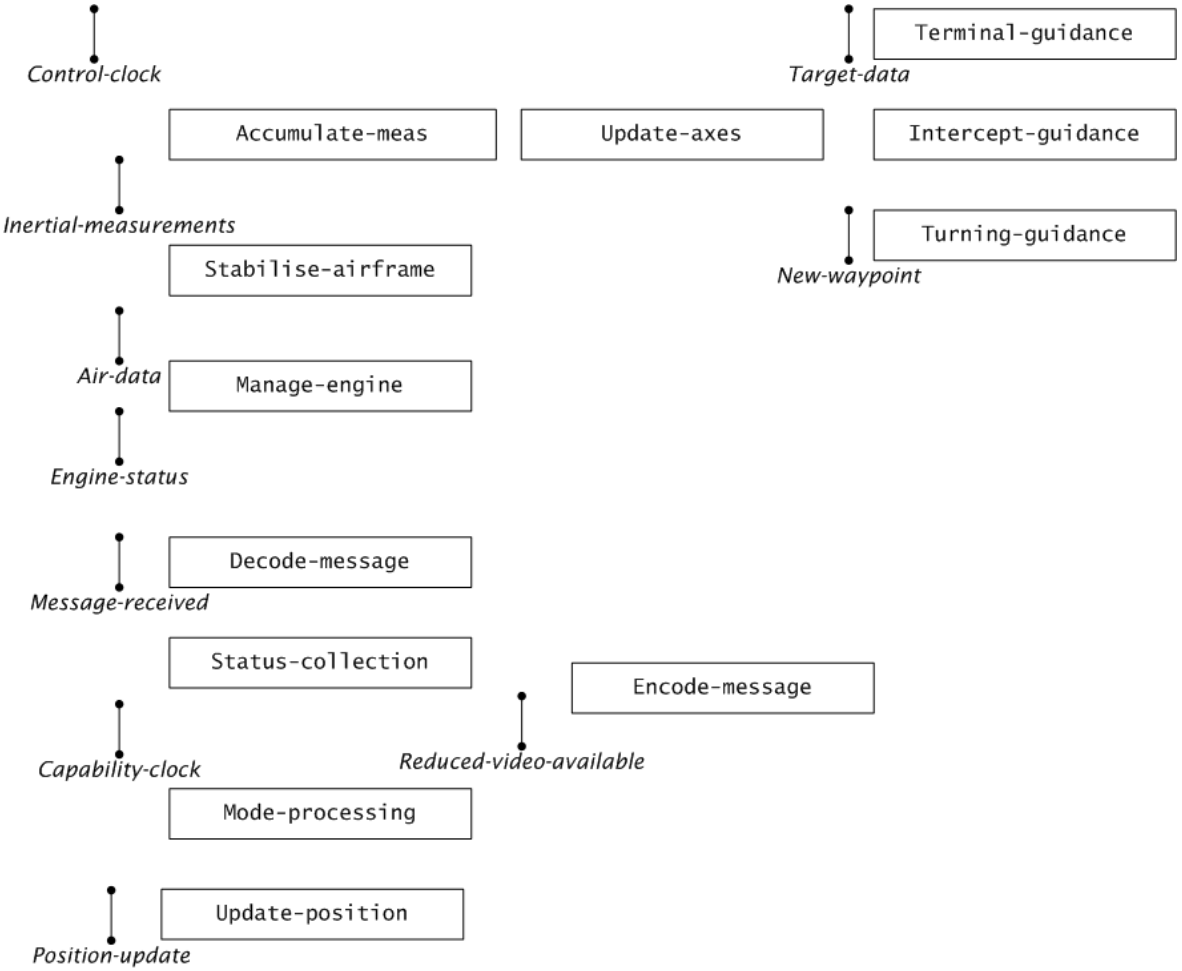


Figure 6.24: BASIC-BLOCK band — activities and events

Finally there are the relationships between the availability of inertial measurement data and the various processes dependent upon this information; these are:

$$\textit{Control-clock} \preceq \textit{Inertial-measurement} \prec \uparrow\texttt{Accumulate-meas} \tag{6.135}$$

$$\downarrow\texttt{Accumulate-meas} \preceq \uparrow\texttt{Update-axes} \tag{6.136}$$

$$\downarrow\texttt{Update-axes} \prec \uparrow\texttt{Terminal-guidance} \tag{6.137}$$

$$\textit{Target-data} \prec \uparrow\texttt{Terminal-guidance} \tag{6.138}$$

$$\downarrow\texttt{Update-axes} \prec \uparrow\texttt{Turning-guidance} \tag{6.139}$$

$$\textit{New-waypoint} \prec \uparrow\texttt{Turning-guidance} \tag{6.140}$$

$$\downarrow\texttt{Update-axes} \prec \uparrow\texttt{Intercept-guidance} \tag{6.141}$$

The above relationships are shown in Figure 6.24.

Again, as with the previous two bands there are no modes defined for this band, but again there are some constraints that can be identified against the modes defined in the higher level bands. In general the activities defined in this band are only related to the vehicle

when in flight, two exceptions however are the `Mode-processing`, `Status-collection` and `Update-position` activities which are essentially modeless being executed both pre- and post-launch. The activity related to the stability of the airframe is only executed post launch, hence:

$$\{\textbf{Pre-launch}\} \not\Rightarrow \{\texttt{Stabilise-airframe}\} \tag{6.142}$$

The three guidance related activities are strongly mode dependent:

$$\{\textbf{Engaging}\} \overset{inv}{\Rrightarrow} \{\texttt{Terminal-guidance}\} \tag{6.143}$$

$$\{\textbf{Navigate}\} \overset{inv}{\Rrightarrow} \{\texttt{Intercept-guidance}\} \tag{6.144}$$

$$\{\textbf{Turn}\} \overset{inv}{\Rrightarrow} \{\texttt{Turning-guidance}\} \tag{6.145}$$

The events related to the final engagement are also clearly mode dependent:

$$\{\textbf{Engaging}\} \Rrightarrow \{\textit{Reduced-video-available}, \textit{Target-data}\} \tag{6.146}$$

6.5.3 Identifying Inter-band relationships

Having competed the definitions of the bands in isolation from one another the inter-band relationships can now be completed. There are three aspects to the inter-band relationships, these are:

- The mappings between the bands themselves;
- The mappings between activities in lower bands and the events in the higher ones;
- The modal hierarchy mappings.

The first of these are defined by the granularity mappings, these has already been defined in Table 6.7 presented in Section 6.5.1.

The modal hierarchy mappings are relatively straightforward in that each mode in the MISSION^{V} band is associated with a similarly named mode model in the FLIGHT-PHASES band; the mappings between these are:

Pre-launch $\sqsubset$ *PrelaunchModel*
Transit $\sqsubset$ *TransitModel*
Loiter $\sqsubset$ *LoiterModel*
Aborting $\sqsubset$ *AbortingModel*
Engaging $\sqsubset$ *EngagingModel*

The mappings between the high level events and the lower level activities are somewhat more complex and extensive, a limited selection of these are developed in the following section.

Event ⟶ Activity mappings

There are many event ⟶ activity mappings that may be defined across the bands for this model, only a subset of those possible are described here.

Considering first the events defined for the VEHICLE-LIFE band, specifically those occurring in the context of the `Vehicle-available` activity, the *Mission-aborted*, *Mission-completed* and *Software-update* events. In all three cases there are direct mappings to corresponding activities in the next lower band (MISSION^{V}), thus:

$$\textit{Mission-aborted} \longrightarrow \texttt{Mission}^{A} \tag{6.147}$$

$$\textit{Mission-completed} \longrightarrow \texttt{Mission}^{C} \tag{6.148}$$

$$\textit{Software-update} \longrightarrow \texttt{Update-software} \tag{6.149}$$

In the MISSION^{V} there are several events that map to activities in the next-lower FLIGHT-PHASES band:

$$\textit{Built-in-test} \longrightarrow \texttt{Perform-vehicle-BIT} \tag{6.150}$$

$$\textit{Load-software} \longrightarrow \texttt{Reprogram-software} \tag{6.151}$$

$$\textit{Plan-rxd} \longrightarrow \texttt{Load-plan} \tag{6.152}$$

In addition to these there are number of event ⟶ event mappings; an example of these would be the mapping between the *Fire-command* event and the *Launch* event. There are also a number of events that are common to both bands, e.g. *Abort* and *Timeout*.

The mapping between the FLIGHT-PHASES and CAPABILITY band include the mapping of the *Local-BIT* event to the `Perform-local-BIT` activity and the *Power-on* event and the `Init` activity. The `Process-image` activity in the CAPABILITY band does not map one-to-one with an event in the FLIGHT-PHASES band, but, of course, there is a relationship due to the modal constraints; the *Target-acquired* event, in the FLIGHT-PHASES band, is "weakly" bound to the `Process-image` activity in that the event must occur in the context of the activity, but the activity is not defined by the event.

There are few pure event ⟶ activity mappings between the CAPABILITY and CONTROL bands, the only notable one being that between *Position-updated* and `Predict-position`. There is also a less well defined mapping between the `Process-image` activity in the

higher band and the set of activities relating to image processing, i.e `Image-extraction`, `Correlation`, `Tracking` and `Aimpoint-extraction`, in the lower band.

The final set of mappings between adjacent bands are those between the events in CONTROL and the activities present in BASIC-BLOCK. Examples here are mappings between:

$$\textit{Engine-update} \longrightarrow \texttt{Manage-engine} \tag{6.153}$$

$$\textit{Stabilise} \longrightarrow \texttt{Stabilise-airframe} \tag{6.154}$$

$$\textit{Axes-available} \longrightarrow \texttt{Update-axes} \tag{6.155}$$

There are also a number of mappings between non-adjacent bands an example here is that between the *Message-transmitted* event in the CAPABILITY band and the `Encode-message` activity in the BASIC-BLOCK band — these are in non-adjacent bands as the rate at which messages are sent is quite low and the processing duration to encode each message is short. Another example, not explicitly stated, is that of the `Mode-processing` activity in the BASIC-BLOCK band that is associated with every mode transition event in both the MISSION^{V} and FLIGHT-PHASES bands.

The model, although not complete, is extensive and is included here to show the level of detail the timebands framework can deliver.

6.6 Local command post model

Having completed the model for the Payload Delivery Vehicle the final component model, that of the Local Command Post, may now be developed.

An outline description of the Local Command Post was given earlier in this chapter in Section 6.1.2.3; here this system is considered in more detail, specifically with respect to it temporal dynamics. Figure 6.25 gives the context of the system, the key elements on which this model focuses are the Control Unit and the Operator Console — these being the elements that are central to this system. Multiple operator consoles may be connected to a single Control Unit allowing many vehicles to be controlled from one facility; additionally a single operator may be responsible for a number of missions, each in different phase. As noted earlier this system is of quite a different type from that of the payload delivery vehicle in that it does not have to meet the same hard real-time requirements and there is a high degree of operator interaction — however the overall performance of the system is dependent on the ability to process information and provide updates to the vehicle in a timely manner.

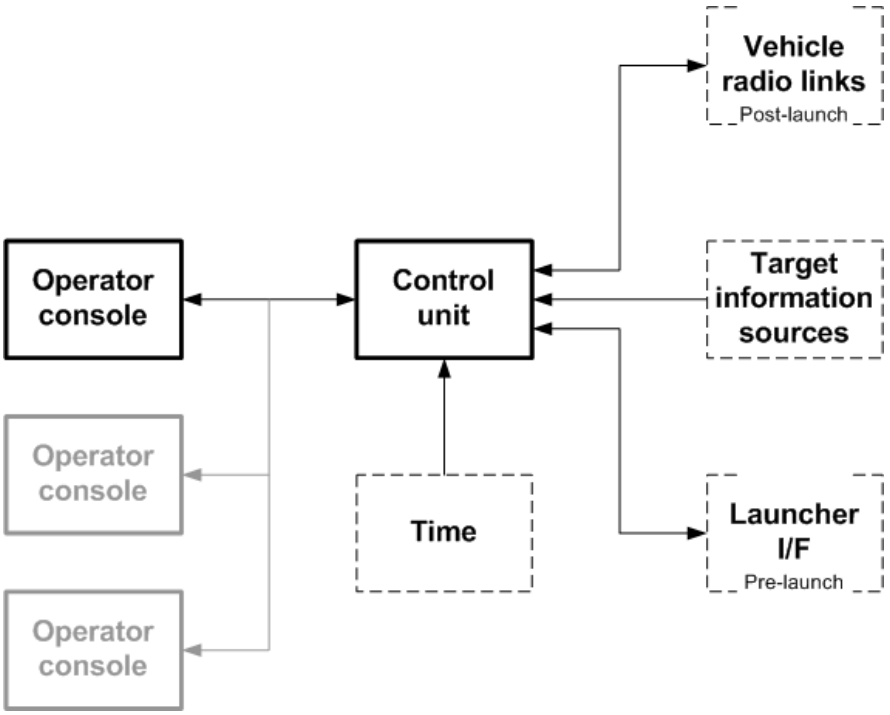


Figure 6.25: Local command post context

6.6.1 Identifying the bands

As for the previous component the following sections follow the process for constructing a timebands model that was defined in the previous chapter; this starts with the identification of the overall context. Again, as for the previous component, the "full" process is applied, and throughout these sections the description and discussions provided in the development of the initial system-of-systems model are assumed, and are not repeated unless there is a specific issue of interest.

Step 1 — Identifying the context of the model

As with the overall system-of-systems model there are many temporal aspects associated with the Local Command Post. Here again, in common with both the overall model and that of the payload delivery vehicle component, the focus is on the in-service usage of the system; here however a more "tactical" view is taken, limiting the scope to that of the system having been deployed. The wider ranging, more "strategic" issues regarding the life of the equipment, and its replacement/renewal are ignored. Again, as in the cases of the previous two models, the temporal issues associated with other phases of its life-cycle, such as design, development and disposal, are not considered.

Taking this more limited in-service perspective there are three fundamental aspect that have to be considered; these are:

- Interactions with the operators;
- Interactions with the vehicles;
- Overall management of the missions.

Quite clearly, as this system is the interface between the operators and the vehicles, the first two aspects are not totally separable; equally however their interactions cannot be so tightly coupled that the vehicle, which is intended to be largely autonomous, becomes heavily dependent on the behaviour of the operator throughout the course of the mission. The management of the missions is really the central role of this system, this aspect is perhaps best seen as being the "bridge" between the operator and the vehicles whilst also being the element that integrates the "tactical" information from other sources.

The system-of-systems model established some key temporal "drivers" such as mission duration and vehicle life, however there was little defined with respect to the Local Command and Post and especially its operators. The key issues with respect to the operators are:

- Shift patterns — how long can an operator be in control of a mission?
- Training — are all operators equally capable, how much training is required to be able to use the system in-service?
- Workload — how many vehicles can a single Local Command Post manage, what is the capacity of each operator, how long does it take to plan a mission?

It is worth noting that unlike the vehicle, the Local Command Post does not have a strongly defined service life. This is not particularly surprising because the system described here is only one of the possible approaches to controlling the vehicle, others being likely to emerge over the life of the system; thus the Local Command Post is important from the viewpoint of the capabilities it provides, rather than that of it being a particular asset. However there is clearly a desire that the system has a reasonable life with something in the range of $5 \ldots 10$ years being appropriate, though this will not be considered further in this case study.

Step 2 — Identifying the stakeholders

As was the case for the payload delivery vehicle the stakeholders for this system are a subset of those defined for the overall system-of-systems model. Not unsurprisingly, given the model is focused on a tactical view of the in-service use of the system, the

stakeholders are similar to those for the payload delivery vehicle. Together with their particular interests, these are:

- The local commanders — the capacity of the system, the workload it places on the operators and the resources required to support the use of the system;
- The mission planners — the ease or otherwise of the detailed planning of missions;
- The operators — the dynamics of the interactive control of the mission, training requirements etc.[14]

Here, unlike the case of the payload delivery vehicle where it was debatable as to whether the latter group were stakeholders or actually part of the system, the operators are truly stakeholder as their involvement with the vehicle is far more wide-ranging than simply the control of a mission — they are clearly also an important part of the system given their close involvement with the control of missions.

Given the close coupling of the operators and local commanders with the system, aspects such as training and the potential for promotion also need to be considered — though as these aspects may be viewed as being more strategic in nature they will not be considered further in the development of this model.

Step 3 — Identifying the temporal boundaries

Identifying the upper temporal bound of this component system is far less straightforward that it was for either the overall system-of-systems or payload delivery vehicle models. Three possible approaches to determining these bounds are:

- Assume the upper bound to be that of the system-of-systems model, 40 years, and capture system upgrade, refurbishment and replacement in a high level band;
- Set the upper bound at a level relating to some specific strategic dynamic of the Local Command Post such as its deployment in the field, something like a period of two years as defined in the system-of-systems model; or
- Take the tactical context of the model as the dominant guide and define the bounds in terms of the actions that are performed in this context, the deployed for operations.

[14] The roles of the planning missions and operating the equipment to control the mission may be fulfilled by the same personnel, however the tasks are quite different.

Given the tactical viewpoint identified in Step 1 above, in this case the latter is the most desirable as any detail in any higher level bands will be somewhat arbitrary and irrelevant. Using this latter approach, the upper bound is set to accommodate the use of the system to conduct a number of missions at a fixed location; this leads to an upper bound, an albeit somewhat arbitrary one, of one month.

Definition of the lower bound could follow the approach taken for the Payload Delivery Vehicle and be low enough to accommodate the fine details of the software executing in the Control Unit; though in this case it is likely that the value of this bound would have to be smaller as the computing systems used here are more powerful and have shorter instructions cycle times than those used in the vehicle. However recalling that the reason for considering such fine-grained detail in the case of the vehicle was purely to facilitate a more extensive exploration the timebands framework; taking such an approach here would both result in a level of detail that is not really necessary for this model, and be unlikely to add further to the appreciation or understanding of the framework more generally.

Given this, the lower bound can be set to a value of one millisecond — more than sufficient to accommodate any activities resulting from the communications between this system and the vehicle or other information sources, and certainly adequate to capture the dynamics of the operator interactions.

In summary, the upper bound of the model is one month and the lower bound one millisecond; these values give the definitions for the first parameters of the highest and lowest bands of this model, namely the TACTICAL and LCP-BLOCKS bands.

Step 4 — Identifying any modal behaviour

In the development of the system-of-systems model significant modal behaviour was identified, and this was reflected through into the model for the Payload Delivery Vehicle; the question here is: should this modal behaviour also be reflected into this model?

Put simply the answer is no. The Local Command Post has a set of functions that are wider than those relating to the simple preparation and control of a single mission. Multiple missions may be undertaken simultaneously and the different aspects of the separate missions may happen concurrently; for example, whilst one mission is entering the end-game another might be being planned, whilst for another the vehicle has just been launched. Clearly at any instant it would be difficult to identify a particular mode model that applied consistently across the entire system, other than possibly the system being "active" or "inactive" — a view which is of little interest here.

It is worth noting that if a more strategic view had been taken, and other aspects of the system been considered then there are perhaps a set of higher level modes, e.g. Maintenance, Training and Operation, that could be used to develop a mode model. However these, and the transitions between them, do not exhibit particularly interesting temporal dynamics in the context of this model. Thus, whilst it is also undoubtedly the case that the modes of the mission will be reflected in some of the more detailed aspects of the local command post, there is no overall mode model of any significance that gives rise to a particular time band.

Step 5 — Identifying any tightly coupled human interactions

Unlike the case of the payload delivery vehicle here there is considerable human interaction, some of which is tightly coupled. The degree of coupling and the associated temporal dynamics vary for the different tasks undertaken. It is perhaps best to consider these interactions from the perspective of the different participants rather than from the perspective of the system. The first stakeholders to consider are the planners.

The planners' role is twofold: firstly they define the route to a loitering location, and secondly they identify particular targets and prepare routes for the final engagement. They also plan routes to an abort zone, but this is of no particular interest here as the dynamics are similar to those in the other two cases. For the first of the two the process is essentially independent of any particular target and has only limited time constraints. The planning involves the selection of a set of waypoints between the launch site and the loitering zone, taking account of constraints, such as available safe lanes, that might be placed on the route — the actual route is generated by the Control Unit computer. The planning of the final engagement is a more "immediate" task in that the target is potentially moving and hence time assumes a greater importance. The only action the planner has to undertake is the association of the vehicle with a particular target. Only the latter could loosely be described as tightly coupled.

In the case of the operators the interactions with the system can be considered in three phase:

- Those occurring pre-launch;
- Those in transit;
- Those during the final engagement.

In the first of these the operator's role is to apply power to the vehicle, assess the results of the BIT process, load the mission plan, and finally initiate the launch of the vehicle. None of these has any particularly demanding temporal constraints. Similarly in the transit

case the operator is simply monitoring the vehicle status, only if the status indicates some abnormal behaviour does the operator have any action to take, and that is to simply abort the mission — given the dynamics of the vehicle in this phase this does not give rise to any significant temporal demands.

In the third case, that of the final engagement, the operator has a much more tightly coupled interaction with the system. During this phase the operator has to confirm the engagement of the target; the confirmation being based on the assessment of imagery returned from the vehicle. Clearly the responses here have to be timely to minimise the possibility of the target being "lost" — the operator has only a few seconds to make the decisions before the vehicle autonomously aborts the mission.

The final group to consider are the commanders. Here there is little that could be described as human interaction, things are more organisational in nature and can be considered later in that context.

In terms of bands it would appear that that there are two sets of dynamics above: those related to tasks and "deliberate" action; and those related to the "immediate" reactions to the situation. These are probably best "inspired" by Newell's *Rational* and *Cognitive* bands respectively — taking the mid-point of each to define the granularity of the band. Given the very specific nature of the roles it may be possible to refine Newell's definitions for these bands, but is not clear at this stage that this would bring any benefits.

In summary there are two bands that relate to human interaction:

RATIONAL For the deliberate actions, especially overall planning process, the pre-launch preparation and in-flight monitoring tasks; it has an upper bound of eight hours, a granularity of ten minutes and a precision of one minute;

COGNITIVE For the immediate interactions, especially those relating to the final phase of the mission; it has an upper bound of one minute, a granularity of one second and a precision of 100 milliseconds.

Step 6 — Identifying any modal-human interactions

Given no modal behaviour was identified in Step 4 above there is no basis for the definition a band based on the interactions between the modes of the systems and the human participants.

Step 7 — Identifying and partitioning the remaining organisational bands

So far the upper bound of the model has been based on the dynamics of the day-to-day operational use of the system, the minimum duration for which was identified as being

related to the shift patterns of the operators. With the upper bound of the RATIONAL band for human interaction being defined as eight hours and the upper bound of the TACTICAL band being defined as one minute, the question is: is there a need for an additional band between these TACTICAL and RATIONAL bands?

If the numerical relationship between the ranges of these bands is considered there is a ratio of approximately 50:1, quite a small difference, and certainly one that is too small to allow for the inclusion of an additional band that would demonstrate any ordering with its neighbours. Given this perhaps an alternative, better, question is: can the TACTICAL and RATIONAL bands be combined to address all of the temporal dynamics at the top end of the model? Taking such an approach would result in a significantly wide band that encompassed actions that were both related to the deliberate acts of a single person, and the dynamics of the overall system, which can include many participants. This would result in the confusion of the two sets of dynamics, which is clearly undesirable — thus the bands should not be combined.

In summary, the model contains a single organisational band, the TACTICAL band with a range of one month, a granularity of eight hours and a precision of one half hour.

Step 8 — Identifying and partitioning the remaining technology bands.

With the lower bound of the model being set at one millisecond and the lowest human interaction band being set at one second there is no real scope for introducing an additional technological band, hence there is a single technological band, LCP-BLOCKS; this has a range of one second, a granularity of one millisecond; again, as for the previous two models, its precision is left undefined. So far as it characteristics are concerned this could be mapped to Newell's BIOLOGICAL band, however such a mapping would be inappropriate as this band is more concerned with the functions of the Control Unit rather than with the lower level processes of the operators and planners.

Step 9 — Determine the granularity relationships between bands and the precision of each band

The timebands model for the Local Command Post consists of four bands; in order, and starting with the highest, these are:

- TACTICAL;
- RATIONAL;
- COGNITIVE;
- LCP-BLOCKS.

Clearly it would be possible to introduce additional lower-level bands to accommodate the dynamics of the detailed software and hardware implementations, but, as discussed earlier, this is unnecessary for this model. The characteristics of the bands identified in the sections above are summarised in Table 6.14, and the inter-band granularity mappings are given in Table 6.15.

Band	Upper bound/Scale	Granularity	Precision
TACTICAL	1 month	8 hours	½ hour
RATIONAL	8 hours	10 minutes	1 minute
COGNITIVE	1 minute	1 second	100 ms
LCP-BLOCKS	1 second	1 ms	–

Table 6.14: Local Control Post model band characteristics

From	To		
	TACTICAL	RATIONAL	COGNITIVE
RATIONAL	48	–	–
COGNITIVE	2.8×10^4	600	–
LCP-BLOCKS	1.68×10^{10}	6.0×10^5	1000

Table 6.15: Local command Post inter-band granularity mappings

For this model the mappings between adjacent bands follow Newell's spacings except in the case of the mapping between TACTICAL and RATIONAL. In this case the value is smaller that Newell's spacings, but is of the same order as for the more narrowly separated bands in the platform delivery vehicle given in Table 6.7. The justification for such a small separation is to be found in the selection of the temporal bounds for the model, the upper limit was chosen to accommodate the shift durations and patterns of the operators and as such the separation is probably appropriate. This band spacing issue will be considered further in the evaluation and assessment of the models and framework presented in the following chapter.

6.6.2 Populating the model — artefact identification, placement and relationships

Having defined the bands and the fundamental relationships between them, the next step in the process is to identify the artefacts and map these to the bands and define their inter-relationships considering each band independently. Each of the sections below provides a description of each band including formal definitions of elements and relationships where appropriate. As with the model of the vehicle developed earlier it should be noted

Band name:	TACTICAL		
Granularity:	8 hours		
Precision:	½ hour		
Events:	*Setup*		Preparation of the deployed facility
	Assess-engagement		Assess the results of the engagement
	Commence-mission		The initiation of an individual mission
	Complete-mission		The completion of an individual mission
	Mission-command		Command to undertake a set of missions
	Plan-routes		Planning and re-planning of route(s) from launch site to the loitering zone, or from one loitering zone to another
	Disestablish		The end of a deployment
Activities:	`Shift`	1	An operator shift (8 hours)
	`Control`	1	Overall control of the local command post, a "continuous" background activity
	`Perform-missions`	1 .. 20	Execution of the missions
	`Record`	1 .. 25	Recording of missions, a "continuous" background activity

Table 6.16: TACTICAL band definition

that the definitions of the bands are incomplete, with only a limited selections of the possible events and activities being defined; again those included ensure a wide coverage of the different aspects of the framework whilst avoiding excessive detail and repetition.

TACTICAL

The TACTICAL band is associated with the management of the system once deployed, and the provision of the resources to enable the system to be operated. The `Shift` activity defines the unit of working time for any operator or planner; at this level in the model its impact is restricted to those activities related to an individual mission or a single operator; however it serves to define boundary events in lower bands across which some activities must not span. The events and activities together with the general characteristics of the band are defined in Table 6.16.

There are a few fundamental precedence relationships that can be stated relating to the initialisation of the system: both of the *Mission-command* and *Setup* events must have occurred prior to the start of the `Perform-missions` activity. More formally these relationships are:

$$\mathit{Setup} \preceq \uparrow\texttt{Perform-missions} \tag{6.156}$$

$$\mathit{Mission\text{-}command} \preceq \uparrow\texttt{Perform-missions} \tag{6.157}$$

The Record activity simply records the use of the system from the point when it has been setup, the end of the activity is dependent on the "disestablishment" of the system; hence there is the simple precedence relationship:

$$\textit{Setup} \preceq \uparrow\texttt{Record} \wedge \downarrow\texttt{Record} \preceq \textit{Disestablish} \tag{6.158}$$

The Control activity defines the role of the local commander, these are bound to the period of the shift, a number of occurrences of the Control activity may occur during the course of the Perform-missions activity; the precedence relationships in this case are:

$$\uparrow\texttt{Shift} \preceq \uparrow\texttt{Control} \wedge \downarrow\texttt{Control} \preceq \downarrow\texttt{Shift} \tag{6.159}$$

$$\uparrow\texttt{Perform-missions} \preceq \uparrow\texttt{Control} \wedge \downarrow\texttt{Control} \preceq \downarrow\texttt{Perform-missions} \tag{6.160}$$

Given the granularity and precision of this band, and the variability in duration of an individual mission it is not possible to define a mission as either an activity or an event, instead an individual mission is defined in terms of two event: *Commence-mission* and *Complete-mission*. All of the events *Plan-routes*, *Commence-mission*, *Complete-mission* and *Assess-engagement* are defined within the context of the Perform-missions and Shift activities, with there being a general precedence between the events in the order given above. As the only occurrences of the Shift activity of interest are those which occur within the context of the Perform-missions activity the following relationships may be stated:

$$\uparrow\texttt{Shift} \prec \textit{Plan-routes} \prec \downarrow\texttt{Shift} \tag{6.161}$$

$$\uparrow\texttt{Shift} \prec \textit{Commence-mission} \prec \downarrow\texttt{Shift} \tag{6.162}$$

$$\uparrow\texttt{Shift} \prec \textit{Complete-mission} \prec \downarrow\texttt{Shift} \tag{6.163}$$

$$\uparrow\texttt{Shift} \prec \textit{Assess-engagement} \prec \downarrow\texttt{Shift} \tag{6.164}$$

As these events relate to a single mission, and there might be many occurrences of these within a the period of the Perform-missions activity it is not possible to give a generalised precedence relationship as missions may overlap, and missions do not have to complete in the order in which they commenced. However, considering a single mission n the precedence relationships can be defined, and these are:

$$\textit{Plan-routes} \prec \textit{Commence-mission}^n \tag{6.165}$$

$$\textit{Commence-mission}^n \prec \textit{Complete-mission}^n \tag{6.166}$$

$$\textit{Complete-mission}^n \prec \textit{Assess-engagement}^n \tag{6.167}$$

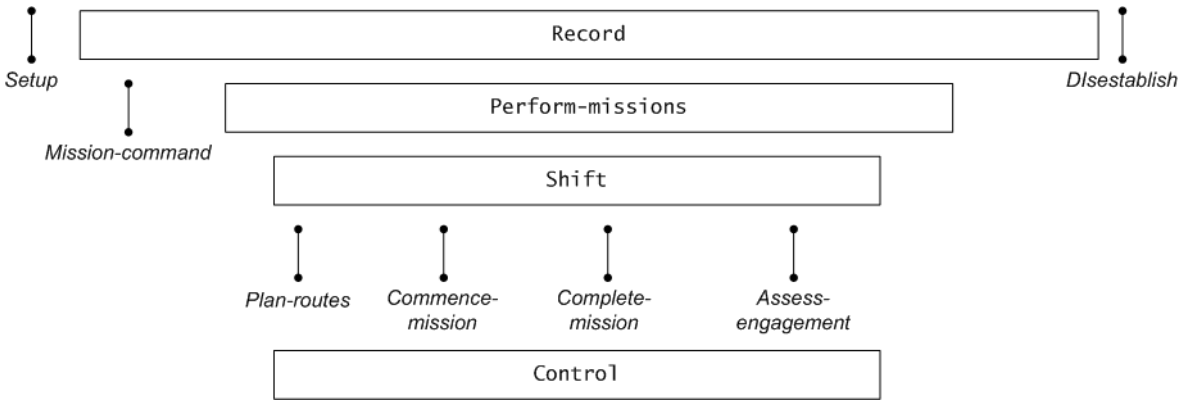


Figure 6.26: TACTICAL band — activities and events

These, of course, are not bound to a single shift.

The above precedence relationships are shown in Figure 6.26.

RATIONAL

The RATIONAL band is the higher of the two bands associated with human interaction — the one associated with the deliberate acts of an operator in the use of the system; this strongly contrasts with the previous band where the focus was on the system as a whole and a community of operators. As noted earlier, so far as the interaction of the individual operator with the vehicle is concerned, there are three phases to each mission: preparation; transit; and target engagement. Over the course of a vehicle mission the operators have a number of tasks to perform from route planning to target selection. The events and activities together with the general characteristics of the band are defined in Table 6.17.

Given the view taken here is that of the single operator, the fundamental constraint for this band is that all actions performed by the operator are constrained by the `Shift` activity defined in the TACTICAL band, which is here propagated to the *Start-shift* and *End-shift* events; hence the following fundamental precedence relationships can be stated:

$$\textit{Start-shift} \preceq \uparrow\texttt{Setup} \preceq \textit{End-shift} \tag{6.168}$$

$$\textit{Start-shift} \preceq \uparrow\texttt{Prepare} \preceq \textit{End-shift} \tag{6.169}$$

$$\textit{Start-shift} \prec \uparrow\texttt{Plan-route} \preceq \textit{End-shift}\textit{Start-shift} \prec \uparrow\texttt{Monitor-transit} \preceq \textit{End-shift} \tag{6.170}$$

$$\textit{Start-shift} \prec \uparrow\texttt{Confirm-engagement} \preceq \textit{End-shift} \tag{6.171}$$

Band name:	RATIONAL		
Granularity:	10 minutes		
Precision:	1 minute		
Events:	*Initial-data-available*	Initial site data available (location, time, air lanes, maps)	
	Initiate-BIT	Initiate vehicle BIT	
	Analyse-BIT-results	Assess results of vehicle BIT	
	Launch	Initiate vehicle launch	
	Vehicle-available	Vehicle available for pre-launch tests	
	Confirm	Confirm engagement with target	
	Reject	Reject target engagement	
	Load-plan	Load planned route	
	Route-end	The vehicle has reached loitering zone	
	Re-route	Load re-planned route	
	Tactical-data-available	Data available from tactical sources	
	Vehicle-status	Vehicle status update available	
	Target-selected	A target has been selected	
	Start-shift	The start of an operator shift	
	End-shift	The end of an operator shift	
	Assess-progress	Assessing vehicle status data	
	End-of-engagement	End of an engagement	
Activities:	`Prepare`	6 .. 12	Perform the initial vehicle preparation
	`Monitor-transit`	1 .. 50	Monitor the status of the vehicle on route to the loitering zone
	`Plan-route`	3 .. 6	Plan the route to the loitering zone
	`Select-target`	1 .. 2	Select target
	`Confirm-engagement`	1 .. 2	Confirm target for final engagement
	`Analyse-tactical-data`	1 .. 1	Analyse tactical data
	`Assess-results`	1 .. 6	Assess the results of mission

Table 6.17: RATIONAL band definition

Additionally no missions or preparation for missions can be undertake until the Setup activity has been completed and a vehicle is available, additionally Setup cannot occur until site data is available; hence the following relationships apply:

$$\textit{Initial-data-available} \preceq \uparrow\texttt{Setup} \tag{6.172}$$

$$\downarrow\texttt{Setup} \preceq \uparrow\texttt{Prepare} \tag{6.173}$$

$$\textit{Vehicle-available} \preceq \uparrow\texttt{Prepare} \tag{6.174}$$

$$\downarrow\texttt{Setup} \preceq \uparrow\texttt{Plan-route} \tag{6.175}$$

Within the context of the Prepare activity all of the pre-launch events and activities take place, hence the following precedence relationships hold:

$$\uparrow\texttt{Prepare} \prec \textit{Initiate-BIT} \prec \textit{Analyse-BIT-results} \tag{6.176}$$

$$\textit{Analyse-BIT-results} \prec \textit{Load-plan} \prec \textit{Launch} \preceq \downarrow\texttt{Prepare}$$

Additionally the Load-plan event requires that a plan has been created, hence:

$$\downarrow\texttt{Plan-route} \preceq \textit{Load-plan} \tag{6.177}$$

Moving on to consider the relationships between activities and events for the phase of the mission where the operator is monitoring the vehicle in flight, transiting from the launch site to the loitering zone; the monitoring activity does not commence until the launch of a vehicle has been initiated, hence:

$$\textit{Launch} \preceq \uparrow\texttt{Monitor-transit} \tag{6.178}$$

This also implies a precedence relationship between the end of Prepare and the start of Monitor-transit; hence:

$$\downarrow\texttt{Prepare} \prec \uparrow\texttt{Monitor-transit} \tag{6.179}$$

Whilst in the context of the Monitor-transit activity the operator is restricted to simply monitoring the status, and the following relationships apply:

$$\uparrow\texttt{Monitor-transit} \preceq \textit{Vehicle-status} \preceq \downarrow\texttt{Monitor-transit} \tag{6.180}$$

The *Route-end* event occurs in the context of the Monitor-transit activity, as does any *Re-route* event, with *Route-end* preceding the *Re-route*; hence:

$$\uparrow\texttt{Monitor-transit} \prec \textit{Route-end} \preceq \downarrow\texttt{Monitor-transit} \tag{6.181}$$

$$\uparrow\texttt{Monitor-transit} \prec \textit{Re-route} \preceq \downarrow\texttt{Monitor-transit} \tag{6.182}$$

$$\textit{Route-end} \preceq \textit{Re-route} \tag{6.183}$$

The final phase of the mission is that of target engagement. This can only start once the transit phase has complete and a target has been selected; thus the following relationships apply:

$$\downarrow\texttt{Monitor-transit} \preceq \uparrow\texttt{Confirm-engagement} \tag{6.184}$$

$$\downarrow\texttt{Monitor-transit} \equiv \textit{Target-selected} \tag{6.185}$$

$$\textit{Target-selected} \preceq \uparrow\texttt{Confirm-engagement} \tag{6.186}$$

The selection of a target is dependent on the availability of tactical target data and its subsequent analysis, hence:

$$\textit{Tactical-data-available} \preceq \uparrow\texttt{Analyse-tactical-data} \tag{6.187}$$

$$\downarrow\texttt{Analyse-tactical-data} \preceq \textit{Target-selected} \tag{6.188}$$

Once in the `Confirm-engagement` activity the operator monitors the target, ultimately confirming of rejecting the engagement; the *End-of-engagement* event signifies that the engagement has been completed (though with no indication of the outcome, hence the following relationships exist:

$$\downarrow\texttt{Confirm-engagement} \equiv \textit{End-of-engagement} \tag{6.189}$$

$$\uparrow\texttt{Confirm-engagement} \preceq \textit{Assess-progress} \preceq \textit{Confirm} \preceq \downarrow\texttt{Confirm-engagement} \tag{6.190}$$

$$\uparrow\texttt{Confirm-engagement} \preceq \textit{Assess-progress} \preceq \textit{Reject} \preceq \downarrow\texttt{Confirm-engagement} \tag{6.191}$$

Finally once the engagement has been complete there is an activity to assess its effectiveness, hence:

$$\textit{End-of-engagement} \prec \uparrow\texttt{Assess-results} \tag{6.192}$$

The above precedence relationships are shown in Figure 6.27.

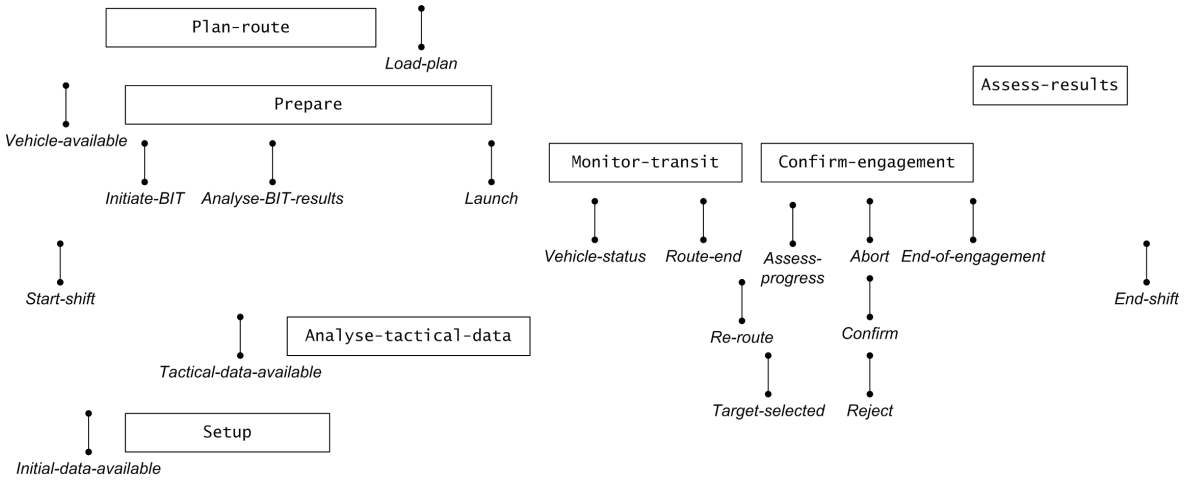


Figure 6.27: RATIONAL band — activities and events

COGNITIVE

The COGNITIVE band is the second and lower of the two bands associated with human interaction — the one associated with the immediate response of the operators to "urgent" information from the vehicle and other tactical sources, and for the detailed interaction with the computing facility for the creation of plans. Similar to the RATIONAL band the focus is on the actions of a single operator, and as with that band the interactions with the vehicle can be split into the three phases: preparation, transit monitoring and engagement management. The planning task can be viewed as a separate function, The events and activities together with the general characteristics of the band are defined in Table 6.18.

Fundamental precedence relationships exist relating to the operators only being able to use the system once "signed-on", however these are omitted as they do not provide particularly useful insights. There are a number of relationships relating to the sequence of activities which act to bring the system into a usable state; these are:

$$\downarrow\texttt{Enter-site-data} \preceq \uparrow\texttt{Enter-safe-lanes} \tag{6.193}$$

$$\downarrow\texttt{Enter-safe-lanes} \preceq \uparrow\texttt{Load-terrain-maps} \tag{6.194}$$

There are a number of precedence relationships that can be stated, the first set of which relate to the preparation of the vehicle; this starts with the vehicle becoming available, and is followed by the application of power and the command to perform pre-launch tests and assessing the results of the tests; hence:

$$\textit{Vehicle-available} \prec \textit{Apply-power} \prec \textit{Initiate-BIT} \prec \downarrow\texttt{Analyse-BIT-results} \tag{6.195}$$

At the end of the analysis of the BIT results, the power can be removed if the vehicle is deemed unserviceable; the precedence relationship in this latter case is:

$$\downarrow\texttt{Analyse-BIT-results} \prec \textit{Remove-power} \tag{6.196}$$

In the case of the vehicle being serviceable the plan can be loaded, but only if the plan has been created; the following relationships relate to the creation of a plan, which is initiated by the *Planning-demand* event:

$$\textit{Planning-demand} \prec \uparrow\texttt{Start-plan} \tag{6.197}$$

$$\downarrow\texttt{Start-plan} \prec \uparrow\texttt{Enter-waypoint} \tag{6.198}$$

$$\downarrow\texttt{Enter-waypoint} \prec \uparrow\texttt{Complete-plan} \tag{6.199}$$

Band name:	COGNITIVE		
Granularity:	1 second		
Precision:	100 milliseconds		
Events:	*Receive-status*		Status received from vehicle
	Planning-demand		Orders to create route plan — loitering location and routing constraints
	Vehicle-available		Vehicle available for a mission for pre-launch tests
	Picture-update		Update of tactical picture
	Apply-power		Power-up a vehicle
	Remove-power		Power-down a vehicle
	Load-plan		Load planned route
	Route-end		Vehicle has reached loitering zone
	Confirm		Confirm engagement with target
	Reject		Reject target engagement
	Initiate-BIT		Command start of BIT tests
Activities:	`Sign-on`	3 .. 5	Logging into the system at the start of a shift
	`Sign-off`	3 .. 5	Logging out from the system at the end of a shift
	`Enter-site-data`	15 .. 30	Entering the site data to setup the system
	`Enter-safe-lanes`	15 .. 30	Entering the air-lane data as part of the setup
	`Load-terrain-maps`	15 .. 30	Loading the maps data needed for route planning — part of the system setup
	`Start-plan`	1 .. 3	Commence the creation of a new route plan
	`Complete-plan`	1 .. 3	Finalise a route plan
	`Enter-waypoint`	5 .. 10	Enter a waypoint for a plan
	`Initiate-launch`	5 .. 10	Perform the launch initiation sequence
	`Abort-mission`	5 .. 10	Perform the abort mission sequence
	`Engage-target`	1 .. 5	Perform the final target engagement sequence
	`Monitor-target`	3 .. 7	Monitor target feedback from the vehicle
	`Assess-status`	10 .. 40	Assess the status of the vehicle whilst in transit to the loitering zone, or whilst in the loitering zone
	`Analyse-BIT-results`	10 .. 20	Analyse the results of the BIT tests
	`Select-target`	20 .. 50	Select the target to engage

Table 6.18: COGNITIVE band definition

Clearly there will be a number of waypoints in any plan, hence there will be a large number of the occurrences of the `Enter-waypoint` activity. Thus the *Load-plan* event is dependent on both the serviceability of the vehicle and the availability of the plan; hence:

$$\downarrow\texttt{Analyse-BIT-results} \prec \textit{Load-plan} \tag{6.200}$$

$$\downarrow\texttt{Complete-plan} \prec \textit{Load-plan} \tag{6.201}$$

Once the plan has been loaded the vehicle is ready to be launched and the operator can start to enter the launch commands, hence:

$$\textit{Load-plan} \prec \uparrow\texttt{Initiate-launch} \tag{6.202}$$

The second phase of the mission can commence once the `Initiate-launch` activity has been completed, the only task for the operator in this phase is to periodically assess status of the vehicle, thus there is the simple relationship:

$$\downarrow\texttt{Initiate-launch} \preceq \textit{Receive-status} \preceq \uparrow\texttt{Assess-status} \tag{6.203}$$

Once the *Route-end* event occurs the operator can command the vehicle to engage a target (assuming that one has been selected for this vehicle):

$$\downarrow\texttt{Select-target} \prec \uparrow\textit{Engage-target} \tag{6.204}$$

$$\textit{Route-end} \prec \uparrow\textit{Engage-target} \tag{6.205}$$

Following the start of the engagement of the target the operator has to monitor the progress of the engagement and confirm or reject the final phase of the mission.

$$\textit{Picture-update} \prec \uparrow\texttt{Monitor-target} \tag{6.206}$$

$$\downarrow\textit{Engage-target} \prec \textit{Confirm} \tag{6.207}$$

$$\downarrow\textit{Engage-target} \prec \textit{Reject} \tag{6.208}$$

Finally at any point after the initiation of the launch of the vehicle the mission may be aborted, hence the following simple relationship applies:

$$\downarrow\texttt{Initiate-launch} \prec \uparrow\textit{Abort-mission} \tag{6.209}$$

The above precedence relationships are shown in Figure 6.28.

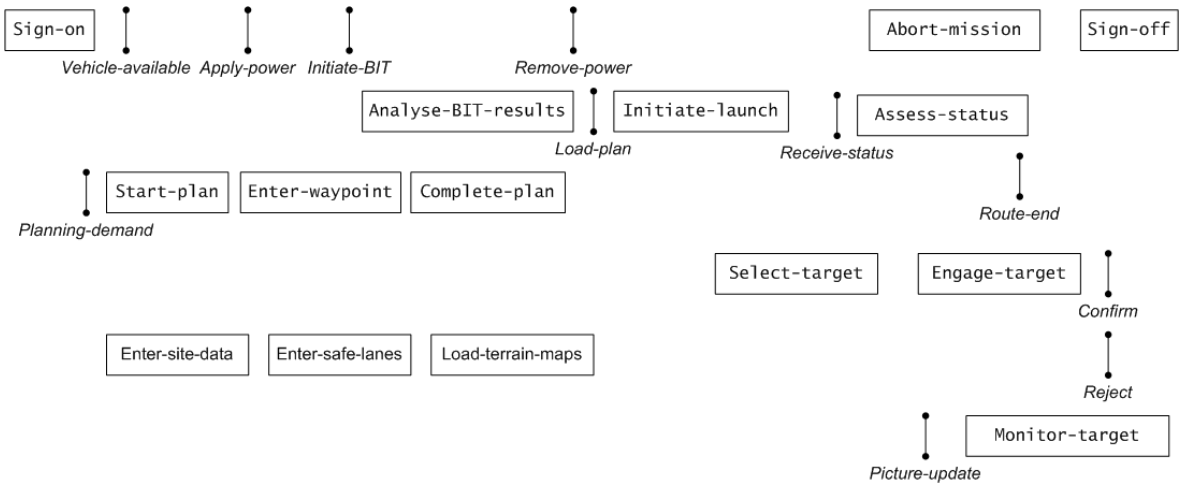


Figure 6.28: COGNITIVE band — activities and events

LCP-BLOCKS

The LCP-BLOCKS band is the lowest band defined for this model and is associated with the dynamics of the interactions between the Control Unit computer system and the external information sources. In this band the focus shifts from the individual operator to that of the controlling computer — hence multiple missions and plans are again considered together as they were earlier in the TACTICAL band. The events and activities together with the general characteristics of the band are defined in Table 6.19.

There are a number of precedence relationships that can be stated relating to the creation of the plan. The system has initially to be setup with data appropriate to the location of the deployment, information such as the allowable air lanes, location altitude, and terrain maps. Once this is loaded the planning may commence, a sequence of waypoints are input by the operator between each of which detailed routes are calculated by the computer; ultimately the plan is finalised and becomes available to be loaded into the vehicles; the precedence of these operations is given below:

$$\downarrow\texttt{Initial-setup} \prec \uparrow\texttt{Initialise-plan} \tag{6.210}$$

$$\downarrow\texttt{Initialise-plan} \prec \textit{Valid-waypoint} \preceq \uparrow\texttt{Calculate-course} \tag{6.211}$$

$$\downarrow\texttt{Calculate-course} \prec \uparrow\texttt{Finalise-plan} \tag{6.212}$$

$$\downarrow\texttt{Finalise-plan} \equiv \textit{Plan-available} \tag{6.213}$$

The tactical picture is maintained using data received from both the vehicle and other tactical sources, additionally the picture maintenance is supported by the prediction of both the target and vehicle positions (these are maintained periodically, in the absence of

Band name:	LCP-BLOCKS		
Granularity:	1 millisecond		
Precision:	–		
Events:	*Vehicle-data-rxd*		Status information received from the vehicle
	Tactical-data-rxd		Data received from tactical sources, e.g. targets
	Valid-waypoint		Validated waypoint input by planner
	Plan-available		Plan completed and available to be loaded into a vehicle
	Update-clock		Stimulus for position prediction
	Command-txd		Command sent to vehicle
Activities:	`Initial-setup`	15 .. 30	Setup of system after deployment
	`Initialise-plan`	1 .. 2	Start of a planning task, initialisation with "process" data
	`Finalise-plan`	5 .. 10	Completion and validation of a plan
	`Calculate-course`	2 .. 15	Calculation of route between two basic waypoints
	`Weight-targets`	2 .. 30	Weighting of targets to identify engagement possibilities
	`Predict-location`	5 .. 10	Update of predicted position of vehicle and target
	`Form-command`	1 .. 5	Creation of a command to be sent to the vehicle
	`Update-picture`	30 .. 70	Updating the tactical picture

Table 6.19: LCP-BLOCKS band definition

updated data from the vehicle or tactical sources, on an occurrence of the *Update-clock* event); the following relationships apply:

$$\textit{Vehicle-data-rxd} \prec \uparrow\texttt{Update-picture} \tag{6.214}$$

$$\textit{Tactical-data-rxd} \prec \uparrow\texttt{Update-picture} \tag{6.215}$$

$$\textit{Update-clock} \prec \uparrow\texttt{Predict-location} \prec \downarrow\texttt{Predict-location} \prec \tag{6.216}$$

$$\uparrow\texttt{Update-picture} \tag{6.217}$$

At this low level the messages (commands) to the vehicle are formed and dispatched, these follow the simple precedence relationship:

$$\downarrow\texttt{Form-command} \prec \textit{Command-txd} \tag{6.218}$$

The commands are sent periodically, hence the relationship for the initiation of the creation of the command is the update clock for this band, hence:

$$\textit{Update-clock} \preceq \uparrow\texttt{Form-command} \tag{6.219}$$

The final relationship of interest in this band is that between the operator and the tactical picture. The operator can modify the picture by weighting targets, i.e. deciding which

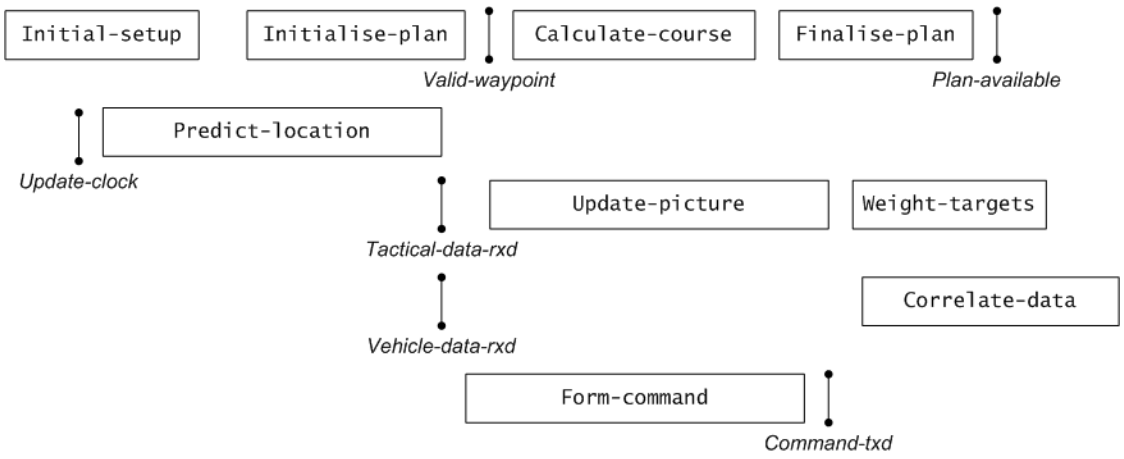


Figure 6.29: LCP-BLOCKS band — activities and events

object is the most likely to be a target. Thus there is a relationship between the picture being available and the assessment of the potential targets, hence:

$$\downarrow\texttt{Update-picture} \prec \uparrow\texttt{Weight-targets} \tag{6.220}$$

The above precedence relationships are shown in Figure 6.29.

6.6.3 Identifying the inter-band relationships

As with the previous component model, having completed the definition of the bands in isolation from one another the inter-band relationships may now be identified and completed. In this case there are only two aspects to consider as this model "displays" no modal hierarchy, these two aspects are: the mappings between the bands; and the mappings between the activities in the lower bands and the events in the higher ones. The first of these is defined by the granularity mappings presented earlier in Table 6.15, the second is discussed in the following section.

Event ⟶ Activity mappings

As with the previous models there are a considerable number of mapping between activities and events across the bands, and again only a subset of those possible are identified and developed here.

Taking a top-down approach and considering first the events in the TACTICAL band, the first event to consider is that of *Setup*; this is mapped to the `Setup` activity in the RATIONAL band. Below this level, in the COGNITIVE band, there are finer grained activities for each of the fundamental acts of the setup process, namely: `Enter-site-data`, `Enter-safe-lanes` and `Load-terrain-maps`. These, as noted earlier, occur in sequence (given in equation 6.193) and there is a mapping between these activities and the `Setup`

activity in the RATIONAL band. However, this mapping is not a simple one-to-one, event-to-activity mapping and is rather difficult to define, but there is certainly a mapping between the start of the `Setup` activity and the start of the `Enter-site-data` activity, and another between the end of the `Setup` activity and the end of the `Load-terrain-maps` activity. The `Enter-safe-lanes` activity does not actually map directly to any event in the RATIONAL band, it relationship to the `Setup` activity is given by its precedence relationships with the `Enter-site-data` and `Load-terrain-maps` activities. Finally for the system setup operation, once the data has been entered the `Initial-setup` activity in the LCP-BLOCKS band completed the process, hence there is a mapping between this activity and the end of the `Load-terrain-maps` activity and the COGNITIVE band. The above mappings are more formally defined thus:

$$\textit{Setup} \longrightarrow \texttt{Setup} \tag{6.221}$$

$$\uparrow\texttt{Setup} \longrightarrow \texttt{Enter-site-data} \tag{6.222}$$

$$\downarrow\texttt{Setup} \longrightarrow \texttt{Load-terrain-maps} \tag{6.223}$$

$$\downarrow\texttt{Load-terrain-maps} \longrightarrow \texttt{Initial-setup} \tag{6.224}$$

The operator shift is fundamental to how the system is used; in the TACTICAL band there is the `Shift` activity which serves to define the *Start-shift* and *End-shift* events in the RATIONAL band. These events in turn are reflected in the `Sign-on` and `Sign-off` activities in the COGNITIVE band; more formally these mappings are:

$$\uparrow\texttt{Shift} \longrightarrow \textit{Start-shift} \tag{6.225}$$

$$\downarrow\texttt{Shift} \longrightarrow \textit{End-shift} \tag{6.226}$$

$$\textit{Start-shift} \longrightarrow \texttt{Sign-on} \tag{6.227}$$

$$\textit{End-shift} \longrightarrow \texttt{Sign-off} \tag{6.228}$$

In the TACTICAL band the creation of the mission plans is identified as the event *Plan-routes*, this maps directly onto the `Plan-route` activity in the RATIONAL band. This `Plan-route` activity spans a number of activities in the COGNITIVE band, these are: `Start-plan`, `Enter-waypoint`, and `Complete-plan`. Thus there is a mapping between the start of `Plan-route` and the `Start-plan` activity, and between the end of `Plan-route` and the `Complete-plan` activity; formally these relationships are:

$$\textit{Plan-routes} \longrightarrow \texttt{Plan-route} \tag{6.229}$$

$$\uparrow\texttt{Plan-route} \longrightarrow \texttt{Start-plan} \tag{6.230}$$

$$\downarrow\texttt{Plan-route} \longrightarrow \texttt{Complete-plan} \tag{6.231}$$

It should be noted that there is no direct relationship between the Plan-route and Enter-waypoint activities, the relationship here is indirect and is inherent in the precedence relationships between Enter-waypoint, and the Start-plan and Complete-plan activities in the COGNITIVE band. Finally, from the perspective of the planning related activities, in the LCP-BLOCKS band there are the Initialise-plan and Finalise-plan activities which map to the ends of the Start-plan and Complete-plan activities respectively; and the Calculate-course activity maps to the end of the Enter-waypoint activity. Formally these mappings are:

$$\downarrow\texttt{Start-plan} \longrightarrow \texttt{Initialise-plan} \tag{6.232}$$

$$\downarrow\texttt{Complete-plan} \longrightarrow \texttt{Finalise-plan} \tag{6.233}$$

$$\downarrow\texttt{Enter-waypoint} \longrightarrow \texttt{Calculate-course} \tag{6.234}$$

In the TACTICAL band a single mission is denoted by two events, it starts with *Commence-mission*, and completes with *Complete-mission*; the former maps to the Prepare activity, and the latter to the *Confirm* activity, both of which are located in the RATIONAL band. After the mission is completed there is the task of analysing the results, this is denoted by the *Assess-engagement* event in the RATIONAL band, which in turn maps onto the Assess-results activity in the RATIONAL band.

There are a number of simple mappings between the RATIONAL and COGNITIVE bands, these include those between:

- The *Analyse-BIT-results* and the Analyse-BIT-results activity;
- The *Launch* event and the Initiate-launch activity;
- The *Target-selected* event and the Select-target activity;
- The *Abort* event and the *Abort-mission* activity; and
- The *Confirm* event and the *Engage-target* activity.

More formally these are defined:

$$\textit{Analyse-BIT-results} \longrightarrow \texttt{Analyse-BIT-results} \tag{6.235}$$

$$\textit{Launch} \longrightarrow \texttt{Initiate-launch} \tag{6.236}$$

$$\textit{Target-selected} \longrightarrow \texttt{Select-target} \tag{6.237}$$

$$\textit{Abort} \longrightarrow \textit{Abort-mission} \tag{6.238}$$

$$\textit{Confirm} \longrightarrow \textit{Engage-target} \tag{6.239}$$

Finally there is a simple mapping between the *Picture-update* event in the COGNITIVE band, and the `Update-picture` in the LCP-BLOCKS band, hence:

$$\textit{Picture-update} \longrightarrow \texttt{Update-picture} \tag{6.240}$$

6.7 Final overall system-of-systems model

In developing the process for creating timebands models, described in Chapter 5, it was recognised that the development of the system-of-systems model would proceed in two phases: first the creation of an initial model, which is independent of those of the constituent components; and secondly a "refinement" step, which occurs after the development of the component models, to reflect the necessary aspects of the dynamics of those systems. With the completion of the component models for the payload delivery vehicle and the local command post the focus can now return to the system-of-systems model for this second refinement step.

The original system-of-systems model was incomplete in some areas, the intent of this step is to "complete" the model changing its characteristics to better align it with the models of the components where appropriate. There are two aspects to this alignment:

- The bands —their characteristics, mappings and orderings;
- The artefacts — including the identification of those that only "emerge" in the system-of-systems context, i.e. are not directly apparent in either of the component models .

Each of these aspects is considered in the sections below, starting with the relationships between the bands of the different models.

6.7.1 Inter-model band relationships

Figure 6.30 gives a comparison of the relative positions and scales, albeit rather qualitatively, of the bands defined for the three models. These three models represent a starting point, there is clearly some compatibility between the bands in the different models, but this is certainly not total; the question is: how can these be made cohesive, or perhaps more correctly how can the system-of-systems model bridge the "gaps" between the component models to provide a cohesive structure?

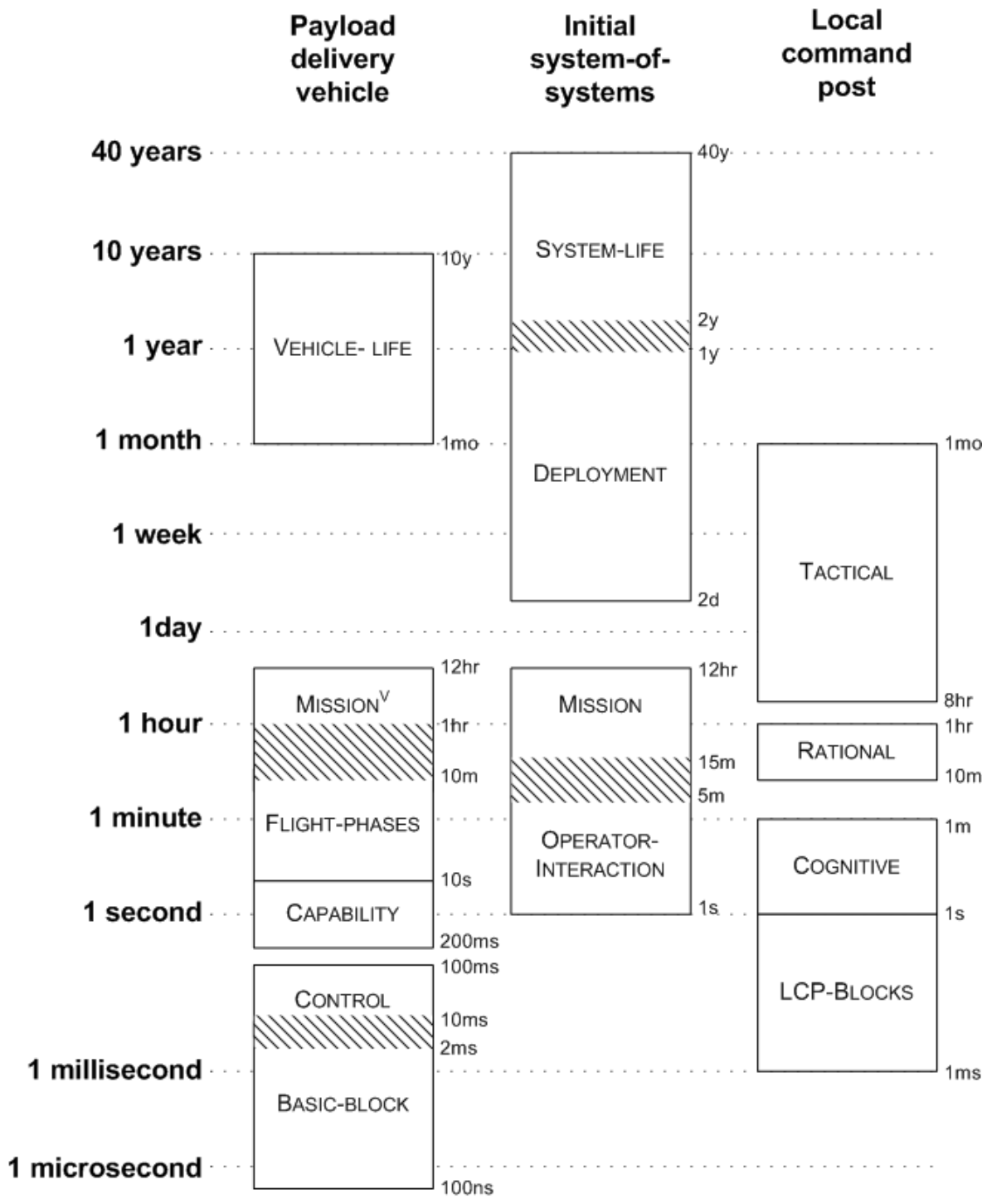


Figure 6.30: A comparison of the bands across the three models

There are three approaches to answering this; these are:

- Maximise compatibility between the bands, i.e. make every lower band in any model map to all higher bands in all models. In simple or small systems this might be possible, but in larger or more complex ones the approach would not be expected to scale well, with the number of inter-band relationships becoming considerable;
- Construct a new set of bands for the system-of-systems model. This approach would introduce the significant problem of accommodating the existing artefacts and relationships defined for the initial system-of-systems model. If this approach were to be appropriate, then it also poses the additional question: is the creation of an initial system-of-systems model unnecessary?
- Define the necessary mappings between the bands in the separate models making a few minor "adjustments" to the characteristics of the bands and their contained artefacts. Ideally this should be restricted to the system-of-systems model as the application of such "adjustments" to the component models could lead to distortions that ultimately reduce the usefulness of those models.

The latter approach seems to be the most appropriate generally, and is certainly the case for the models presented here where, as can be seen in Figure 6.30, there is already a significant degree of compatibility.

Before considering the specific changes that are required to bring these models together it is worth considering what should be done in the case of bands that cannot be mapped to one another without significant changes. The first point to consider is whether or not there is actually any need for there to be a mapping between the bands. Where the band in a component model is only defining behaviours internal to that model, i.e. it does not "export" any events that feature in other models, then there clearly is no need for that band to map to any other in a separate model, indeed it would be incorrect to have such a mapping. This circumstance is most likely to be apparent in the lower implementation related bands.

Where the there is a need to have a mapping between the bands that are not originally closely aligned there are a set of constraints that define what mappings are possible, with the possibilities being determined by the relationships between the granularities and precisions of the bands, and the maximum durations of the individual activities. There are three possible mappings between the artefacts in the different bands, these are: activity to activity; event to event; and activity to event. Assuming there are two bands $B1$ and $B2$,

where $B1$ has the coarser granularity and precision, the following mapping possibilities exist:

- All activities in $B1$ can be described as activities in $B2$;
- Any activities in $B2$, where the product of its duration and the granularity of $B2$ is greater then the granularity of $B1$, can be described as an activity in $B1$;
- Events in $B2$ can be described as events in $B1$; and finally
- Any activity in $B2$, where the product of its duration and the granularity of $B2$ is less than the precision of $B1$, can be described as an event in $B1$

Expressed more formally these mapping possibilities are:

$$G^{B1} \geq G^{B2} \Rightarrow \mathcal{A}^{B2} \rightarrow \mathcal{A}^{B1} \tag{6.241}$$

$$(\#\mathcal{A}^{B2} \times G^{B2}) \geq G^{B1} \Rightarrow \mathcal{A}^{B2} \rightarrow \mathcal{A}^{B1} \tag{6.242}$$

$$\rho^{B1} \geq \rho^{B2} \Rightarrow \mathcal{E}^{B2} \rightarrow \mathcal{E}^{B1} \tag{6.243}$$

$$(\#\mathcal{A}^{B2} \times G^{B2}) \geq \rho^{B1} \Rightarrow \mathcal{A}^{B2} \rightarrow \mathcal{E}^{B1} \tag{6.244}$$

It should be noted that these simply define mappings which are possible and are not an attempt to define a general set of mappings which must always be developed — however outside of these constraints no direct mappings are possible.

Returning to consider the specifics of the models of this case study there are clearly, as shown in Figure 6.30, two instances where there is a significant degree of compatibility (or near compatibility) across two or three bands.

The first instance where there is such a degree of compatibility is that of the MISSION^{V}, the MISSION and, perhaps to a lesser extent, the RATIONAL bands. The MISSION and MISSION^{V} bands only differ in terms of the lower bound (granularity) and the precision, both of which are slightly lower in the case of band in the system-of-systems model. Given the MISSION band does not define any activities that depend on this lower bound it is quite appropriate to redefine this band to have the same characteristics as the MISSION^{V} band; the fact the precision becomes coarser ensures that all events in the MISSION band remain events. The mapping between these bands and the RATIONAL band is also relatively straightforward, with the changes made to the characteristics of the MISSION band defined above all three bands share the same values for granularity and precision, the only difference now being the more limited range of the RATIONAL band. As such it is possible for any activity or event defined in any of these bands to map to any other like

artefact so long as the range of the RATIONAL band is not exceeded (though this could be extended to some extent if required with no significant impact).

The second instance where again there is a significant degree of compatibility is that of the COGNITIVE, the OPERATOR-INTERACTION and, again to a lesser extent, the FLIGHT-PHASES bands. It is perhaps not surprising that the COGNITIVE and OPERATOR-INTERACTION bands are quite strongly aligned, the only difference being the upper bounds of the bands, the reason for this is that the OPERATOR-INTERACTION band also defines the activities captured in the lower end of the RATIONAL band. However this introduces a problem in that nominally the same activity is defined in two places with different granularities and precisions, for example the `Plan-route` is defined in both bands. The question is: which is "correct"? Certainly from Newell's work the route planning, a deliberate act, would be placed in the RATIONAL band, however from the system-of-systems viewpoint it is certainly something in a more specific, to this system, operator time frame — defined here by the OPERATOR-INTERACTION band.

When the mapping between the COGNITIVE and OPERATOR-INTERACTION bands, and the FLIGHT-PHASES band is considered the situation is much less clear. The granularity and precision of the FLIGHT-PHASES band are much coarser that those of the other two bands, with the precision of the FLIGHT-PHASES band, one second, being the same as the granularity of the other two bands; this results in the need to map very short (single tick) activities in the other bands to events in the FLIGHT-PHASES band, and the inability to directly map other short activities, those with durations in the range $2 \ldots 9$ seconds to anything in the FLIGHT-PHASES band, these would have to be mapped to relatively long activities in the CAPABILITY band.

Table 6.20 gives the redefined band characteristics for the system-of-systems model in the light of the changes suggested above, and provides an appropriate precision for the lowest band; these are minor changes that have no impact on the content of the band other than a simple rescaling of durations and persistence values. Having redefined the band characteristics the inter-band granularity mappings for this model may now be defined, these are given in Table 6.21.

Returning to reflect more generally to the relationships between the bands shown in Figure 6.30, there are two other aspects that need to be considered: the "gaps" between bands; and the regions where bands overlap. The gaps between bands indicate that there are no "timings of interest" that have a duration that falls in these gaps, or, put more strongly, it is not permissible for any activity to have a duration that is in the range of any of these gaps. Whilst such a position might seem to be quite restrictive this has to be viewed in

Band	Upper bound/Scale	Granularity	Precision
SYSTEM-LIFE	40 years	1 year	1 month
DEPLOYMENT	2 years	2 days	4 hours
MISSION	12 hours	10 minutes	60 seconds
OPERATOR-INTERACTION	15 minutes	1 second	100 ms

Table 6.20: Revised band characteristics for the system-of-systems model

From	To		
	SYSTEM-LIFE	DEPLOYMENT	MISSION
DEPLOYMENT	182 .. 183	–	–
MISSION	52416 .. 52704	288	–
OPERATOR-INTERACTION	9.06×10^9 .. 9.11×10^9	1.728×10^5	600

Table 6.21: Overall system inter-band granularity mappings

the context of a system with a limited range of operations and a relatively small finite set of durations.

The regions of overlap, by contrast, indicate areas where it is possible to define activities in either of two contexts; whilst this could be viewed as a degree of ambiguity (often viewed as being undesirable in some way), it could also be viewed more positively as a less deterministic solution, affording more freedom in the actual implementation of the system — generally considered to be good attribute. To a large extent which of these two applies is dependent on the context: in the case of a requirement it would be seen as being non-deterministic, whilst in the case of an implementation it would be viewed as an ambiguity.

Having addressed the issue of how the bands of the different models may be interrelated and having refined the characteristics of the bands for the system-of-systems model, the issue of how the artefacts in the different models are related may now be considered.

6.7.2 Inter-model activity and event relationships

In considering the issues relating to the mappings between the activities and events in the component models, and those in the system-of-systems model it is clear that there are only a limited number of sources, points of origin (band or artefact) or reasons for existence, for all of the artefacts in the system-of-systems model. These sources depend to some degree on whether the artefact is an activity or an event, and are:

- The environment — only events;

- From the system-of-systems context itself, these are effectively "self-defined" — activities only, although these may be reflected as events in higher bands;
- Directly from an artefact defined in a particular component model — either activities or events depending on the granularity and precision mappings;
- Indirectly from a number of artefacts from different component models, the "emergent" case — these may be events or activities in any band, clearly any activity in a lower band may also appear as an event in any higher band.

Table 6.22 below identifies the source of all artefacts present in the system-of-systems model against the above four possibilities, identifying the band of the source artefact in the case of the "direct" category; the table is ordered in accordance with the bands of the system-of-systems model.

Artefact	**Type**	**Source**			
		Self-defined	**Environment**	**Direct**	**Emergent**
SYSTEM-LIFE					
Mid-life-update	Event		✔		
End-of-development	Event		✔		
Deploy	Event		✔		
Disposal	Event		✔		
`In-service`	Activity	✔			
DEPLOYMENT					
Start-missions	Event		✔		
Complete-missions	Event		✔		
Configure	Event			TACTICAL	
Mission-received	Event		✔		
Conduct-mission	Event				✔
`Deploy-system`	Activity	✔			
MISSION					
Launch	Event				✔
Abort	Event			RATIONAL	
Timeout	Event			MISSION^{V}	
Engage	Event				✔
Re-route	Event				✔
Route-end	Event			MISSION^{V}	
Prelaunch-tests	Event			COGNITIVE	
Load-mission	Event				✔
Mission-received	Event		✔		

continued on next page

continued from previous page

Artefact	Type	Source			
		Self-defined	Environment	Direct	Emergent
Plan-available	Event			RATIONAL	
End-of-mission	Event				✔
`Configure-vehicle`	Activity			MISSION^{V}	
`Monitor-vehicle`	Activity			MISSION^{V}	
OPERATOR-INTERACTION					
Mission-received	Event		✔		
Image-received	Event				✔
Fire	Event			COGNITIVE	
Vehicle-ready	Event		✔		
Vehicle-status	Event			CAPABILITY	
Load-mission	Event			COGNITIVE	
Send-update	Event				✔
Engage-target	Event			COGNITIVE	
Abort-mission	Event			COGNITIVE	
Final-commit	Event			COGNITIVE	
`Plan-route`	Activity			RATIONAL	
`Assess-target`	Activity			RATIONAL	
`Assess-vehicle`	Activity			COGNITIVE	
`Prelaunch-runup`	Activity			RATIONAL	
`Replan`	Activity			RATIONAL	

Table 6.22: Origins of artefacts in the overall model

Perhaps not surprisingly there are only a few activities that are defined uniquely by the system-of-systems model and have no relationship with the components. These, of course, should only exist for behaviours that relate solely to this particular use of this set of components; the two instances identified in the table above clearly meet this criterion. Similarly the number of external events are also quite limited. The artefacts in both these categories are predominately to be found in the higher bands.

In the case of the artefacts that are directly related to ones in the other models, the mappings are all sound with respect to the precision and granularity requirements, with no further refinement being needed to accommodate the mappings. It is interesting to note that, given the limited lower band of the system-of-systems model, these artefacts originate from quite a wide cross-section of the component model bands. Whilst in some cases the mappings can be traced back directly to a single artefact in a particular band of

a given model, in other cases the mapping is more complex with the system-of-systems artefact being mapped to a number of artefacts in the component band. Considering the *Prelaunch-tests* event; this may be mapped to a period spanning the *Initiate-BIT* event and the end of the `Analyse-BIT-results` activity in the COGNITIVE band. The COGNITIVE band is the lowest level at which the *Prelaunch-tests* event may be expressed as an "activity"; in the next higher band, RATIONAL, the prelaunch checks are denoted by two events *Initiate-BIT* and *Analyse-BIT-results*, however it is not possible to define their separation in the granularity of that band — they do nonetheless exhibit a precedence relationship, which was given in equation 6.176.

The question arising from this "disjoint" mapping is: is it necessary to define an activity in the COGNITIVE band to "describe" the overall prelaunch tests process? The need to create such an activity appears to be somewhat superfluous, and possibly misleading in that an activity would be defined for a process that is disjoint with respect to the local command post model — the majority of the duration of the pre-launch tests process is to be found in the payload delivery vehicle model. This need, or otherwise, to create such "placeholder" activities will be considered further as part of the evaluation of the model in the next chapter.

The final group of mappings to consider are those described as "emergent", and this is perhaps the most interesting group. Some of the relationships forming these artefacts are quite complex, two of the simpler ones are described and considered in detail below.

The *Engage* event relates to the process of the operator deciding to engage a particular target and the vehicle responding accordingly, which in this case means it enters the **Engaging** mode. This process spans a number of models with relies on a number of relationships both within and between bands; however for the process to be represented by an event in the MISSION band is necessary that the elapsed time from the beginning to the end of the process is less than the precision of that band, i.e. 60 seconds. In more detail the process is: the operators enters a command, which is then validated, the command is transmitted to the vehicle where it is in turn decoded; the content of the message is then used to cause a mode transition.

A second example is that of the *Image-received* event in the OPERATOR-INTERACTION band. In this case the process starts in the vehicle where images are collected periodically; these are suitably "reduced" to make then suitable for transmission to the command post. When received the information is used to update the overall tactical picture and this is then presented to the operator. Again the duration from the image being captured to it being presented to the operator must be less that the precision of the band, in this case 100 milliseconds.

Both the cases above expose a problem relating to the incompleteness of the modelling: in both cases the data has to be transmitted between the two components but the characteristics of the transmission have not been modelled. In the former case where the precision is very coarse this is probably not an issue, however in the latter the position is a little more complicated. In this case the whole process must be complete within 100 ms, once the durations for the data reduction, message formation, message extraction and picture update have been allowed for the reminder sets a constraint on the maximum transmission time. In this case that would amount to something in the range $20..30$ ms, and given the capability of the data link this might be considered to be a rather demanding requirement. If this cannot be reliably achieved then the model would have to be changed, two approaches to this are possible: change the definition of the band to relax the precision to allow for the longer transmission times; or, reconsider the "meaning" of the event.

The first approach is somewhat draconian as it impacts on all artefacts in the band, therefore the second approach appears to be more appropriate. So far as the operator is concerned the important issue is having an adequate update rate to allow target assessment, the fact that there might be slightly more latency in the images is less of an issue; by redefining the event to map to only the `Update-picture` activity the precision constraint will be met, however it is also important to ensure that the update rate is also achieved and this is a detail that is not currently captured in these models (or simply by the timebands framework more generally?).

The system-of-systems model has exposed a fundamental of issue with regard to the completeness of the modelling of the components; missing details such as the transmission delays in the above example makes it difficult to develop a complete and consistent model — this issue will be considered further in the next chapter as part of the evaluation of this model.

For the properties that can be checked it is clear from the above discussion that tool support is vital; again this issue will be returned to in the next chapter.

6.8 Summary

This chapter has presented some of the results of a case study in which the timebands framework was applied to a system of some significance for the first time. This has resulted in the creation of three quite different models: two for system components, and one for the overall system-of systems; some simple metrics for these models are given in Table 6.23, in excess of 200 precedence and cross-band mapping relationships were also defined.

Entity	Model			Totals
	Payload delivery vehicle	Local command post	System-of-systems	
Bands	6	4	4	14
Events	63	41	25	129
Activities	31	34	8	73
Modes	16	–	5	21

Table 6.23: Simple metrics for the models

In addition to the timebands framework defined by Burns *et al.* [11], the extension to accommodate modes defined in Chapter 4, and the process for creating the models defined in the previous chapter have both been used in the development of this case study. The system modelled has a wide range of characteristics that have allowed the framework to be explored quite extensively, and generally the models have provided a number of interesting insights. Perhaps these are most apparent in the consideration of the final system-of-systems model where artefacts have "emerged" from the interactions between the artefacts located in the component models. The final system-of-systems model has also exposed a significant issue regarding the completeness of the model.

There are some initial general observations that can be made regarding the process of creating the models; these are:

- The process in general has been successful in so far that the models were created and it was possible to relate the two components through the system-of-systems model;
- It seem likely that for a given domain there will be a "standard" set of bands; this seems to be especially true for systems, and aspects of systems, where there is a high degree of human involvement, and is also expected to be true in the lower technology-related bands where the same form of implementations are used across different components;
- In the case of the system-of-systems model there is "tension" between the characteristics inherent in the components and those desired of the overall system — the system-of-systems model has to be able to accommodate this "tension", bridging the "gap" between the components.

The next chapter will consider both specific aspects of the timebands framework, and more general issues regarding the process of creating timebands models, in the light of this case study.

Chapter 7

Evaluation, Assessment and Conclusions

The case study undertaken in the previous chapter has resulted in the development of a suite of three timebands models: two for component systems, and one for the overall system-of-systems. In this chapter these models are used in support of an in-depth, evidence-based evaluation and assessment of the timebands framework in terms of two fundamental criteria: the strength of its technical content; and its application.

The construction of the models in the case study is the first application of the timebands framework to a system of significant size and diversity, this diversity has proved to be useful in allowing a relatively broad evaluation of the process developed earlier in Chapter 5. The key concerns to emerge through the case study were:

- The importance of precision, and its relationship with granularity;
- The dominance of events over activities;
- Some difficulties with precedence and inter-band mappings;
- The practicalities of constructing timebands models;
- The influence of the component models on the refinement of the system-of-systems model.

There are two contrasting approaches that could be taken towards the evaluation and assessment of the models and framework. In the first, each model would be considered in its own right and conclusions regarding the framework and the process could be

drawn from these evaluations and assessments — a model-centric approach. The alternative approach is to take the key concerns regarding the framework that emerged in the construction of the models, and use the models to illuminate their evaluation and assessment — a framework-centric approach. Here this latter approach is taken, as in the model-centric approach the deeper, more diverse, and perhaps more important issues regarding the framework may not be so readily apparent. The key concerns to be considered, and the questions they raise, being:

1. **Bands and their characteristics:** How are the characteristics related and what are the "effects" of these relationships?

2. **Activities and events:** Is the original model correct or suitable, how can the inevitable variability in key temporal properties such as durations and periods be accommodated or captured?

3. **Precedence and inter-band mappings:** Are the definitions of precedence and inter-band mappings consistent, and how do they relate to the fundamental characteristics of the bands?

4. **System modes:** Is the concept useful in the timebands context, and how is it affected by the consideration of the two previous questions?

5. **Model creation process:** Did the process work, and what issues did its use expose?

6. **Tool support:** What kind of tool support is required to make the application of the timebands framework a practical proposition?

The evaluation and assessment is focused on the fundamental goal of this research stated in Section 1.2:

> *... to evaluate the timebands framework and identify how (or whether) the concept of timebands may be used to provide a "better" architectural description for embedded real-time systems.*

Each of these above concerns and their related questions are addressed in the "Specific issues" section below; this is followed by a brief consideration, in more general terms, of both certain aspects of the framework, and the modelling process. Following a summary of the evaluation and assessment, in which a number of open issues are identified, the chapter concludes with an overall summary of the thesis, some final conclusions with respect to the central proposition, and recommendations for future research.

7.1 Specific issues

The key issues are, unsurprisingly, not entirely independent of one another; however they are considered in the sections below in a largely independent manner — the issues being ordered to minimise the impact of any dependence. The issues relating to the original framework are considered first, followed by those relating to aspects added in this thesis, finally issues relating to the wider application of the framework are considered. Each section closes with a statement of the conclusions of that aspect of the evaluation and assessment.

7.1.1 The bands and their characteristics

Being the fundamental notion on which the timebands framework is founded the bands are the obvious place to start any evaluation and assessment. There are four fundamental aspects to the bands, which are all linked in some way; these are:

- The precision;
- The granularity;
- The scale of the band;
- The relationships between bands — granularity mappings, and the gaps and overlaps between bands.

Whilst the granularity and precision are well understood in their own right their relationship within any given band has not been given much attention to-date. In the case study a ratio of 10:1 between precision and granularity was used for most bands, the lowest ratio being 10:1 for a number of bands across the models, and the highest being $\approx$ 30:1 for the VEHICLE-LIFE band of the payload delivery vehicle model. There was little to underpin the choice of a ratio of 10:1, save for the fact that it appears to be a common value used in measurement and control across a variety of engineering disciplines. The US military standard for test methods for micro circuits, MIL-STD-883G [30], requires a Test Accuracy Ratio (TAR), essentially the ratio of the measurement being made to the accuracy of the measuring equipment, of between 10:1 for routine processes and 4:1 for special processes; thus as a starting point the ratio of 10:1 does not seem unreasonable. The question to be answered here is: what constitutes an appropriate ratio?

Perhaps the best point from which to start answering this question is to consider the impact of this ratio. This ratio fundamentally defines the limit of the discrimination that can be achieved between activities and events within a band; any artefact with a

duration between the granularity and the precision cannot be described directly within that band, and some other derived aspect would need to be considered, e.g. some critical point or event within the associated activity. This results in a tension: on the one hand a small ratio will allow the straightforward inclusion of more artefacts in the band, but the events and activities will not be particularly well discriminated; on the other hand a large ratio restricts, perhaps considerably, the artefacts that can be accommodated directly in the band. Before considering further what constitutes an appropriate ratio between the quantities, it is useful to consider both their origins and wider relationships.

The precision is largely defined by entities that are not originally defined in that band; events originating from an activity in another band, or something in the environment are the "drivers" for the precision. The granularity however is very much a property that is internal to the band, it being the measure of time within that band, and only becoming visible in a wider context when the inter-band granularity mappings are considered. The linkage between the two is usually to be found when considering the mappings between bands; for an activity in a lower band to map onto an event in a higher band the duration of the activity must be less than the precision of the higher band; thus the granularity of its band must also be less than the precision of the band describing the event, possibly considerably so where the activity mapped to the event has a duration of a very large number of ticks.

It is clearly difficult to generalise what is, and what is not, a "correct" ratio, the trade-offs between the various aspects considered above are complex and will certainly depend on the particular attributes of any given model. A nominal value of 10:1 thus seems to be acceptable at this point; however with wider use of the framework it should be possible to refine this value.

The concept of scale, or an upper bound on the durations of activities defined in a given band, is one that has been introduced in this thesis. The notion of such an upper bound follows from Newell's approach to bands [70] where the top of one of his bands is limited by the base of the next. The concept of an explicit scale introduced in this thesis does not restrict the timebands framework to the degree that would be the case if Newell's approach were to be taken in that it does not prohibit bands from overlapping; equally it does not leave the question regarding the maximum duration of an activity completely unbounded as was the case with the original framework. Its inclusion encourages a more precise definition of each band, and serves to highlight the overlaps and gaps within the banded structure, as shown in Figure 6.30 in previous chapter. In the figure an example of a gap is that between the VEHICLE-LIFE and MISSION^{V} bands of the Payload Delivery

Model, and an overlap that between the MISSION^{V} and FLIGHT-PHASES bands in the same model.

Formally the scale of the band may be defined as a positive number of ticks in the granularity of that band:

$$scale : \mathcal{B} \rightarrow \mathbb{N}_1$$

A peripheral advantage of including the concept of scale is that in some cases it makes the determination as to whether two bands are totally ordered trivially simple; this condition occurs when the scale of one band is a time of less than the precision of a higher band, hence all activities in the lower band may be defined by events in the higher one and vice versa — however if the condition does not hold then it is not possible to state that the bands are totally ordered without explicit consideration of all of the mappings between the bands. This implies, with the introduction of scale, that there are two notions of total ordering; the first is that in the original definition of the framework where the bands were totally ordered if the precision of the higher band was greater than the granularity of the lower band — this could be described as *band total ordering*. The second is the one described above where precision of the higher band is greater than the scale of the lower band — this could be described as *artefact total ordering*. These are more formally defined thus:

$$\rho^H > G^L \tag{7.1}$$

$$\rho^H > scale(L) \tag{7.2}$$

Following on from the concept of scale there is the issue of how the bands are related to one another to be considered. The original framework does not provide for any formal association between a band and its defining characteristics of granularity and precision, or even between these properties. Between the bands the granularity mapping relationship, and the band ordering are both formally defined, but there is no association between the precision of one band and the granularity of another.[1] The lack of formality in some of these definitions has resulted in the omission of constraints that should have been applied across bands; for example the need for the activity associated with an event through the *Mapping* relationship to have a duration of less than the precision of the band of the event cannot be captured formally. This is a significant weakness of the framework, one that complicates the construction of models. One approach to its solution might be to capture

[1]However, the fact that the precision of one band must be measurable in the granularity of another is noted informally.

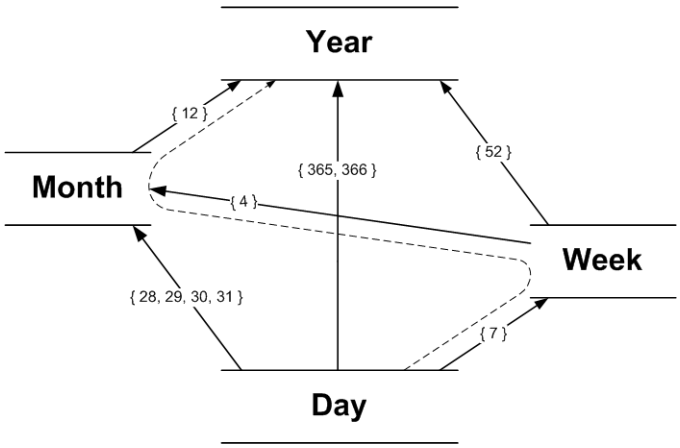


Figure 7.1: Granularity relationships between Years, Months, Weeks and Days

the ratio between the precision and the granularity as another of the characteristics of the band; this, together with the inter-band granularity mappings would allow the precision of one band to be readily related to the granularity of another.

An interesting observation with regard to the inter-band granularity mappings is the possibility of being able to identify multiple "paths" when applying granularity mappings transitively. By way of an example consider a system consisting of four bands, day, week, month and year; it is possible to define six granularity mappings, these are:

$$DayWeek == \{7\}$$

$$DayMonth == \{28, 29, 30, 31\}$$

$$DayYear == \{365, 366\}$$

$$WeekMonth == \{4\}$$

$$WeekYear == \{52\}$$

$$MonthYear == \{12\}$$

Inappropriate use of these mappings can give unexpected results; for instance if the dashed "path" in Figure 7.1 is followed then a year is apparently 336 days — certainly not the expected answer. The real issues here is to ensure that the mapping are used appropriately and ensure that the necessary granularity mappings are provided.

This example points to another issue, that of inadequate band separation; whilst the *DayYear* mapping is close to that between Newell's bands (10^3) the *WeekMonth* mapping is very restricted. Perhaps, as with the ratios between precision and granularity, there should be a minimum ratio for inter-band granularity mappings. In the models produced for the case study the lowest ratio was 48:1, between the RATIONAL and TACTICAL bands

in the local command post model. This value does not seem to be inappropriate for the circumstances of that model, but equally it is not possible to use this to infer something more general regarding the wider acceptability of this ratio; more examples of models need to be produced in order that the acceptable lower bound of this value can be identified.

Conclusions of evaluation

In general and individually the concepts associated with the bands are well, but informally defined; the following detailed conclusions may be drawn:

1. The lack of a definition of the relationship between granularity and precision is a major weakness — formal definition of this relationship would undoubtedly strengthen the framework as it would allow inter-band relationships to be rigorously verified.
2. The correct use of scale can help to prevent artefacts from being placed in too low a band, and this concept should be included as a property of the band.

In relation to the last point, it is currently unclear as to whether the imposition of a scale will be too restrictive when a wider class of systems are modelled, but its inclusion undoubtedly helps to clarify the overall structure by encouraging a deeper consideration of the definition of each band.

7.1.2 Activities and events

The activities and events are, after the bands themselves, the two most important elements of the original timebands framework; if the bands are considered to define the shape of the model then the events and activities are its constituent parts. In the original definition of the framework the event was seen as being simply a specialisation of the activity, one having a duration of zero ticks in its defining band. This simple specialisation however failed to capture the key constraint that restricts the duration of the event to being less than the precision of its band.

In constructing the models for the case study the definition of the event by this simple specialisation has been called into question; there are two main reasons for this and these are:

- The models are dominated by events — especially at the higher levels of abstraction where very few activities are defined, indicating that the event is quite a different concept from that of the activity; it is only in the lower bands do the activities become dominant. A further related issue is that the interactions with the environment

can only be signified using events — to do otherwise would require that the details of these interactions were modelled in terms of the characteristics of some band, making them part of, rather than something external to, the model;

- Whereas the duration of an activity is measured in ticks in the granularity of its band, the duration of an event is, as noted above, required to be no more than the precision of its band. The events and activities differ quite fundamentally in that the duration of an activity has a minimum lower bound, of one tick, but no upper limit, the "duration" of an event contrastingly has no lower bound, but its upper bound must be less than the precision. Again this suggests that events and activities are rather independent concepts.

Another, perhaps more paradoxical, issue with the original definition is that the activity itself relied on events to define its beginning and end, making the activity clearly dependent on the event; however the event is itself also an activity — this appears to confuse the terms unnecessarily.

From a different viewpoint there are two observations, which emerged from the construction of the models in the case study, that further suggest that there are limitations with the current definitions of activities and events, and that their redefinition would be beneficial; these are:

- There is a need to allow activities to have a range of durations. In most circumstances it is simply impractical to define a single value for the duration of an activity that addresses all eventualities. In the case study only the highest level activities, for example the `In-service` activity in the SYSTEM-LIFE band of the system-of-systems model, could be defined in this way; generally this tends to be the case in the higher bands of the models where the time was a requirement[2] rather than the result of some implementation decision. The `Plan-route` activity in the RATIONAL band of the Local Command Post model is an example of this latter case where the duration of the activity driven by an implementation decision.

- There is a need to identify the repetition rate of, or minimum and maximum separations between, occurrences of events; knowledge of these rates and separations are fundamental to ensuring that the events are defined in the appropriate band. It would be inappropriate to place an event with a high repetition rate in a high level band where the granularity was unable to express that rate, or worse still where

[2]Thus it might be considered to be a maximum or minimum value.

the precision was so coarse that individual occurrences of the event could not be discriminated. In some limited cases it is undoubtedly possible that the rate may be captured by the characteristics of the band; generally however the bands are too broad to capture this usefully. This is especially the case for sporadic events that have highly variable separations between occurrences, or equally for that of systems which exhibit several different periodic rates within a single band.

Given all of the above points it appears that redefining the terms for both the event and the activity, removing the dependence of the event on the activity, providing ranges of rates for the events, and changing activity durations to accommodate ranges of values, would be beneficial to the framework. These redefinitions are discussed in turn below, starting with the events.

Redefinition of the event

As was the case for the original framework an event is defined in the context of a single band, the additional defining characteristic of the event provided here a range of times signifying its range of occurrence repetition rates or arrival separations. Formally the definition of the event remains unchanged:

$$band : \mathcal{E} \rightarrow \mathcal{B}$$

There must be some events defined for any band, otherwise the band has no content; hence:

$$events : \mathcal{B} \rightarrow \mathbb{P}_1\, \mathcal{E}$$

$$\forall\, b : \mathcal{B} \bullet events(b) = \{e : \mathcal{E} \mid band(e) = b\}$$

The range of the *period* of, or the limits of the separation between, occurrences are defined in terms of the granularity of its band, with the upper bound being limited to the scale of the band; thus:

$$period : \mathcal{E} \nrightarrow \mathbb{N}_1 \,..\, \mathbb{N}_1$$

$$\forall\, e : \mathcal{E} \mid e \in \operatorname{dom} period \bullet \max(period(e)) \leq scale(band(e))$$

A single value range for the *period* would imply that the jitter in this value was less than the precision of the band; if the jitter were to be between the precision and the granularity then the range would be three adjacent values — the expected, and plus or minus one.

The constraint ensures that if the period is defined then it must be less than the scale of the band.

It is important to note that the definitions of the *period* given above is only a partial function hence not all events have this attribute; the immediate question arising from this is: what is the meaning of having an undefined *period*?

There are three possibilities to consider:

- The event repeats with a separation between the precision and the granularity;
- The event repeats with a separation of no more than the precision;
- The event is so infrequent in the band that it exceeds the scale of the band, thus effectively it does not repeat; such a circumstance might arise where an event originating in the environment has a direct impact on a very low level band (e.g. a rare failure event).

Both of the first two above would be attempting to define something that was strictly outside the context of the band, as values of less than the granularity are not discernible. It therefore appears that the latter is the most appropriate interpretation — though in a sense this could also be seen as being something outside the context of the band in that its *period* is too long to be accommodated by the scale; however there is a need to accommodate singleton occurrences such as birth, or the end of development, and thus the approach can be justified.

Considering events more generally, the possibility of sporadic or aperiodic events also need to be addressed. In this case rather than the single concept of *period* there are those of *minimum inter-arrival* time and *jitter* that need to be considered. In this case the *jitter* can be viewed as an increase in the spacing of events over the *minimum inter-arrival* time. The formal definitions of these two quantities applied to events in the timebands framework are:

$$
\begin{array}{|l}
minInterArrival : \mathcal{E} \pfun \mathbb{N}_1 \\
jitter : \mathcal{E} \pfun \mathbb{N}_1 \\
\hline
\forall\, e : \mathcal{E} \mid e \in \operatorname{dom} minInterArrival \bullet minInterArrival(e) \leq scale(band(e)) \\
\forall\, e : \mathcal{E} \mid e \in \operatorname{dom} jitter \bullet jitter(e) \leq scale(band(e)) \\
\forall\, e : \mathcal{E} \mid e \in \operatorname{dom} minInterArrival \land e \in \operatorname{dom} jitter \bullet \\
\quad minInterArrival(e) + jitter(e) \leq scale(band(e)) \\
\forall\, e : \mathcal{E} \mid e \in \operatorname{dom} period \bullet e \notin \operatorname{dom} jitter \land e \notin \operatorname{dom} minInterArrival
\end{array}
$$

The definitions above capture both the constraint relating to the *minAinterArrival* and *jitter* of an event, individually and in combination, not exceeding the scale of its band, and the exclusivity of *period*, and *minInterArrival* and *jitter* for any particular event.

Redefinition of the activity

In common with the event and the original definition of an activity, an activity is also defined within the context of a single band, the redefinition of the activity here being largely limited to changing the definition of the duration from a single value to a range of values. The formal definition of the activity is repeated below and remains unchanged:

$$band : \mathcal{A} \rightarrow \mathcal{B}$$

With the decoupling of the activity from the event the framework no longer imposes an absolute requirement to define any activities in a band, though in practice this would be unlikely in any band other than possibly the highest. Formally this may be expressed thus:

$$activities : \mathcal{B} \nrightarrow \mathbb{P}\,\mathcal{A}$$

$$\forall\, b : \mathcal{B} \bullet activities(b) = \{a : \mathcal{A} \mid band(a) = b\}$$

The duration (#) is redefined as a positive range, one which excludes the possibility of a zero duration, and has an upper bound constrained to the scale of the band. Unlike the events where the *period* was optional, here the duration must be defined, otherwise the activity would not be an activity in the general meaning of the framework.

$$\# : \mathcal{A} \rightarrow \mathbb{N}_1 \,..\, \mathbb{N}_1$$

$$\forall\, a : \mathcal{A} \bullet \max(\#a) \leq scale(band(a))$$

Another possible change that could be considered would be, as with the event, the provision of a range of occurrence separations; however for the activity this is unnecessary and probably inappropriate. The “occurrences” of an activity should be defined by the occurrences of a corresponding event in a higher band, thus the iteration rate of the activity is defined by that of the higher level event. Whereas in the original framework this might not have been entirely obvious, with the inclusion of the period for the event the position is clarified somewhat, and such an attribute is not required for the activity.

Interdependencies between events and activities in the same band

In the original definition of the framework, there were interdependencies between events and activities. Both the start and end of an activity were defined as being events in the

same band as the activity, furthermore the definition of the within relationship was also based on these start and end events. Each of these definitions is considered below in the light of the changes to those of the event and the activity given above.

The formal definition of the start and end of an activity can be restated given the redefinition of the events above; there are two changes to note:

- A change to identify that the start event exhibits strict precedence over the end event, before this was only weak precedence as an activity could also be a single event, whereas now they must be separate;
- The definition of the duration of an activity composed of two others is changed to state that the duration range of the composed case is set by the sum of the minimum and maximum durations of the component activities.

$$
\begin{array}{|l}
\uparrow_ : \mathcal{A} \rightarrow \mathcal{E} \\
\downarrow_ : \mathcal{A} \rightarrow \mathcal{E} \\
\hline
\forall\, A : \mathcal{A} \bullet band(\uparrow A) = band(A) = band(\downarrow A) \land \uparrow A \prec \downarrow A \\
\forall\, A, B, C : \mathcal{A} \bullet \\
\quad (\uparrow C \equiv \uparrow A \land \downarrow A \equiv \uparrow B \land \downarrow B \equiv \downarrow C \Rightarrow \\
\qquad \#C = (\min(\#A) + \min(\#B)) \,..\, (\max(\#A) + \max(\#B))
\end{array}
$$

The within relationship is also affected by the variability in the duration of the activity, and now only holds if the maximum duration of the contained activity is no more that the minimum duration of the containing activity; the original precedence relationships between the start and end event so the two activities still hold, thus the relationship now is.

$$
\begin{array}{|l}
_ \,\mathsf{within}\, _ : \mathcal{A} \leftrightarrow \mathcal{A} \\
\hline
\forall\, A, B : \mathcal{A} \bullet \\
\quad (B \,\mathsf{within}\, A \Leftrightarrow \uparrow A \preceq \uparrow B \land \downarrow B \preceq \downarrow A) \land \\
\quad (B \,\mathsf{within}\, A \Rightarrow (\forall\, a : \#A \bullet \neg(\exists\, b : \#B \bullet b > a))
\end{array}
$$

Conclusions of evaluation

The creation of the models has exposed the importance and uniqueness of the events in the framework. The following conclusions regarding the exploitation of this importance can be drawn:

3. The events should be defined as being unique entities in the framework.

4. An optional “period”, “minimum inter-arrival” and “jitter” attributes should be provided for the events, with the upper bound of the range of the period, and the sum of the minimum inter-arrival and the jitter being limited to the scale of its band.

In addition the concept of the activity is clarified by this change and may be further strengthened by:

5. Changing the duration of a activity to be a positive range, the upper bound of this range, as with the period of the event, should be limited to the scale of its band.

These changes have only a minor impact on other aspects of the framework but considerably improves its clarity.

7.1.3 Precedence and inter-band mappings

The precedence relationships between two events are constrained such that the events have to be defined in the same band; events can exhibit strict or weak precedence over each other, or they may be defined as equivalent — indistinguishable in their band. The original definition of the framework quite deliberately avoids any equivalence between time and precedence; hence there is no set of stated “rules” linking precedence to the precision or granularity of the band. Whilst this allows orderings and causal relationships to be stated independent of any notion of the passage of time it does not identify any temporal invariants over precedence, or vice versa.

There are however some such invariants that may be readily defined, and some others that are more problematic. In the first category it can be argued that events separated by at least the granularity of their band *may* exhibit strict precedence; this is only *may* because they could be unrelated. At the other end of the temporal spectrum for events to be equivalent they must be separated by less than the precision of the band — though here again they do not have to be equivalent. Other than these two simple examples the position is less clear and two questions are raised:

- Can strict or even weak precedence be apparent where the separation between the events is less than the precision of the band?
- Can strict or weak precedence be apparent where the separation between the events is less than the granularity of the band, but at least as much as its precision?

The answers to both of these questions comes from a consideration of preservation of precedence that follows from the mapping between events and activities in different

bands. Precedence and inter-band mappings are not unrelated concepts in that precedence present in lower bands must be preserved in higher ones; the reverse however, is not a necessary condition. Between bands where there is the mapping between events and activities there is the property of the preservation of precedence, if event Y weakly precedes event Z, and these are mapped onto activities W and X respectively in a lower band, then preservation demands only that the start of W occurs before the end of X. In the original definition of the framework this inter-band mapping was only defined for the case of weak precedence — the mapping in the case of strict precedence was, unfortunately, omitted and needs to be defined.

Before considering the meaning of strict precedence, and its relationship to the mapping between bands it is worth noting that in the case study there are a number of instances where an event in one band does not map onto an activity in a lower band, but instead maps onto a pair of events. An example of a such a case is that, in the system-of-systems model, of the mapping between the *Conduct-mission* event to the start of the `Configure-vehicle` and the *End-of-mission* event — see Figure 6.7 in the previous chapter. The original definition of *Mapping* did not allow for this circumstance, but the definition may be simply restated to allow for this by redefining it in terms of an event mapping to an ordered pair of events thus:

$$Mapping : \mathbb{P}(\mathcal{E} \rightarrowtail (\mathcal{E} \times \mathcal{E}))$$

$$\begin{aligned} Mapping = \{ m : \mathcal{E} \rightarrowtail (\mathcal{E} \times \mathcal{E}) \mid m \neq \varnothing \land \forall\, r : \mathrm{ran}(m) \bullet first(r) \preceq second(r) \land \\ \exists\, hi, lo : \mathcal{B} \bullet lo \sqsubseteq hi \land m \in events(hi) \rightarrowtail (events(lo), events(lo)) \land \\ (\forall\, E, F : \mathrm{dom}(m) \bullet E \preceq F \Rightarrow first(m(E)) \preceq second(m(F)))\} \end{aligned}$$

An additional constraint is strictly necessary — that stating that the separation between occurrences of the mapped events needs to be less than the precision of the upper band, however this is not possible formally as there is no formal definition of the precision.

This allows a more flexible set of mappings including, in addition to event ⟶ activity, the possibility of mapping any event in a higher band to pairs of arbitrary events in the (same) lower band, the only constraint being that the first event displays at least weak precedence event over the second of the pair in the lower band. This also allows for the case where an event in one band maps to a single event in the lower band. Of course the definition of the mapping of an event to an activity given in the original definition remains valid as the event-pair may be mapped onto the start and end events of the activity, though the redefinition of the activity also needs to be considered.

Returning to consider the case of strict precedence for the mapping of an event to an event-pair there are two possibilities, these are shown in Figure 7.2. In the first instance,

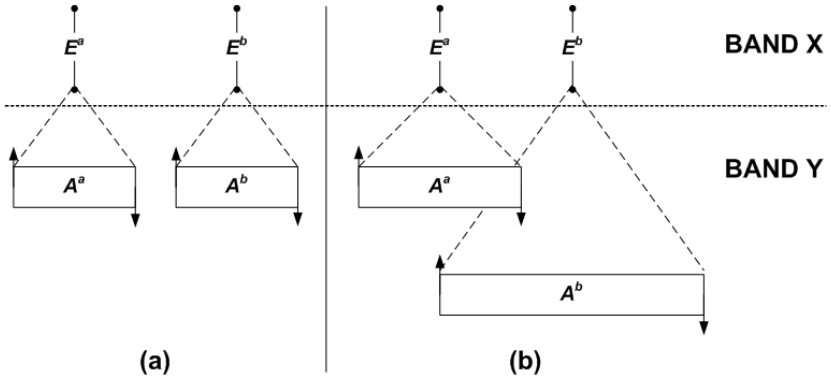


Figure 7.2: Possible schemes for inter-band mappings for strict precedence

given in case (a) in the figure, event E^a displays strict precedence over event E^b, and when mapped to the activities in a lower band the activity A^a, associated with event E^a, completes before the start of activity A^b, mapped to event E^b. In this case there is either strict or weak precedence between the end of A^a and the start of A^b. In the second case, example (b) in the figure, again *Ea* displays strict precedence over E^b, but in this case the activities are overlapped. The start and end of A^a displays either strict or weak precedence over the corresponding points in activity A^b. As strict precedence can indicate causality the overlap show in case (b) appears to be inappropriate, thus example (a) gives the general pattern for strict precedence. This leave one point to be resolved: should the precedence relationship between the end of A^a and the start of A^b be strict or weak?

Considering these two options further: if the precedence is strict then the mapping of these two events onto artefacts in yet lower bands will require that these artefacts also exhibit strict precedence. This is counter to the intent of the framework where precedence is preserved in the transition from lower to higher bands, but not in that from higher to lower bands — defining this as a strict precedence relation, in effect, introduces a degree of over-specification. The alternative of making this precedence weak avoids this "over-specification" problem as in subsequent lower level mappings only the weak precedence rules have to be followed, this reduces the "determinism" and provides more scope at the lower implementation levels. Incorporating this addition for strict precedence into the definition of the *Mapping* relationship is quite simple, with the relationship becoming:

$$
\begin{array}{|l}
\mathit{Mapping} : \mathbb{P}(\mathcal{E} \rightarrowtail\!\!\!

The last line of the definition addresses the strict precedence case for the inter-band mapping, this allows for the implementation to provide a strict ordering without the need for this to be reflected back into the requirement.

Having defined the "rules" for the strict precedence and its mapping down through the bands the next issue to consider is that of the asymmetry in the relationships between bands. As noted earlier the strength of precedence is not preserved in the transition from coarse to finer bands, but it is in the case of transition from finer to coarser bands. Thus if the ordering between requirement events specified in some higher band is to be preserved in some lower band (e.g. there is causality between the two events), then strict precedence has to be applied. If however the events in the higher band are due to activities in lower bands then the precedence stated in the lower bands is naturally preserved between the higher level events.

The *Mapping* relationship is only defined to accommodate the first case and not the second. In the latter case where, for example, two activities exhibit a strict precedence relationship, with one completing simultaneously with the other starting, it is necessary that the corresponding events in the higher band also exhibit this strength of precedence.[3] The inverse mapping, or *Mapped* relationship, is in many ways similar to that of *Mapping* with its definition being:

$$
\begin{array}{|l}
Mapped : \mathbb{P}((\mathcal{E} \times \mathcal{E}) \rightarrowtail \mathcal{E}) \\
\hline
Mapped = \{m : (\mathcal{E} \times \mathcal{E}) \rightarrowtail \mathcal{E} \mid m \neq \varnothing \land \forall\, r : \mathrm{dom}(m) \bullet first(r) \preceq second(r) \land \\
\quad \exists\, hi, lo : \mathcal{B} \bullet lo \sqsubseteq hi \land m \in (events(lo), events(lo)) \rightarrowtail events(hi) \land \\
\quad (\forall\, E, F : \mathrm{dom}(m) \bullet first(E) \preceq second(F) \Rightarrow m(E) \preceq m(F)) \land \\
\quad (\forall\, E, F : \mathrm{dom}(m) \mid E \neq F \bullet second(E) \preceq first(F) \Rightarrow m(E) \prec m(F))\}
\end{array}
$$

As the definition of the strict precedence is always contained within that of weak precedence there in no conflict between the mapping of strict precedence due to its definition including the weaker form, i.e. $A \prec B \Rightarrow A \preceq B$.

Conclusions of evaluation

The concepts of precedence and inter-band mappings play an essential role in the construction of useful models. There are a number of changes that should be made that will further strengthen this role; these are:

6. Removing of the artificial constraint that only allows mappings between bands to be between events and activities, to additionally allow event to event-pair mappings.

[3] The same "requirement" is also true in the case of weak precedence.

7. Defining, formally, the inter-band mapping for the case of strict precedence between events.

8. Defining, formally, the "inverse" inter-band mappings from the low-level event-pairs to the higher level events — the *Mapped* relationship.

9. Inclusion of the constraint that a mapping relationship must honour the precision of the higher band — this requires first that precision is formally defined.

The first of the above follows from the changes to the definition of the activity and event presented in the previous section. Allowing event to event-pair mappings would make it possible to map an event to a pair of events separated by less than the granularity thus a mapping would become a "continuous" relationship that would be less sensitive to changes in the granularity in the lower band.

7.1.4 System modes

The concept of system modes was introduced to the timebands framework as part of this thesis, with an element of its definition, the transitions, being based on the events defined for the original framework. The changes made earlier to the events and the inter-band mappings have a positive impact on the definitions of modes and mode models allowing the hierarchical mapping of modes to be strengthened, as such some changes are required to the definitions surrounding modes. These changes will be considered later but first the utility of the concept as demonstrated through the case study will be considered.

The strongest modal behaviour was noted in the model of the payload delivery vehicle; here two bands, MISSION^{V} and FLIGHT-PHASES, exhibited modal behaviour with the mode models being hierarchical across the two bands. In addition the overall system-of-systems model exhibited similar modal behaviour in its MISSION band to that apparent in the MISSION^{V} band of the vehicle model. The mode models for the vehicle are largely self-contained in that their transitions are largely independent of external events, the only notable exceptions being those of *Power-on* and *Power-off*. Whilst many of the other transitional events ultimately have their origins in either the environment, or another part of the wider system, the actual events used in the definitions of the transitions are ultimately due to activities in the lower bands.[4]

The mapping between modes in a higher band, and mode models in a lower one could only be explored in the payload delivery vehicle model, and this exposed a potential

[4]It should be noted that the same could have been argued for the power cases had the models included aspects such as power conditioning.

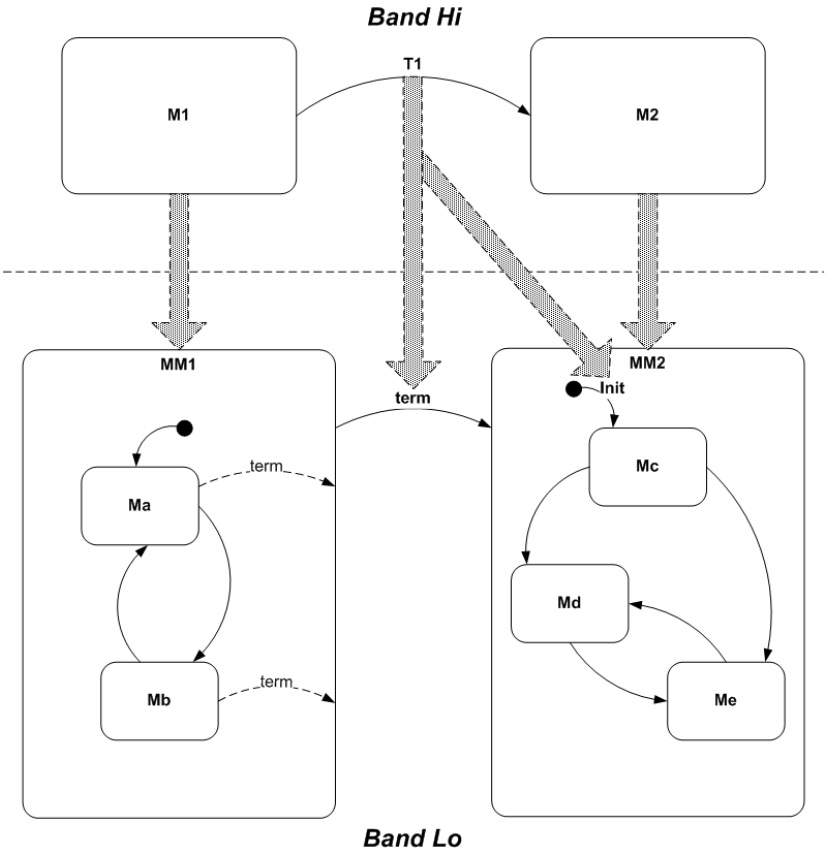


Figure 7.3: Mappings between modes and mode models

limitation with the approach. The modes defined in the higher bands did not all map onto decomposed models in the lower band, the modes **Loiter** and **Aborting** being the cases where this occurred. The problem that arises is that the inter-model transitions can be viewed from the perspective of either:

- The modes and the transition in the higher band; or
- The models in the lower bands where the transition becomes mapped to a *term* transition in first model, and the *init* transition to the entry mode in the second model.

It is clearly important that these two views are consistent.

The approach taken to alleviate this problem in the case study was to define lower level models of these “un-decomposed” modes that consist of a single mode, and defined no inter-mode transitions, simple the *init* and *term* transitions.

An additional weakness in the definition of the mapping between the modes in one band, and the corresponding mode models in another is that of the mapping between the transitions at the higher level and their realisation at the lower level; this mapping is shown in Figure 7.3. Recalling, from equation 4.1, that a transition is not defined as being sim-

ply an event, but consists of two modes and an event, the issue here is one of how to ensure these mappings are also captured consistently. Fundamentally there are three aspect to align: the starting and ending modes, and the transitional event. The modes define the starting and ending mode models, a mode can only ever map to a single model and vice versa, thus these are simple. The transitional event will map at the lower level to two events: a *term* event in the starting mode model, and the event of one of the *init* transitions in the ending model. Thus in the case shown in Figure 7.3 the mappings are:

$$M1 \rightarrow MM1, \qquad M2 \rightarrow MM2, \qquad T1 \rightarrow (term, init)$$

The event relating to the transition in the upper band can be mapped to two events in the lower band: one of the *term* events defined in the source model, and the event associated with the *init* transition defined in the destination model. Using the original definition of the framework, this mapping would have required that there was an activity, bounded by these events, in the lower band. This requirement however poses a conflict, the activity has to have a duration in the lower band, but this is still an inter-mode transition which by its definition, given in Chapter 4, is based on an event. The use of two events to define the transition, whilst not reducing to a single event, allows the duration of the transition to remain "unmeasurable" in the lower band — this is another example where the redefinition of the inter-band mapping discussed earlier would be beneficial.

These changes also help to clarify an issue left unresolved in Chapter 4 regarding the need to decompose a mode model in a higher band to a set of mode models in a single band — this mapping of the transitions makes this necessary, it would not be possible if all the lower-level mode models and the events relating to the term and init transitions were not in the same band.

Formally this mapping of the transaction may be expressed as follows; first the definition of a transition between mode models:

$$\mathcal{T}^{model} : ((\mathcal{E} \times \mathcal{E}) \times \mathcal{M}^{\mathcal{M}}) \times \mathcal{M}^{\mathcal{M}}$$

$$\forall x : T^{model} \bullet$$
$$(band(first(first(first(x)))) =$$
$$band(second(first(first(x)))) =$$
$$second(first(x)).band =$$
$$second(x).band) \wedge$$
$$(first(first(first(x))) \prec second(first(first(x)))) \wedge$$
$$(first(first(first(x))) \in second(first(x)).term \wedge$$
$$second(first(first(x))) \in \mathrm{dom}(second(x).init))$$

A restatement of the definition of a transition between modes (originally given in Chapter 4, as equation 4.1):

$$\mathcal{T}^{mode} : (\mathcal{E} \times \mathcal{M}) \times \mathcal{M}$$

$$\forall x : \mathcal{T}^{mode} \bullet$$
$$band(first(first(x))) = band(second(first(x))) = band(second(x))$$

Finally the mapping between the transitions at the different levels:

$$band : \mathcal{T}^{mode} \rightarrow \mathcal{B}$$
$$band : \mathcal{T}^{model} \rightarrow \mathcal{B}$$
$$\mathcal{T}_{Mapping} : \mathbb{P}(\mathcal{T}^{mode} \rightarrowtail \mathcal{T}^{model})$$

$$\forall t : \mathcal{T}^{mode} \bullet band(t) = band(second(t)) \land$$
$$\forall t : \mathcal{T}^{model} \bullet band(t) = second(t).band \land$$
$$\mathcal{T}_{Mapping} = \{m : \mathcal{T}^{mode} \rightarrowtail \mathcal{T}^{model} \mid m \neq \varnothing \land$$
$$\forall a : \mathrm{dom}(m) \bullet band(m(a)) \sqsubseteq band(a) \land$$
$$second(first(a)) \sqsubset second(first(m(a))) \land$$
$$second(a) \sqsubset second(m(a)) \land$$
$$\exists map : Mapping \bullet (first(first(a)) \rightarrowtail first(first(m(a)))) \in map\}$$

Again, due to the limitation of precision not being included in the formal definition of the framework the additional constraint that the duration across the *init* and *term* event must be less than the precision of the higher band is omitted.

A final, and somewhat simpler issue to consider is that of the upper bound on the persistence of a mode. The original definition of persistence did not constrain the upper bound, however for consistency with other aspects of the framework this upper bound should be limited to the scale of the band; hence the definition of modal persistence becomes:

$$persistence : \mathcal{M} \rightarrow \mathbb{N}_1 \,..\, \mathbb{N}_1$$

$$\forall m : \mathcal{M} \bullet max(persistence(m)) \leq scale(band(m))$$

Another aspect not to be overlooked with the change to the definition of the activity duration is ensuring in circumstances where an activity is constrained to execute in a given mode that its maximum duration does not exceed the maximum persistence defined for the mode.

Conclusions of evaluation

The addition of modes to the framework provides a useful hierarchical extension that allows behaviours to be capture in a concise manner, constraining occurrences of events

and the execution of activities. There are some improvements to the definitions of modes within the framework which flow from the earlier changes to the bands, and activities and events; these are:

10. The representation of the mapping of transitions across bands benefits from the change to the definition of *Mapping*, this strengthens the model and removes a problem in that transitions no longer have to be resolved to an activity in the band where the modes are refined to mode models, it is now possible to simply refine the transitional event to a pair of events whose separation is less than the precision of the higher band.

11. The upper bound on the persistence of any mode should be limited to the scale of its band — this introduces a potential problem with the hierarchical decomposition of modes where the corresponding mode model contains a single mode.

The latter point introduces a potential problem with the hierarchical decomposition of modes where the corresponding mode model contains a single mode — in this case the constraint on the upper bound of persistence needs to be relaxed.

7.1.5 Model creation process

The model creation process developed in Chapter 5 has been applied in a largely mechanically manner in the creation of the three models developed for the case study; the area where the process was observed as being the least well-defined was that of the refinement of the overall system-of-systems model. This point will be considered later but first some other important modelling issues are discussed; these discussions build on those earlier in this chapter relating to bands, and activities and events, and their associated precedence relationships — a more general discussion of the process follows later in Section 7.2.2. The points below are discussed in their order of occurrence in the process.

Band selection and definition — using standard sets of bands

In the creation of the models generally the bands have been defined from first principles, the only exception being the use of Newell's RATIONAL and COGNITIVE bands as the basis for the bands of the same names in the Local Command Post system model. Given this use of Newell's bands a topic that requires consideration is that of building models, or parts of models, from sets of predefined bands — to a limited degree the modelling process already hints at the idea of standard bands in its identification of the possible existence of modal and human interaction bands in any model.

Newell's bands would clearly represent one such set of predefined bands, and it is likely that other sets could be created for different forms of technology. These sets of bands could be used to address, in general, the middle and the lower bands of the structure; this leaves at least two regions to be addressed. In the higher bands of the model the temporal dynamics are defined by organisational or strategic issues, thus in some limited and well defined domains, an example being that of the military environment of the system described in the case study, it is likely that an appropriate set of high level bands could be defined, and used generally for in models of systems relating to that organisation. More widely it may also be possible to define some appropriate bands for a less well defined structure such as "Western Society", where there is the use of measures such as years, quarters, months and weeks to organise business; however in this case the bands are likely to be less clear-cut, and hence partially ordered. In the general, domain neutral, case it seems unlikely that it will be possible to define a set of suitable discriminated bands at this level in the model given the diversity of the perception of time across societies described by Levine [54]. The other areas of the model to consider are those specifically related to the capabilities and behaviours of the system — these aspects are highly unlikely to be amenable to a common set of bands as the behaviours themselves will vary from system to system.

Having identified the possibility of some regions of the model where predefined sets may be useful, the practicality of the application of such an approach needs to be considered. Newell's bands relate to general human "capabilities", however quite often the systems that would be modelled have a very restricted range of roles for the human participants, and those are roles for which they are often highly trained, and in which they become highly proficient. In such circumstances it appears, as demonstrated in the case study, that Newell's bands will need to be tuned to match the dynamics of the particular system, thus in this case the bands are not used directly, but still form a useful basis and provide bounding cases.

The other area where standard sets of bands could possibly be used is in the lower technology dependent region. In this region there are no predefined sets of bands but the definition of small sets of bands for different technologies should be possible. In this case however there is a risk; unlike humans where the "technology" is stable, there are types of implementation technology where the temporal dynamics are forever changing, a notable case being that of computers which are getting ever faster. The risk in this latter case is that the bands of the model will be too closely tied to the implementation technology rather than the required dynamics of the system, and this may, through the imposition of some false constraints, increase the difficulty of making modifications to

a system at a later date. This does not necessarily preclude the provision of such fixed bands but it simply requires that they are not allowed to dominate the natural dynamics of the system.

This also demonstrates that the band-wise refinement is not the end of the story — replacement technology bands may be used provided that they do not "disturb" the higher bands, i.e. so long as they are also a refinement — they do not however have to be a refinement of, or equivalent to, the original band they replace.

Populating the bands with artefacts

In the model creation process the population of the model with artefacts was intended to follow the definition of all of the bands, and the definition of the inter-band mappings; however by the time that this point in the process has been reached some of the events have already been identified and placed — notably those related to the inter-mode transitions. This does not seem to represent any specific problem and it probably indicates that the process as currently defined is somewhat too prescriptive.

Aside from this premature allocation of modal-transition events the only other allocation issue is that relating to the events that originate in the environment; in these cases it is not uncommon for the events to be "required" in a number of bands, a requirement which offers three options:

- The event could be placed in the highest band where it is used, the model would then provide mappings to lower level events for use in the other bands;
- The event could be placed in the lowest band where it is required, and the model could reflect this as necessary into the higher bands; or
- A "copy" of the external event could be defined independently in each band where it was needed; in this case no mappings between the events would be provided.

It is not entirely clear which of these approaches is the best, or even that a consistent approach is needed. If the external event has diverse uses at different levels of the model then the latter option of redefining the event independently in each band is acceptable. In attempting to decide between the other two cases it is necessary to return to the issue of the preservation of precedence that follow inter-band mapping of artefacts.

Considering a case where the external event precedes, and causes, some internal process, the definition of the event in the highest band and its subsequent mapping to other events down through the bands leads to a problem — precedence weakens as the mapping

traverses the bands hence the causality may be diminished or lost. Taking the alternative approach of placing the event in the lowest band and flowing it up through the bands ensures the preservation of the causality — and is hence the approach that should be used where the event appears in several bands for a related reason.

The definition of the inter-band event to activity mappings proved to be the most difficult part of the allocation aspect of the process. As noted earlier the *Mapping* definition was only originally provided for event ⟶ activity mappings, but in a number of cases there was no tangible activity between the mapped start and end events. The changes made to the definition of the mapping proposed in an earlier section of this chapter should however alleviate this problem.

Refinement of the system-of-systems model

Returning to consider the major issue noted in the introduction to this section, that of the refinement of the systems-of-systems model being the least well-defined part of the model creation process. In this case the process failed to identify the issue of emerging behaviours that result from the interactions of the component models, this is not surprising as the framework only considers the case of a single model, which is clearly not an appropriate scope for these emergent behaviours.

The most immediate and obvious approach to addressing this issue is to create a single unified model by merging the initial system-of-systems model with those of the components. Attempting such a merger of even the relatively limited models developed in the case study would be complex, not least because the bands do not readily align. Making the small changes to the system-of-systems model in the refinement step of the process was both simple and justifiable, but attempting to merge all models into one as part of this refinement would be complex and more difficult to justify. The fact that the bands of two component models are underpinned by completely different rationales suggests that any such model would be of limited value.

The experience of the application of the system-of-systems refinement step suggest there is a need to be able to relate artefacts in separate models to allow the emergent behaviours to be captured more formally. How this might be done is not entirely clear at this point and is perhaps one of the more complex issues associated with the framework that needs to be resolved with respect to its practical application. One aspect of importance in forming these relationships is being able to capture, and possibly model, key elements of the environment more fully. In the case study it was noted that where there was an interaction between the payload delivery vehicle and the local command post via some unmodelled communications system, and there were potential problems in not being able

to ensure that the “communication” events met the appropriate precision constraints. This seems to imply that, in some cases, additional aspects will have to be modelled at the system-of-systems level in order ensure “completeness” of coverage of the interactions.

Conclusions of evaluation

The process produced three models that could be related to one another successfully; the major conclusions here are:

12. There appear to be some opportunities for using “standard” sets of bands as the basis for parts of models — this opportunity should be recognised by the model creation process.

13. Some issues with artefact allocation were noted, two significant ones being:

    - The order of allocation — in some cases, modes and their associated transitional events, artefacts are allocated when the band is defined – perhaps indicating that the ordering defined in the current process is too rigid;
    - Identification of the “correct” band, or bands, for external events.

    These both require careful consideration, neither poses an insurmountable problem;

14. There are the important issues relating to refinement and emergence, this is the least well defined and understood aspect of the modelling process — the issues surrounding cross-model interaction appear complex and require further consideration.

Other aspects of the process are considered more generally in a later section.

7.1.6 Tool support

A clear problem with creating the models in the case study was the lack of available tool support; today there are no tools supporting any aspect of the timebands framework or the creation of timebands models. However given the concise nature of the framework and the availability of a formal definition, the provision of tools should be relatively straightforward. It should be remembered that, as noted earlier, there are a number of issues relating to omissions in the formal definition; for instance the relationships between granularity and precision both within and across the bands — resolving these issues will somewhat simplify the provision of tools. This section can only consider the basic capabilities that might be required of any tools, their complete specification is outside the scope of this thesis.

Simply stating that tool support is necessary does nothing in itself to identify their scope and capabilities; the fundamental questions regarding tool support are:

- What is the scope of the tools — what do they need to be able to do?
- What forms of checking of the models should they provide?
- What forms of report or interfaces to other notations are required?

Whilst these questions are quite strongly related they are addressed individually in the sections below.

Scope

Consideration of the scope of the tools really serves to highlight the necessity in identifying the underlying information structure: what data needs to be captured, how should the namespace be managed, etc. Given this thesis envisaged that the timebands framework would be used in conjunction with other forms of description for the functional aspects of the architecture, it is important that the information captured and processed by any tools be readily accessed by other tools supporting the functional aspects — the use of an extensible technology such as XML would seem to be appropriate in this regard.

The basic functions of even the most minimal set of tools would include:

Model input The entry of the elements of the model, i.e. the bands, events, activities and modes, and providing their fundamental characteristics (granularity, precision, period, duration and persistence). This should include support for the process defined in Chapter 5 as this would simplify the creation of the model. Clearly there is a need to be able to build incomplete models to support the process with respect to the creation of the initial system-of-systems model — which is inevitably incomplete;

Relationship formation Using the basic elements of the model to create relationships including: the granularity mappings between bands, the precedence relationships between the events within a band, and the mappings between events and activities in different bands. A further issues here is the need to be able to relate events which are relevant in more than one band;

Visualisation There needs to be some way by which the model can be "seen"; there are many aspects to this, the bands and their contents, the relationships between elements within a band and the across bands. In the case of a system-of-systems it is useful to be

able to view the ways in which the bands in the different models are aligned, making the gaps and overlaps visible;

Cross-model relationships Particularly in a system-of-systems context there is a need to be able to relate artefact in different models to help to identify the emergent artefacts in the system-of-systems model.

In order that models can be used flexibly there is the need to be able to import and export both complete and incomplete models.

Checking

There are two fundamental aspects to checking that are internal to the timebands models themselves: completeness, and correctness. Whilst the scope of the former is quite obvious that of the latter is perhaps less so.

A model which may be described as "complete" will have the following characteristics: all bands are fully defined, including their orderings; and all artefacts are either refined to an activity or marked in some way as originating in the environment — in the system-of-systems context the last characteristic could be extended to allow for the origin being in another model. It should be noted that there is no need to define any relationships between the artefacts for the model to be "complete", however without such relationships the model is likely to be of little use.

Correctness is a broader concept and can be seen across a number of dimensions, these include:

- The ordering of the bands — granularity and precision should get no coarser at each lower level in the model;
- The precedence relationships between events must be honoured through any mappings across the bands;
- The mappings of events to activities and events to event-pairs must meet the constraints of the precision of the higher band — in the case of the event to event-pair mapping the determination of this might require the traversal over a number of bands;
- The mappings between modes and mode models, and the refinement of the simple inter-mode transitions in higher bands, to those between mode models in lower bands must be consistent;

- The validity of the relationships formed across models in a system-of-systems context, the directly mapped and emergent artefacts;

This last aspect is perhaps the most difficult of the above from a correctness-checking viewpoint.

Reporting

Reporting, or the need to extract data from the model, is a rather open-ended issue and hence is the least clear with respect to the need for specific tool support. There are however a number of areas where such capability might be envisaged; these are:

- The extraction of metrics relating to the model, thus providing some form of measurement of "quality" or complexity;
- The extraction of data to allow timebands model data to be used in conjunction with other architectural or design descriptions;
- The export of the model data to support other analyses over the model.

Undoubtedly other uses of the data would be found over time.

Conclusions of evaluation

Tool support is clearly an essential requirement for the timebands framework if it is ever to be adopted for significant systems; the major conclusions are:

15. The development of and underlying model based on common technologies is key to tool provision, this is assisted by the concise nature of the framework and the availability of a formal definition;
16. Data formats have to be accessible if the timebands framework is to be used in conjunction with other notations;
17. There are three aspects tool support: model creation and maintenance, checking for completeness and correctness, and various aspects of reporting.

Clearly this is only a starting point, tools supporting other aspects of model creation such as ones providing the ability to perform an impact analysis of "what-if" changes would probably be of great benefit in a system-of-systems context where in the case study it was observed that small band "adjustments" were needed in the refinement of the final overall model.

7.2 General issues

There are a number of more general issues that have emerged during the creation of the models, some of these relate to aspects of the framework, and others to the overall modelling process. In the following sections a few of these issues are briefly considered.

7.2.1 ...with the framework

There are two areas where the framework could be improved with respect to its application, both of which relate to the definition of precedence relationships. In the first case there is a suggestion to extend the notation to allow for a more concise expression of these relationships; in the second another extension is proposed that allows these relationships to be constrained by modes. These are discussed in the sections below.

Extended expression of precedence relationships

Whilst the original definition of precedence is sound it does not address the possibility of any conditionality in any of these relationships. In the case study there are a number of precedence relationships where one event is "dependent" on a number of others, and there are other cases where an event may be proceeded by any of a number of others. Such conditions may generally be expressed as conjunctions and disjunctions of relationships respectively, however some convenient "short-cuts" can be simply provided to allow for such circumstances; these are:

$$
\begin{array}{|l}
_ \preceq^{All} _ : \mathbb{P}\, \mathcal{E} \leftrightarrow \mathcal{E} \\
_ \prec^{All} _ : \mathbb{P}\, \mathcal{E} \leftrightarrow \mathcal{E} \\
_ \preceq^{Any} _ : \mathbb{P}\, \mathcal{E} \leftrightarrow \mathcal{E} \\
_ \prec^{Any} _ : \mathbb{P}\, \mathcal{E} \leftrightarrow \mathcal{E} \\
\hline
\forall\, E : \mathbb{P}\, \mathcal{E};\ F : \mathcal{E} \mid F \notin E \bullet (\\
\quad ((\forall\, e : E \bullet e \preceq F) \Leftrightarrow E \preceq^{All} F) \land \\
\quad ((\forall\, e : E \bullet e \prec F) \Leftrightarrow E \prec^{All} F) \land \\
\quad ((\forall\, e : E \bullet e \preceq F) \Leftrightarrow E \preceq^{Any} F) \land \\
\quad ((\forall\, e : E \bullet e \prec F) \Leftrightarrow E \prec^{Any} F))
\end{array}
$$

An example of multiple dependencies is that of the dependency of ↑Image-extraction on a number of other events given in equation 6.111 in the previous chapter. This would be more succinctly stated using one of the above forms thus:

$$\{\mathit{Capability\text{-}clock}, \mathit{Video\text{-}image}, \mathit{Axes\text{-}available}\} \preceq^{All} \uparrow\texttt{Image-extraction}$$

Whilst this is a relatively trivial example the use of such notation would be of significant benefit in the construction of larger, more complex models.

Modal constraints over precedence

Considered purely from a timebands perceptive the modal aspects provide additional scope for constraining the occurrences of events and execution of the activities; however this could be made more flexible if the constraints were to be extended to allow the application of some form of "scope" over precedence relationships. As the modes are indicative of overall conditions or behaviours then it would seem reasonable to be able to restrict certain precedence relationships, which also describe an aspect of behaviour, to particular modes.

In the development of the concept of system modes for the timebands framework no consideration was given to the notion of such constraints. Taking the definition of modal constraints provided in Chapter 4 as a template, a modal constraint over precedence could be expressed thus:

$$\{x, y\} \overset{w}{\Rrightarrow} \{(a, b), (c, d)\} \tag{7.3}$$

$$\{x, y\} \overset{s}{\Rrightarrow} \{(e, f), (g, h)\} \tag{7.4}$$

This can be interpreted in modes x and y the following precedence relationships hold:

$$a \preceq b \tag{7.5}$$

$$c \preceq d \tag{7.6}$$

$$e \prec f \tag{7.7}$$

$$g \prec h \tag{7.8}$$

An example of a possible use of this constraint would be its application to the weak precedence relationship between the end of the `Process-image` activity and the *Guidance-update* event given in equation 6.98 in the previous chapter. This relationship only holds in the **Engaging** mode, hence the following could be stated:

$$\{\textbf{Engaging}\} \overset{w}{\Rrightarrow} \{(\downarrow \texttt{Process-image}, \textit{Guidance-update})\} \tag{7.9}$$

Without such a statement the fact that this relationship only applies in the **Engaging** mode would only be apparent by considering the other modal constraints over the activity and event, and in complex cases such constraints over precedence might not be readily apparent.

Conclusions of evaluation

There are two areas of the framework where some extensions to the formalism would allow more complete models to be constructed; these are the provision of:

18. More convenient notations to capture multiple and selective precedence relationships.

19. Modal constraints over precedence relationship.

It seems likely that with wider use additional "improvements", similar to those described above, will be identified — these will further strengthen the framework.

7.2.2 ...with the modelling process

Whilst a number of the specific issues with the modelling process have been discussed earlier in Section 7.1.5 it is also useful to consider the process as a whole from start to finish identifying any residual issues at each step; to that end each step of the modelling process is discussed in turn in the sections below:

Step 1 — setting the context The context of the model was, for each element of the case study quite straightforward.

Step 2 — identifying stakeholders As noted in the previous chapter, on pages 186 and 223, in the creation of the two component models, there is a tension as to whether certain human elements of the system should be considered stakeholders, a part of the environment of the system, or part of the actual system. It appears difficult to generalise the resolution of this tension, the deciding factor is probably the depth of direct involvement in the behaviour of the system. Stakeholders can be of two basic forms, those that provide something that the system requires, and those that demands something of the system; these are both somewhat peripheral to the behaviour of the system; thus if they are "involved" in the behaviour of the system, they should be considered part of the system.

In the case of the Payload Delivery Vehicle it was considered that there were no human elements to the model, only some stakeholders, however in the case of the Local Command Post system the position is somewhat different. In this case there are three groups of stakeholders were identified: commanders, planners and operators, however it was considered that, in the case of the last two of these, their involvement with the operation of the system, and its resultant behaviour was such that these groups should be considered part of the system — in addition to being important stakeholders. Taking a

different view for each of the component models leaves an unanswered question: how should the groups given different roles in the component models be "treated" when the system-of-systems model is considered?

The answer to this really depends on the nature of the system-of-systems model and the level to which these groups affect its structure. In the case study the initial system-of-systems model defined the operators as being stakeholders, even though a time band was defined for operator interaction; in this case that approach was entirely appropriate as in this context they were both stakeholders and part of the system, here there does not necessarily appear to be a conflict between these two roles.

Step 3 — setting the bounds Generally the upper bound of the model appears to emerge quite naturally from the context (there is always some upper limit that is generally accepted), however there is, perhaps, a greater difficulty in defining its lower bound. The lower bound is driven strongly by the uses to which the model will be put, if these uses are identified clearly in the context then this bound should also emerge, though possibly not be as simply as for the higher bound.

The process implicitly assumes that a linear set of bands will be the result of its application, but this need not be the case. For example, at the lower levels there might need to be different bands at similar temporal levels to accommodate different technologies such as electrical, mechanical and software; and similarly, different forms of test and analysis may be applied across each of these. Whilst this issue was not considered in the case study, its lack of recognition appears to be a clear shortcoming in the process.

The process was perhaps too "ambitious" in that it suggests attempting to define the bounds of the system in terms of the characteristics of its highest and lowest bands; in the case of the models created in the case study this was not done, instead only the overall bounds of the model were identified, the scale of the highest band was defined, along with the granularity of the lowest. The bounding bands were only fully characterised once all other bands had been defined.

Step 4 — identifying modal behaviour Two of the three models created displayed a significant degree of modal behaviour, with two modal bands being defined for the payload delivery model— this step presented no issues in the case study.

Step 5 — identifying tightly coupled human interaction Human interaction was identified most strongly in the Local Command Post model. As noted earlier, two bands in this model, RATIONAL and COGNITIVE, follow very closely the bands of the same names

in Newell's definition; the use of his bands as a basis for those defined for this model was discussed earlier in Section 7.1.5.

Step 6 — identifying modal–human interaction None of the models created explored the issues associated with this step of the process. Given the diversity of the subject matter across the models this is perhaps a little surprising, and is possibly indicative that the step is redundant. However, given the immaturity of the process and its application, the step will remain. In further applications of the process it would be interesting to consider if, for the component models at least, there is the case for having either modal or human interaction bands, but not both — though from the experience gained in generating the models in the case study this appears unlikely to be the case for the system-of-systems models.

Steps 7 and 8 — identifying and partitioning the remaining bands These steps pose no real issues; assuming that either some modal or human interaction bands have been identified earlier in the process then these two steps are well segregated. Of course it is not mandatory to have either model or human interaction bands, if both of these were to be absent then there might be an issue with the coupling between the technological and organisational "regions" of the model, but this was not experienced in the case study.

Step 9 — determining the inter-band relationships In the process as presented in Chapter 5 the intent of this step was to define all of the inter-band mappings and define the precision of each band. In practice the precision of each band was defined when the band itself was identified; to a large extent this was due to the selection of a fairly uniform ratio between precision and granularity — thus the precision could be defined in the absence of the details relating to the other bands. Defining the inter-band granularity mapping is a largely mechanical exercise for the models created in the case study as the bands do not significantly overlap. It would be interesting to consider the generation of these mappings for other models where there is a greater degree of overlap of the bands, or where not all bands are directly related.

Identification and allocation of artefacts The aspects of the process related to the identification and allocation of the artefacts presented no real issues, this was probably helped in the cases of the models created here by the fact that the bands were largely distinct minimising the potential for ambiguous allocations. The process had already identified the importance and uniqueness of events, and the fact that their frequency of occurrence was crucial in identifying in which band they should be placed — thus this

aspect of the process is unaffected by the changes to the activities and events suggested in Section 7.1.2 above.

Artefact interactions and relationships The final step in the process is perhaps the most complex — the forming the relationships between the artefacts both within and across the bands. The intra-band interactions were easily identified for each of the models, as were those relating to modal constraints. The mappings between bands were however somewhat more problematic, largely because of the restriction of having to map an event to an activity; the suggestion earlier in this chapter to allow event to event-pair mapping would certainly simplify this part of the process.

Conclusions of evaluation

Overall the process worked in that it produced a set of models that could be used together in a system-of-systems context; the major conclusions being:

20. The process is possible somewhat prescriptive and that for a different class of system it might not be as successful.

21. The goal of **step 3** of the process, that of setting the bounds of the model, should be weakened to only define the scale of the upper band and the granularity of the lower band, the final characteristics of the bounding would then be set at the point in the process where the final bands are defined (in **steps 7 & 8**).

22. The bands for the models produce were relatively strongly ordered, a system with a non-linear set of bands would be an interesting example to consider in the future.

In the absence of any evidence from the modelling of other classes of systems there appears to be no good reason to change the fundamentals of the process, except perhaps for weakening the aim of the step 3 as described above.

7.3 Evaluation and assessment summary

The models produced in the case study have exposed some issues that need to be considered in the development of the framework and its application in developing future models. The solutions to these issues do not necessarily provide the final answers, indeed this would be highly unlikely given this is the first application of the framework and process, but they certainly provide some useful guidance for the future.

A number of important specific issues have been discussed, and changes and clarifications to the framework have been proposed to address some of the apparent limitations. The most fundamental changes are those to the relationship between the activity and

the event, and the provision of ranges for duration, and the concepts of period, minimum inter-arrival and jitter for each of these respectively. Additional flexibility has been suggested for the concept of inter-band mapping, which in turn improves the clarity mappings across a hierarchy of mode models.

The application of the process defined in Chapter 5 has succeeded in producing a set of models for both component systems and an overall systems-of-systems, and although a number of issues need to be considered further it is perhaps too early to consider significant changes to this process. Tool support is a notable area that needs to be addressed if the timebands approach is to be adopted more widely.

7.3.1 Open questions and issues

Following from the evaluation and assessment there are a number of areas where there are significant outstanding issues, some important ones being:

The lack of a formal relationship between granularity and precision This is only aspect of the framework where there is a real lack of clarity, its resolution would significantly enhance the framework and "remove" a source of uncertainty;

Failure conditions The models have only been considered from the perspective of "normal" operation where all timings are adhered to, no consideration has been given to the issues of failures or "broken" timings;

Consideration of model "quality" To-date there has not been any consideration of assessment of the quality of the models other that from a rather vague qualitative viewpoint, this is not surprising as this is the first application of the framework. In the longer term there is the question of how can the quality of a model be assessed, what are the essential metrics that allow the comparison and assessment of models;

How may the framework be deployed to the greatest effect The case study considered timebands in isolation from other notations. Normally other views of the system will be formed using a range of notations — this poses the question regarding how best to combine the use of the different notations;

Value of behaviours The behaviour defined in the timebands framework has not been used in any of the models developed for the case study, as a consequence no real conclusion can be drawn regarding is utility — except of course to question that if it was not required for these systems then for what type of system would it be useful?

7.4 Overall summary and final conclusions

Over the course of this thesis the focus has been on how best to capture the temporal aspects of a system, the fundamental hypothesis being that: giving time a central role in the structure of systems would result in generally better real-time systems. The case study, and the subsequent evaluation and assessment of the framework that it motivated have shown that the timebands framework may be applied to systems of industrial proportions with some tangible benefit; a clear example being the exposure of the emergent temporal properties that become apparent when the component systems are incorporated into a system-of-systems.

Detailed conclusions regarding the framework and associated modelling process were presented earlier in this chapter, thus all that remains here is to provide an overall summary of the thesis and return to consider the original central proposition in the light of these detailed conclusions. A series of specific topics which would warrant future research are also identified.

7.4.1 Thesis summary

This thesis is motivated by the observation that time is an often overlooked, or at least under-considered, aspect of a systems character. Current architectural and design notations are focused on the functional aspect of the system, with little attention being placed on non-functional properties; in the case of real-time systems this appears to be a fundamental weakness given the importance of time to these systems.

In order to address this weakness the timebands framework had been proposed and developed by Burns *et al.* [11]. This largely ignores the functional aspects of a system and provides an apparently simple banded structure of abstractions of time, allowing instantaneous occurrences at one level to be refined to processes, which operate over some measurable period of time, at some lower level. Whilst immature, and rather limited in some areas, this framework appeared to be a good basis for providing the temporal view of systems that it all too often lacking — and usefully it is formally defined.

Some of the limitations and omissions of the timebands framework were addressed. The inclusion of support for the concept of modes, as distinct from simple states, was a major addition — modes in this context are considered to define a behaviour of the system, as distinct from its functions or conditions. After consideration of two contrasting approaches, this extension was included in such a way that the:

- Original framework was left unchanged; and
- Dependence of the extension on the original framework was minimised.

The formal definition of the framework was enhanced to accommodate this extension.

The extension provides a flexible set of mode-related constraints that may be applied to occurrences of events and activities; the constraints refer to modes defined in the same or higher bands than those of the constrained events and activities. In addition to this extension, a process was defined to support the creation of timebands models; this addressed the creation of models for both stand-alone, or component, systems and those of systems-of-systems.

The extended framework and process were applied to an extensive industrially based case study, resulting in three models: two for component systems, and one for an overall system-of-systems. The systems modelled covering a wide spectrum of temporal issues, from real-time autonomous behaviour to human interaction.

Finally the results of the case study were used to support an evidence-based evaluation and assessment of the framework. Through the application of the modelling process in the case study, and the subsequent assessment, the fundamental nature and importance of events has become increasingly clear. As a consequence of this increasing importance suggestions for changes to the framework have been proposed that distinguish events and activities more clearly. This change has resulted in the strengthening of other areas of the framework, such as the definitions of mappings between events and other artefacts in separate bands, and the mapping of transitions through modal hierarchies.

The conclusions emerging from this evaluation and assessment are used in support of the overall conclusion presented in the next section.

7.4.2 Final conclusions

The final question to be answered is how does the central proposition stand in the light of the conclusions identified in this chapter?

The central proposition, originally presented in Chapter 1, states:

> *Current approaches to the development of real-time embedded systems fail to capture adequately their non-functional properties, especially time, in a consistent and coherent manner. Using a banded notion of time as the primary consideration in their architectural organisation addresses this failing. This results in systems that have well defined temporal properties that do not unnecessarily constrain their implementation, and provide a common framework for the combination of these systems.*

Considering each of the main points of this proposition:

- In general the treatment of non-functional properties in general, and time in particular, is not well supported by current architectural and design notations, it is doubtful that any of the developed/mainstream approaches could be described as either consistent or coherent.
- The use of a banded notion of time as the primary consideration in forming the architectural structure certainly ensures that temporal issues are captured in a consistent and coherent manner — as demonstrated in the case study.
- The timebands framework fundamentally supports levels of abstraction for time; as abstraction facilitates implementation choices it follows that the timebands framework, whilst capturing the fundamental timing properties of the systems, does not impose undue constraints on their implementation.
- In the case study the models of the components and the system-of-systems were combined successfully, thus demonstrating the ability of the framework in supporting such combinations.

Thus the central proposition holds. However, the evaluation undertaken in this thesis is limited in certain respects; the key ones being:

- It focused on embedded real-time systems, and ones with relatively limited man-machine interactions — wider socio-technical systems were not considered;
- The conflicts between the bands of the various models were relatively limited and hence easily resolved — attempting to combine pre-existing models, with diverse band structures, in the construction of a complex system-of-systems model may prove to be a more problematic undertaking;
- The evaluation did not address the issue of combining the timebands models with the functional and structural views — these need to be combined if the benefits of the timebands approach are to be maximised.

7.4.3 Suggestions for further research

To a large extent, the suggestions for further research, presented below, stem from both the open issues and questions identified earlier in this chapter, and a more general observation made earlier in the thesis. The topics are:

Completing the formal representation of the framework. There is a need to finalise the formal definition of the framework, the relationship between granularity and precision being the fundamental aspect that is not currently adequately addressed. As a

consequence of this improvement changes to the constraints over mapping and inter-mode transitions will undoubtedly be required.

Extension to the framework to address the issue of state. Whilst the framework was extended to include modes as part of this thesis there appears to be equal merit in also including the concept of state and considering how to exploit the conditionality this can impose.

The use of the framework with respect to failure conditions. One of the advantages of the framework stated by Burns *et al.* [11] was its ability to handle failures, for example the early or late arrival of events. Approaches to the treatment of these failures and their interactions with other facets of systems such as safety would be an interesting topic.

Consideration of model quality and the definition of metrics. Knowing what are good and bad timebands models, and what are the key drivers for this quality, are ultimately essential to the successful exploitation of the timebands framework. The definition of metrics and their quantification are central to being able to assess model quality.

Repeat the case study with a fixed, predefined set of bands. Building the models using a fixed, predefined set of bands would allow the bespoke approach to band definition adopted in this thesis to be compared with a potentially simpler alternative. This would give some useful insights as to what degree it might be possible to fix the bands, and of the value of the bespoke approach. A potential benefit of being able to use a fixed set of bands would be to make the framework more accessible.

Deployment of the framework: tooling, education and technology transfer. The provision of tools is clearly essential; there are a number of areas of research that can be seen under this general heading, not least of which is the definition of a metamodel for the framework. This would provide the basis for future tools and is thus a fundamental first step. Simply providing tools however only solves part of the problem, there also needs to be associated education and guidance on the timebands approach.

Inter-working with other notations. If the timebands framework is to used to its full potential the its relationship is of crucial importance. Approaches to the combination of timebands with other notations need to be investigated and trialled using significant examples.

Assessment of the framework across a wider domain. The case study is based on a product for the military sector, a reasonably constrained domain. How does the approach perform when applied to a more diverse environment?

Investigating the relationship between functions, states and modes. Perhaps a more philosophical question relates to the discussions in Chapter 3 regarding the difference between modes and state, and the hypothesis that modes are in some way a function of function and state. This is a far wider and more fundamental issue but its understanding would allow a better assessment of the value of modes in the timebands framework (and their use in systems more generally).

7.4.4 Contributions and closing remarks

The timebands framework is certainly a novel approach to the often overlooked non-functional property that is central to embedded real-time systems — that of time.

This thesis has contributed to the framework in three major ways; it has:

- Identified and defined additions and changes to improve its utility and clarity;
- Defined a process for the application of the framework; and
- Applied this process to develop a number of models as part of a large-scale case study based on a system of significant size and diversity.

The approach chosen to evaluate the framework, that of a single large industrial case study, had both advantages and disadvantages. Whilst it only explored the framework in a limited domain, it both resulted in a range of models, each of which explored the framework in some depth, and had to deal with real issues relating to modelling a system of industrial proportions. The alternative approach of developing a number of more synthetic case studies to explore the framework more widely would not have given the same depth of insight, though admittedly they would, if carefully constructed, given a broader coverage. The case study was a significant undertaking, spanning several months.

It is often the case with novel, or niche, notations and approaches, that the move from the academic environment to industry usage proves to be a difficult transition, an obvious example being that of formal methods [43]. There are undoubtedly a number of reasons for this difficulty, however one of the main ones is that of accessibility — the notations are often complex, and there is little in the way of (industrial strength) tool support, or education and guidance in their usage. The timebands framework has a fundamental advantage in this regard however: its simplicity. This is the key to it being easily understood thus making the provision of tools being relatively straightforward and minimising the education challenge.

Appendix A

A Review of Architecture Description Languages

There are a large number of notations available that may be used to describe the architecture or design of a system, generally these are focused on the functional and structural views of the system. This appendix provides a broad review of a range of these notations.

The scope of this appendix is somewhat wider than that suggested by the title — rather than simply considering languages that are universally agreed to be Architecture Description Languages (ADL) the scope is widened to include those languages that might at best be considered to be on the periphery of this group. In general the term "Architecture Description Language" is missing the prefix "software" — most, but not all, ADLs are focused on the software aspects of the system, and thus often fail to consider the wider aspects of the system such as computing and other electronic hardware, electro-mechanical actuators and measurement sensors, and its operational environment. When considering embedded real-time systems it is important that the descriptions capture not only the software aspects, but also those of the computing platform, and the interactions between these two domains. In this thesis the issues relating to real-time systems are considered in a yet wider systems context, and include the "issues" relating to human interaction and wider organisational issues, this demands that the scope of the languages extend somewhat from that of simply the software domain. For the remainder of this thesis the term ADL will be used quite loosely to encompass any language that might be used to describe aspects of the architecture of the system.

In reviewing ADLs it is quite hard to know either where to start, or indeed what aspects of the languages to consider. However there are a number of important areas that should be considered. The origins and history of ADLs are fundamental to understanding their

"rationale" and limitations, thus this is, perhaps, the logical place to start. As will become clear ADLs are largely, but not exclusively, focused on the structure and functionality of the system, and are less concerned with its behaviour — however for this thesis the interest is in how the non-functional properties relating to time are accommodated. Thus when considering specific ADLs the focus has to be on this aspect, though it is also useful to see how other non-functional properties are also addressed or accommodated. Finally the current status of ADLs, from both research and usage viewpoints, is of significant interest in that this defines the pool from which suitable languages may be drawn for the application to real systems.

Given these issues there are a number of topics that need to be addressed in this review; these are:

- The origins and history of ADLs — where did they come from, and what motivated their development;
- What exactly are ADLs — the essence of architecture in general, and ADLs in particular;
- The relationship between ADLs and the UML — the issues associated with the use of the UML and it variants for architectural description;
- Other ADL-like languages — languages from other areas that attempt to address architectural issues;
- Application to real-time embedded systems — how has the support for these systems been addressed and how is it developing.

The remainder of this appendix will address, in turn, each of the above issues, starting as suggested above with their origins and history. Finally there is a more in-depth consideration of a number ADLs — including a comparison against a set of criteria and some limited examples of their use.

A.1 Origins and history

The origins and history of ADLs are not entirely clear, or indeed generally agreed. In a special issue of IEEE Software in 2006, celebrating the previous ten years of software architecture conferences and workshops, two chronologies for the development of software architecture were given by Kruchten *et al.* [52], and by Shaw and Clements [95]. Where Shaw and Clements see the origins of software architecture firmly located in the

mid-1980s, Kruchten *et al.* identify the first reference to software architecture as being by Ian Sharp at a NATO conference on software engineering in 1969 [86]. However in both [52] and [95] the views were more specifically focused on the discipline of software architecture in general, rather than on ADLs in particular.

A.1.1 Origins

It is clear that the impetus for the creation and development of ADLs parallels the work on software architectures, indeed at times it is can be quite difficult to separate the two — however their origins are somewhat different. Research into software architecture started, in earnest, in the early 1990s, however its origins can be traced much earlier — to programming in the large in the late 1960s, notably by Dijkstra [29], in a similar era to that of the NATO conference mentioned above. The origins of ADLs come from somewhat later work — the study and development of Module Interconnection Languages (MILs) in the mid–late 1970s, initially by DeRemer and Kron who described the language MIL75 [28].

The motivation for software architecture research in the 1990s resulted from the perceived need to manage the problems associated with the increasing size of software intensive projects and systems; too many of these were seen to be failing and not delivering the promised benefits. Reuse was seen as being one of the areas that would benefit from a clearer view of architecture. Much of the funding for this work appears to have come from the US Military. Shaw and Garlan [97] note that as the ideas around architecture developed, the then current descriptive forms associated with software design, informal diagrams using boxes and lines, were seen as not addressing the large-scale concerns associated with architecture. It was also recognised that these informal notations were highly ambiguous, and that a way of addressing and describing the higher level abstractions of the software (and systems) was required; from this field of research a number of ADLs rapidly emerged.

Over a period in the early-mid 1990s a large number of different ADLs were described; examples of these include: Wright [1], UniCon [97], Rapide [56], MetaH [103], Darwin [57], and Modechart [49]. Some of these were developed from first principles, others were derived from existing notations; most, but not all, of these languages were developed in academia — a notable exception being MetaH, which was developed by a commercial organisation (Honeywell).[1]

[1] In the early years of ADL development there were few, if any, other examples of commercially supported languages and associated toolsets.

There is no apparent unified philosophy underpinning these ADLs, they all appear to have evolved out of separate strands of research, each focused on different issues — some even originate from the same organisation; for example both UniCon and Wright were developed at CMU [96, 1]. There appears to have been a strong degree of domain specificity in the early languages, hence a wide proliferation and variable approach is perhaps not unexpected. Possibly due to the acknowledged limitations of the informal boxes and lines approach Medvidovic and Taylor [61] noted that there is a strong element of formalism behind a significant number of these original ADLs. This, in addition to the "deliberate" domain specificity, also probably had some impact on the generality of any given language in that the style of the formalism strengthened the domain specificity; however the formalism does have the advantage of ensuring coherence and consistency.

Following on from these original (first generation) ADLs, which were only applicable to a limited number of aspects of the system or a single "class" of system, there emerged a class of languages classified as architectural interchange languages, or architecture meta-languages. These provide frameworks that allow the original, highly specialised ADLs to inter-operate; thus enabling different aspects of a system to be analysed using the appropriate ADL whilst retaining a common and coherent description. The first of these interchange languages appears to have been Acme from CMU [33]; somewhat later Wile described a similar language AML [104].

More recently (2000 – 2005) there has been continued development of ADLs in academia. Whilst a few of the original set have faded away, the development of a second generation of ADL can be seen to emerge. These have built on the foundations of the original ADLs and the architectural interchange languages, and through the use of XML as the meta-language for the ADL, have enabled extensibility. This has the potential benefit of facilitating the interchange between different ADLs, and simplifying the development of other, specialised ADLs or extensions. The xArch project (UCI and CMU), which exploits XML in this way, has produced xADL [22] and xAcme [93]. The benefits of using XML as the meta-language are seen as being wide-ranging not least because of its clarity, openness and extensibility, and its support for traceability and collaboration, noted by Pruitt *et al.* [85].

Since 2003 the academic focus for ADLs has shifted somewhat. The original languages, together with their derivatives and successors have rather "stagnated", and whilst there have been some additions these have been few compared to the proliferation that occurred through the 1990s, a more recent example being π-ADL [80]. Over this period there does seem to have been a wider consideration of the issues of non-functional prop-

erties. Wile [105] proposes calculi to address non-functional issues, and Denford *et al.* [27] discuss the refinement of architectures based on their non-functional properties; however this remains a field where there is still relatively little evidence of research.

A.1.2 Standardisation

Concurrent with this research, and commencing in the late 1990s, software architecture and ADL related work has been undertaken in various standards organisations. The Institute of Electrical and Electronic Engineers (IEEE) has developed a standard to cover the use of ADLs [45] — IEEE-1471(2000) *A Recommended Practice for Architectural Description of Software-Intensive Systems*. This standard is independent of any particular ADL or particular set of capabilities, its goal being to identify what constitutes a software architecture, the artefacts and "views". This approach, of not specifying the actual language, can be contrasted with that adopted by other groups where the goal has been to define a standard ADL — albeit limited to a particular domain.

The Society for Automotive Engineers (SAE) have adopted one of the original ADLs (MetaH) as the basis for their standard ADL for avionic systems, the "Avionics Architecture Description Language" (AADL), [89]; whilst, in the area of XML, the Open Group have published the specification for ADML — a mark-up language for architecture description [37]. Interestingly, this latter language is also based on an earlier ADL — in this case Acme. The standard for AADL was finally issued by the SAE as AS5506 [90]; more recently this has been updated as AS5506/1 [91] to include a number of annexes describing aspects such as a UML profile and data interchange formats. It is interesting to note that when finally published the acronym AADL had changed, removing the focus on avionics, becoming simply the Architecture Analysis and Design Language.

A.1.3 ADLs, the UML and more recent developments

Throughout the development of ADLs there has been much consideration of the integration of ADLs with other design notations; examples come from Robins *et al.* [88] and Péraire *et al.* [82] — the alternative design notation in both these cases was the Unified Modeling Language (UML). The benefits suggested from this work were largely seen as being the leveraging of mainstream tools — it is also suggested that such integrations may help in the comparison of ADLs as implicit assumptions would have to be explicitly stated in common terms. However in the early days of the UML (pre-UML 2) and ADLs, the differences in approach and abstraction between ADLs and UML restricted progress in this area. This lack of progress is probably exacerbated by the view, widely held in the UML community (e.g. [53]), that UML is itself an ADL, and they therefore believed that

there were no issues to address. This early view was, perhaps, somewhat undermined by one of the stated goals of the revision of UML to the UML2 standard was to better support architectural views — in [18] there is a discussion as to the merits or otherwise of the UML being an ADL.

The standard for UML2 was finally published in 2004 by the OMG [74], it has been subsequently updated [77, 78]. Even with UML2 there is a recognition that in its basic form the UML does not address key aspects of architecture for certain classes of systems, or at certain levels of abstraction. These issues have resulted in the UML profile for Modelling and Analysis of Real-Time and Embedded systems, or MARTE, and the language SysML, the Systems Modelling Language to address the system's engineering domain — these are defined in [79] and [76] respectively. It is interesting to note that there is a convergence of sorts between the UML and AADL in that the MARTE profile describes how it might be mapped to AADL like description, or how MARTE could be used to describe systems in an AADL like manner. Likewise one of the annexes to the AADL standard provides a mapping to the UML.

Whilst the UML and AADL could both be considered to be largely domain independent, "new" languages continue to emerge that are very domain specific, a recent example being EAST-ADL [26] which focuses on embedded electronics systems in the automotive industry. This language demonstrates a trend in the more commercial languages — that of being compliant with, or mappable to the UML.

There has also been some consideration given to the use of the UML in describing aspect-oriented architectures; Gorton [36] notes that a number of the approaches to aspect-oriented design extend the UML by developing an appropriate profile. He suggests that the UML combined with the Model Driven Architecture (MDA) approach will assist in the process of generation of the aspect code and the associated weaving — though no examples of this are cited.

A recent trend has been the emergence of architectural frameworks such as DoDAF [31] and MoDAF [65]. Theses are not ADLs as such, but are frameworks defined for particular domain that guide how the architecture is constructed and what artefacts need to be produced.

A.1.4 Modelling non-functional properties

Non-functional properties, especially dynamic or run-time ones, have never been a particular focus for ADLs. Taking an extreme example, the description of the language ARMANI [66] states that issues relating to run-time behaviour are not considered.

More recently, there does seem to have been a wider consideration of the issues of non-functional properties. As mentioned earlier Wile [105] proposes calculi to address non-functional issues, and Denford *et al.* [27] discuss the refinement of architectures based on their non-functional properties. The MARTE profile for UML [79] contains a sub-profile specific to non-functional issues, and more specific ones associated with time, resource allocation and utilisation; as yet there is no evidence that this profile will result in architectures that better address these issues.

Overall however, this remains a field where there is still relatively little evidence of research, or progress towards a common approach to addressing non-functional issues.

A.2 What is (and what is not) an ADL?

Having looked briefly at the history and development of ADLs and the state of current research, the substance and role of an ADL can now be considered in more depth. There are two areas to be consider:

- The relationship between software architecture and ADLs;
- What precisely is an ADL.

These are considered in turn in the sections below.

A.2.1 The relationship between software architecture and ADLs

There appears to be a general consensus that there is no consensus as to what is, and what is not, an ADL. The definition of what constitutes an architectural description language is not clear cut, and there certainly is no simple definition of one to-date. Given the close alliance of ADLs with the study of software architecture it might be expected that the definition of this term would also be variable (or at best imprecise), and indeed it is. In their seminal paper Perry and Wolf [83] define architecture as being comprised of *"Elements, Function and Rationale"*, with rationale being the "glue" that binds the other elements into an architecture. Other examples of these definitions include:

> *Software Architecture is a high-level view of a system that focuses on structure, communications protocols, assignment of software to hardware components and so forth.* — Medvidovic and Taylor [61].

> *Software Architecture is a level of design that goes beyond the algorithms and data structures of the computation: designing and specifying the overall system structure emerges as a new kind of problem. Structural problems include gross organisation and global control structure; protocols for communications, synchronisation and data access; assignment of functionality to design elements; physical distribution; composition of design elements; scaling and performance; and selection among design alternatives.* — Shaw and Garlan [97].

> *The software architecture of a program or computing system is the structure or structures of the system, which comprise software elements, the externally visible properties of those elements, and the relationships between them.*— Bass *et al.* [3].

There is often agreement on the general principles, however the details of the definitions are far from consistent. The second of the definitions above, importantly, identifies the non-functional issues of scaling and performance; these are often overlooked. It is also interesting to note that a very clear divide between hardware and software is often apparent (as made especially clear in the first of the quotes above). Given it has already been recognised that embedded real-time systems are a collaboration between hardware and software, this software focus is one of which to be wary when considering these languages.

Whilst the definitions of software architecture and ADLs might be difficult to pinpoint, the purpose(s) and general characteristics of an ADL can be identified more easily. ADLs have been created and evolved to provide a vehicle allowing the gross structure of systems to be described in a systematic, precise, unambiguous and consistent manner; they replace the generally vague and ambiguous (boxes and lines) diagrammatic forms that are commonly used to describe high-level system and software designs.

The basic characteristics of an ADL, identified by Shaw and Garlan [97] and others, are similar to those for any programming language (PL). They should support:

- Components
- Operators
- Abstractions
- Closure
- Specification

However these are only general categories — they note that the content of each in the case of ADLs is very different from that in the case of programming languages. As an example of the difference, the specification element in the case of the programming language relates the semantics to the syntactic form in the area of functional operation; in the case of the ADL this has to encompass the non-functional issues of performance, fault-tolerance etc.

Shaw and Garlan [97] suggest that a system to support architectural design must provide models, notations and tools to describe the architectural components and their interactions; it must support large-scale, high-level designs and the adaptation of these to specific implementations. It is clear that there is a close relationship between the language and the environment (of the system): language provides precise description, environment the context that make the description usable (and reusable). They suggest that there are six classes of properties that characterise what an ideal architectural description language should provide; these are:

- Composition
- Abstraction
- Re-usability
- Configuration
- Heterogeneity
- Analysis

However other researchers have determined that ADLs should have a somewhat different set of properties; Luckham and Vera [56] suggest:

- Component abstraction
- Communications abstraction
- Communications integrity
- Ability to model dynamic architectures
- Ability to reason about causality and time
- Hierarchical refinement
- The ability to map behaviours to architectures (relativity)

Clearly there is some difference of view as to the properties an ADL should possess, in order to resolve some of this difference it is necessary to consider the characteristics of an ADL in more detail.

A.2.2 What is an ADL — its essence and characteristics

Moving on to consider the concepts addressed by ADLs in more detail. Whilst there is some common ground in the above lists it is clear that there are different points of view as to what constitutes an ADL — thus there is no agreed definition of what makes a language suitable as an ADL. Even though the set of properties used to define an ADL might vary between researchers there is a common set of concepts that are consistently supported, at least to some degree, across all ADLs conceived as ADLs, *cf.* those developed for a different purpose and adopted as an ADL. These are:

- Components
- Connectors
- Compositions
- Constraints — though these were not universally recognised in the early stages of ADL development, and in some cases are implicit rather than explicit.

More recently Bass *et al.* [4] have moved away from the term *component* and have adopted that of *element*. Their reasoning being that the term component is closely associated with component based software engineering, which has a decidedly runtime flavour, the term element is chosen to convey something more general.

The degree of support afforded each of these four concepts gives some indication as to the differing views of the roles of an ADL in the academic community. In the case of connectors there is a broad range: from being discrete items with "first-class" status; to being inherent in the architecture (they exist only to interconnect a number of components). Supporting the first extreme, Metha *et al.* [63] have proposed a taxonomy of software connectors.

In an early survey comparing ADLs, Clements [16] notes that the languages can be discriminated from one another by how much architectural information they represent. He goes on to assert, supported in part by the results of his survey, that languages born as ADLs show clear advantages over those built for some other purpose and later adopted as ADLs. The weakness of the non-ADL used as an ADL appears to be the ability to produce

artefacts that do not, reasonably, correspond to an architectural view of the system. An example suggested here is the language Modechart, defined by Jahanian and Mok [49]; the state machine components can be interpreted as architectural components, but it easy to produce artefacts that have little relation to the architecture of the system. It is also clear that some non-ADLs are capable of modelling some aspects of architecture.

Medvidovic and Taylor [62] suggest that the focus on conceptual architecture, and the explicit treatment of connectors as "first-class" entities is key in differentiating ADLs from MILs, PLs, and Object Oriented (OO) notations and languages. In this it is important to note that the boundary between ADLs and MILs is less clear-cut than between ADLs and the other forms (due to the recognition of "connectors" in MILs); however, Prieto-Diaz and Neighbors [84] note that MILs are neither concerned with what a system does, nor how it is implemented; these are both issues considered to be fundamental (at some level) for ADLs.

It is perhaps useful to consider briefly why programming languages are not ADLs. In general these are far too low-level to be suitable for the role, both in terms of the functions performed and the connectivity between components. Some later approaches in this area that made a better attempt to recognise the problems of the interaction between components (e.g. occam [51] and Ada [100]) fail the ADL test as they only support a small fixed set of communications paradigms, or they enforce a strict, single purpose organisation — thus they are not ADLs in even the broadest sense.

The focus of an ADL needs to be on the artefacts as structures rather than the details of their function and implementation. Interestingly, in his survey Clements [16] also noted that while the languages differed greatly in a number of aspects there were equally some very strong correspondences — notably the support for data and control flow as interconnection mechanisms. The differences probably originate in the domain specificity of the languages.

ADLs that do not define a particular relationship between the description and implementation have the components and connectors at a high level of abstraction; these are sometimes described as *implementation independent* languages — examples include Wright and Rapide [1, 56]. These contrast with ADLs where there has to be a much tighter relationship between an architecture and its implementation, here there is a direct relationship between a component and its implementation; these are sometimes described as *implementation constraining* languages — early example of these include MetaH and UniCon [103, 97]. The terms *implementation independent/constraining* are due to Medvidovic and Taylor [62]. More fundamentally, however, Clements [16] suggests that a language

is an ADL if the essence of the information and abstractions are architectural in nature, and any supported analysis relies on such architectural information.

Medvidovic and Taylor have developed, and maintained, a classification and comparison framework for software ADLs (most recently published in [62]). This recognises the diversity in many dimensions among ADLs and attempts to identify a set of characteristics that allows ADLs to be compared with one another. But even at the fundamental level of *focus* they identify significant variation across the ADLs surveyed.

A common trait across early ADLs is that they often have a formally defined syntax and semantics, and several make use of formal notations to model certain aspects of the architecture, examples being the use of CSP [42] with Wright, and RTL [48] with Modechart. This formalism has a benefit from a consistency and lack of ambiguity viewpoint; however there is a down-side to this. Medvidovic and Taylor [61] recognised that on occasions the need to leave decisions open is desirable at the higher levels of abstraction, and that formalism can make this difficult; this appears to imply that the formalism is only used in a concrete way, as an abstract approach is more likely to "provide" non-determinism, which in turn gives scope for implementation options. At this time it is not clear which, and how much formalism is appropriate at the architectural level, or which formal methods are best suited to architectural modelling.

There appears to be a clear and important distinction that is often overlooked in the literature: that between ADLs that are used to describe purely software structures/systems, and those that support the modeling of a wider class of system, i.e. those encompassing the hardware, platform or some external agent (e.g. operator) in addition to the software. In the case of an embedded real-time system the ability to take this latter viewpoint is clearly crucial — the hardware–software partitioning is one of the key decisions in the implementation of an embedded real-time system, and is fundamental to its architecture. In the case of other classes of software centric/intensive system both the hardware and associated operating system or middleware are often pre-defined components (or constraints) of the solution and hence the software architecture has to be melded to fit.

In [61], Medvidovic and Taylor attempt to separate *"fact from fiction"* regarding software architecture. This work was motivated, in part at least, by an emerging backlash where the software architecture discipline was not seen to be delivering the expectations in terms of project control success for industrial scale applications. They note that an ADL is only the notation that helps architects model *an* aspect of the architecture. This leads them to the conclusion that a number of ADLs may be required if an architecture is to be modelled in a wide, varied and complete manner — though they give no insight as to the

practicality of such an approach. They also note the distinction between architecture and design; in their view the architecture is a layer of abstraction above the design, and is distinguished from the design by the focus being on system structure and interconnection. However their viewpoint appears to be based on the premise that traditional software design (object-oriented is cited) is focused simply on algorithms and data types. Their view is that as the architecture is refined the interconnections lose their prominence by becoming distributed across the lower level architecture's elements finally resulting in the transformation of the architecture into a design. This appears to be ending up at the point of the software program rather than the software design. It is not clear from their work if the same (or at least such extreme) issues exist for design methods based on data- and control-flow models — given ADLs generally support these concepts (as noted by Clements [16]); this would be an interesting point to consider further. In discussing the differences between architecture and design Clements notes [15] that whilst architecture is design, but not all design is architecture. The distinction between the two concepts is not entirely clear and it is for the architect to decide what is, and what is not, architecturally important. Ultimately he concludes that *"architecture is truly in the eye of the beholder"*.

Medvidovic and Taylor do however recognise, though provide no evidence, that ADLs and architecture are good at handling large scale designs constructed using off-the-shelf components, which are in turn implemented in a number of programming languages (thus supporting the concept of heterogeneity), and that the separation of computation from interaction is one of the greatest benefits. Conversely, they note that ADLs are neither good at capturing the fine grained evolution of the system, nor are they able to guarantee the properties of the implemented system — detailed code changes might have no apparent impact on the overall architecture but obviously can have a big impact on its behaviour at runtime. From this it is clear that:

- ADLs must support the appropriate level of analysis;
- There is a strong traceability from architecture to implementation;
- The architecture is maintained throughout the development process.

Following from this there is a necessity to support the ADL based descriptions of the system with behavioural information derived from the implementation if any analysis at the ADL level is not to remain abstract, and hence be of limited use. The ADL provides one model of the system; there is a need to verify, correct, and evolve that model as the implementation progresses.

The lack of early industrial success of software architecture and/or ADLs has also been noted by Medvidovic *et al.* [60] and by Di Nitto and Rosenblum [71]. The former identify the lack of specific techniques to support flexible architecture-based design and evolution. In the latter case, the interest was how middleware infrastructures influence the architectural style and how ADLs "perform" under these conditions. They note a potential for conflict between the inherent architectural assumptions of the ADL and the requirements of the middleware, and that the variation in the expressive powers of ADLs is a fundamental element of the problems regarding what constitutes an ADL. As middleware can be considered to be an implementation constraint, it seem quite understandable that there will be some conflict between the ADLs characterised as *implementation constraining* and the possibility for some independently imposed environment; however it is not clear from this work whether the same problems also exist for ADLs from the *implementation independent* group.

Beyond the basics of the languages there has been some research by Medvidovic [59] considering the role of ADLs in the field of dynamic architectures — architectures that can change at runtime. This is seen as being a difficult area with little support in the original ADLs, one of the major problems being establishing the conditions under which architectural "re-wiring" can take place. The scope of such architectural re-wiring is not clear — at one extreme there is the mapping of a constant application to a changing platform, whist at the other there is the more "modal" issue of the application fundamentally changing whilst mapped to a constant platform; these are clearly quite different problems.

As noted earlier, the original ADLs were highly specific to particular problems or domains, and specific forms of analysis, thus, as noted above by Medvidovic and Taylor [61], the modelling of multiple aspects of the system requires the use of multiple ADLs, and corresponding descriptions of the system architecture. The language Acme [33] attempted to address this diversity by providing a mechanism that allowed interchanges between different ADLs. Acme is a simple generic ADL; it provides a common interchange format for existing architecture design tools and can be used as the basis new tools. The obvious intention was to allow a number of different ADLs to be used in a development, each modelling the area/view that was its specific strength whilst maintaining a single and consistent presentation of the architecture. Whilst this is a laudable goal it does have some problems — notably the different approaches and views adopted by the separate ADLs. Di Nitto and Rosenblum [71] have described a wide variation with little overlap in the abilities of ADLs to support modelling of architectural styles induced by middleware infrastructures. From this, there appears to be little semantic commonality between ADLs amongst which interchange would be desired, thus limiting the ability of Acme to fulfill

the interchange goal. However, it is not all negative and there are some documented uses of Acme in this role, including the mapping of MetaH into Acme described by Barbacci and Weinstock [2].

From the sudden development of a (relatively) large number of independent and diverse ADLs in the 1990s, the direction of the development of new languages has changed, notably in the use of a common underlying technology — XML. There are currently a number of languages that are available, some of these have their origins in the original research, others have evolved to meet other needs and shortcomings noted in the earlier work — these include languages that allow the co-operation between different ADLs. Table A.1 details a number of the significant, currently available ADLs together with their source and a brief description of their domain of application etc. Of the original ADLs, many are still available; a number of these have migrated towards an XML based variant, and the development of others (and in some cases active maintenance) has effectively ceased — in some cases it is difficult to determine the current level of development and support. As can be seen from the table there is not a great range of commercially developed products — if a purely software perspective is adopted then a number vendors would probably contend that their UML tool set would fulfill the architectural need — the issue as to whether or not UML is an ADL will be returned to later. The few commercial examples of ADLs that do exist are often domain specific — e.g. MetaH and ControlH for avionics systems.

As observed earlier, any single ADL may only be suitable for modelling a limited number of aspects of a given architecture, and that a number of ADLs might be required to model a system satisfactorily. Recent work towards the use of XML as the underlying meta-language for an ADL should simplify interchange between the different languages. The xArch project [24] is focused on this problem. There have been other developments, some of these have tended to focus on analysis and frameworks for analysis. The view stated by Heng and Di [39] is that the current status of ADL research can be likened to the early days of research into programming languages where the focus was on features and notations rather than methodology and verification, and in the case of ADLs very little has been published in these latter fields. They suggest that the field of ADLs (as languages) is now mature enough and that the more emphasis should be placed on these other areas of concern. From this has come the development of SAM (Software Architecture Model) that is founded on two formal notations: Petri Nets and 1st order temporal logic, used to define behaviour and properties respectively. Whilst this is an interesting, orthogonal development recognising the problems of development and verification it does little to make these (especially the latter) accessible to industry practitioners.

ADL	Organisation	Notes
AADL	SAE	A language targeted at real-time embedded systems derived from MetaH [91]
UML	OMG	A “standard” software design notation[78]
Aesop	CMU	A language for “style-oriented” architectural design [32]
Armani	CMU	An ADL and an architectural design environment [66]
Leda	Malaga	An ADL supporting inheritance and dynamic reconfiguration [13]
SysML	OMG	Systems engineering derivative of the UML [76]
MARTE	OMG	Real time profile for the UML [79]
Acme	CMU	An architectural interchange language that can also be used as a simple ADL [33]
Aesop	CMU	Specification of architectures in specific styles [32]
Darwin	Imperial College	Support for highly distributed systems [57]
MetaH (& ControlH)	Honeywell	Avionics specific, adopted by the SAE [103]
Rapide	Stanford	An Executable Architecture Definition Language — modelling and simulation of dynamic behaviour [56]
SADL	SRI	Formal refinement of architectures through levels of decomposition [69]
UniCon	CMU	Glue code for connecting existing components using common interaction protocols [97]
C2 (or C2SADL)	UCI	Hierarchical systems, originally developed to describe large GUI applications [60]
Wright	CMU	Dynamic behaviour of concurrent systems — underpinned by CSP [1]
xAcme	CMU	Part of the xArch project, active tool support, update to Acme [93]
xADL	UCI	Part of the xArch project, active tool support [24]

Table A.1: Examples of ADLs

A.3 Related areas of research and similar languages

This section briefly identifies and discusses three closely related areas of research: the UML, system and architecture design languages, and the aspect-oriented approach.

A.3.1 The UML

Whilst there is clearly some dispute as to whether or not the UML is an ADL, there has been a significant amount of work over recent years into the development of the UML2 standard, [74]; a significant requirement of this revision of the language was to allow software architectures to be described using the UML.

It is certainly true that, with the advent of UML2, the language has become more capable with regard capturing and describing architectural concepts. However there are clearly some issues with the basic language demonstrated by the need to develop further languages or profiles for specific domains; some examples are:

- The Systems Modelling Language, SysML— a derived language for the systems engineering domain, supporting high-level abstractions and is not specifically software focused;
- The UML profile from Modelling and Analysis of Real Time and Embedded systems, MARTE — an attempt to address the issues associated with the structure and implementation of real-time and embedded systems, including the issues of hardware-software partitioning;
- EAST-ADL, described by Debruyne *et al.* [26] is a development firmly targeted at the (European) automotive industry.

The first two of these originate in the OMG, the latter has been developed by the EAST-EAA project sponsored by ITEA (Information Technology for European Advancement).

A.3.2 System and Architecture Design Languages

In the sections above, the focus has been on languages that are either generally considered to be ADLs or are in some way related to the UML. In parallel with their development a number of other languages have been developed where the designers have chosen to distinguish, in some cases quite deliberately, their offerings from ADLs. Examining these suggests that, in some cases, they possess a number of the properties that would make them appropriate from consideration in the description of real-time embedded systems — the work relating to a number of these is briefly described and discussed below.

A major field in which these alternative ADLs can be found is that of hardware–software codesign. ADLs here range from those developed specifically to address the issues associated with small-scale (physically) System-on-Chip (SoC) designs, through to those addressing the issues associated with significant systems built from heterogeneous components. A number of languages have been developed in this field, amongst these three that could be considered as being ADL-like are Ptolemy [9, 47], SystemC [81], and OpenJ [106]. Ptolemy has its roots in the development of real-time embedded systems, originally focused on signal processing applications. The term ADL is sometimes used to describe Ptolemy — but in this case the authors intend it to mean an "Architecture Design Language".[2] Both SystemC and OpenJ are sometimes classified as "System *Design* Languages" — in the case of both of these the systems in question are very small-scale (physically), usually being limited to the SoC problem. In the case of SystemC this is re-enforced by its standardisation [46]; from this it is clear that this language is targeted at the hardware domain, albeit operating at a higher level of abstraction than that supported by, say, VHDL. Both Ptolemy and SystemC are examined in more detail later in this appendix.

In addition to these there is another ADL-like language developed for Command and Control systems; this is described by Ravindran [87]. He chooses to describe his language, again, as a "System Design Language", however in this case the focus is large scale distributed systems. Whilst he acknowledges the existence of a number of ADLs, it appears that one of the key reasons for creating a new language in this case was the lack of appropriate abstractions for the properties pertinent to command and control systems[3] — his system description language is designed to address precisely these properties, e.g. quality of service.

A.4 Application to real-time and embedded systems

Having looked at the state of research into ADLs in relatively general terms it is now appropriate to consider their use in relation to the construction of embedded real-time systems.

As has been noted above, the architectural description of an embedded real-time system must encompass both hardware and software aspects; unfortunately there are few ADLs that provide this in a clear manner, examples of these include AADL [91] and Ptolemy [9]. More unfortunately, there is a general problem that languages that consider the hardware–software issue are generally application domain specific, and are in the *implementation*

[2] How this is differentiated from an Architecture Description Language is discussed in a later section.

[3] The domain specificity issue again.

constraining group. No domain encompasses the general case for embedded real-time systems; Bass *et al.* [3] note that only half the ADLs they surveyed could support hard real-time characteristics such as deadlines, and only a few could handle softer real-time issues such as priority;[4] though it should be noted that this survey only relates to the early ADLs. Generally the analysis and modelling of real-time systems is limited to finite state analysis supported by notations such as CSP [42], the ability to perform more continuous analysis, e.g. schedulability, is seen as being in the domain of add-ons for most languages. The XML based approach is clearly extensible — but it is not clear how much effort would be required to develop the extensions necessary to support real-time aspects in a fully integrated or "central" manner.

The development of languages such as AADL [91] and MARTE [79] are more modern examples of ADLs where there has been some more in-depth consideration of issues associated with real-time systems. The more flexible of these appears to be MARTE which brings time more to the fore and allows it to be treated as more than a simple attribute of a functional element; however it remains to be seen as to whether this profile will be applied effectively for real-time systems. Both AADL and MARTE are fundamentally targeted at the implementation of systems, at the higher levels of abstraction languages such as SysML [76], the other recent language from the OMG, do not address time or real-time issues — though given the common UML heritage of MARTE and SysML there might be an opportunity to incorporate some of the MARTE sub-profiles into SysML.

A.5 A closer look at a number of languages

In the following sections a number of ADLs are considered in respect of their capabilities both specifically with respect to temporal issues and more generally; before considering the language the general approach to their comparison is outlined.

A.5.1 Selection and comparison approach

In taking a closer look at a number of the languages it is clearly impractical to consider all of the ADLs, and to a large extent this is also unnecessary in that surveys by a number of authors [102, 16, 62] have already attempted to characterise and classify ADLs. Here the languages to be examined are selected from the overall pool on the basis of two simple criteria:

- They should be able to support real-time systems in the most general sense — languages which are specific to unrelated application domains are excluded;

[4] Whilst priority might not be considered an architectural feature in itself it is a necessary attribute if certain types of analysis are to be performed at the architectural level — e.g. response time.

- They should not demand a particular implementation technology or approach, for example they should not prescribe a software-only solution, demand the use of a specific programming language, or rely on a specific middleware infrastructure.

Additionally, languages which are intended to have a broad appeal are chosen in preference to those developed for particular niche interests, or particular domains.

Using these criteria the languages to be examined are: AADL (with MetaH), SysML, the MARTE profile, Acme (and xAcme), Ptolemy, SystemC and xADL — given the inclusion of SysML and Marte, the UML itself has been excluded. These cover a broad spectrum of approaches and include examples from both the commercial and academic environments.

Having selected the languages the next step is to define the criteria for their comparison, for this seven are chosen; these are:

1. The modelling of structural elements;
2. The modelling of constraints;
3. The modelling of properties or attributes;
4. The scope to allow heterogeneous implementation solutions;
5. The extensibility of the underlying description;
6. Availability of tool support;
7. Examples of real-world usage.

In the application of these criteria there is an overall focus on temporal issues. Clearly these criteria are note entirely objective, thus the comparison will inevitable be somewhat subjective.

Finally, in order to give a better understanding of the general appearance of these languages together with the forms of information they attempt to capture, a simple example "system" is presented using the notations of two contrasting early ADLs: Acme, an *implementation independent* language that gives connectors "first-class" status; and MetaH, an *implementation constraining* language that treats connectors as in-line elements.

The "system" described in this case is that of simple producer and consumer processes connected in such a way that the periodic producer is never blocked and the aperiodic consumer waits for data to be made available by the producer. Whilst simple this arrangement is a common pattern in embedded real-time systems. The WCETs of the producer

and consumer are 100 μs and 200 μs respectively; the period of the system is 400 μs. At first sight perhaps WCET does not appear to be a good "architectural" choice, it being very low level, however its choice is somewhat dictated by the limitations of the languages and their capabilities with respect to time.

A.5.2 Description and appraisal of the languages

SysML

The emergence of SysML [76] is perhaps indicative of the unsuitability of the UML as an ADL in its own right. To some extent it is the view that the UML is a software (design) centric notation that works against it being accepted more widely as a UML — the plethora of diagram types and different aspects of the model make it a complicated language to use consistently, and difficult to use when describing aspects of the system that are either abstract or relate to implementations that are not based in software. SysML is intended to overcome the major concern of "systems' engineers" — namely that the UML is a software design notation.

In considering the criteria set out above, clearly SysML does not demand specific forms of implementation and, having it origins in the UML, it supports a wide variety of elements, properties and constraints. Extensibility is possible in so far as that allowed by the UML itself — stereotypes being an obvious approach.[5] A large number of vendors support SysML within their UML tool offerings. Perhaps the one are where SysML is not strong is in its handling of time in general and real-time issues in particular.

SysML is widely used in industry, particularly in the defence sector where it has been used to "implement" the DoDAF and MoDAF frameworks.

MARTE

The UML profile for Modelling and Analysis for Real Time and Embedded systems [79], or MARTE for short. is an emerging use of the UML intended to specifically address the issues of real-time and embedded systems, as such it provides support for the specification, design and verification/validation stages. It actually replaces an earlier profile for Schedulability, Performance and Time, described in [75]. Unlike SysML, which is defined as a language in its own right, this is defined as a profile (an extension) of the UML, its goal being to extend the UML to support model driven development of real-time and embedded systems. It does not introduce any new core language features but standardises how the ones already present in the UML should be used for real-time and embedded systems.

[5] Though whether these are really an extension rather than a customisation is open to debate.

In [79] the primary benefits of this profile are stated as being:

- *Providing a common way of modeling both hardware and software aspects of a RTES in order to improve communication between developers.*
- *Enabling interoperability between development tools used for specification, design, verification, code generation, etc.*
- *Fostering the construction of models that may be used to make quantitative predictions regarding real-time and embedded features of systems taking into account both hardware and software characteristics.*

MARTE is wide ranging, and specifically addresses a number of the topics often associated with real-time and embedded systems. MARTE addresses these issues through a number of sub-profiles each of which addresses a different concern. The sub-profiles are organised in three groups and address a number of areas; these are:

- MARTE foundations
  - NFP profile — non-functional property modelling
  - Time profile — time modelling
  - GRM profile — generic resource modelling
  - Alloc profile – allocation modelling
- MARTE design model
  - GCM profile — generic component modelling
  - HLAM profile — high-level application modelling
  - DRM — detailed resource modelling; consisting of:
    SRM profile — software resource modelling
    HRM profile — hardware resource modelling
- RTEA (Real-time and Embedded Analysing)
  - GQAM profile — generic quantitative analysis modelling
  - SAM profile — schedulability analysis modelling
  - PAM profile — performance analysis modelling

Perhaps the most interesting aspect of MARTE, from an RTES viewpoint, is the treatment of time; here, rather than adopt the usual approach of simply providing temporal information in the form of attributes to other elements of the description time is given a much more prominent role. In essence two distinct models of time are supported: chronometric and logical. The former relates to real clock which "ticks" at a particular rate, the rate is subject to various characteristics such as drift and jitter. The logical clocks are ones which can be defined as a sequence of "abstract" ticks.

As with SysML, originating in the UML, it has a wide variety of elements, properties and constraints, and there are usual UML approaches to extension. Being a very recent development only time will tell whether or not the language is:

- Adequately supported by suitable tools;[6]
- The language is adopted by practitioners.

Clearly the real strength of MARTE in comparison to other languages (if it is adequately supported) appears to be that of its approach to real-time, however the final details of tis support are not yet clear as the "standardisation" process is not yet complete.

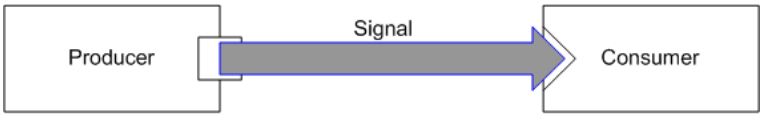
xAcme (and Acme)

xAcme [93] is derived from an early ADL Acme [33]; as noted earlier, this was originally developed to enable interchange between other ADLs, hence it is rather open in the concepts that it supports, and is of the *implementation independent* group. There have been a number of studies to show how Acme could be used to allow different ADLs to interchange information, e.g. between Wright and Rapide in [34], and from MetaH to Acme in [2].

xAcme is one of a pair of ADLs built on a standard architectural infrastructure (xArch [24]), which uses XML schema for the underlying description. xArch is a vehicle for representing the structure of the architecture, thus extensions are required to include higher level concepts such as types, styles, constraints etc. xAcme provides one possible set of extensions to this basic capability. The concepts of components, connectors, the links between them, collections (configurations) and architecture itself are all part of xArch; xAcme extends this structural core to include constraints, properties, multiple representation, types and families.

Tool support is available (in an early form) in Acmestudio, a Windows based GUI tool originally developed to support Acme. In addition, the use of XML as the underlying

[6] There should be few impediments to this as MARTE is "simply" a profile of the UML.




```
System simple_pc = {
        Component producer =   { Port result;
                                 Properties { WCET : Integer = 100;
                                              periodic : boolean = true }}
        Component consumer =   { Port seed;
                                 Properties { WCET : integer = 200;
                                              aperiodic : boolean = true }}
        Connector signal =     { Roles {send, receive};
                                 Properties { capacity : integer = 1 }}
        Attachments = { producer.result to signal.send;
                        consumer.seed to signal.receive }
        Properties =   { period : integer = 400 }
        }
```

Figure A.1: Acme solution to the producer–consumer architecture example

descriptive form opens up the possibility of using other commercial tools to create and manipulate the descriptions, and the ability to extend the language to accommodate additional concepts and information.

Being a relatively new ADL there is little example of its use to describe real architectures, however, as noted above, its predecessor Acme has been used to interchange architectures between other ADLs.

A possible Acme solution is shown in Figure A.1, the use of Acme rather than xAcme here is to provide a simpler textual representation of the system, the XML used for xAcme is rater verbose. Both the graphical and textual notations are given.[7] All entities, the two components and the connector, are explicit in the description. Any entity and the system itself may have properties associated with them; the precise properties that may be used are not defined by the language, it simply provides a general syntax for the style in which they may be added. Processing of the additional data is the task of additional bespoke tools.

Being a small example this only makes limited use of the features of the language; other language constructs include ones that allow design fragments to be reused in different systems.

AADL (and MetaH)

AADL is one of the few (perhaps only) examples of commercially supported ADLs that do not have their roots in the UML; in this case, as noted earlier, its origins are in MetaH.

[7]The syntax used follows that given in [33].

MetaH [103] was an early ADL, again a commercial one (from Honeywell), which was focused on the avionics domain. MetaH did not explicitly model connectors as "first-class" entities, and it was an *implementation constraining* language. It supported the modelling of properties to allow schedulability, security and reliability analyses, but only within a relatively narrow execution framework, and only, as indicated in the example given in later, for components; in MetaH components always map to a thread of execution. Unlike most ADLs it recognises explicitly the need to map the software components to some hardware platform (processor), this is achieved through the use of configurations (applications).

AADL builds on MetaH to address some of the original deficiencies, it:

- Provides all of the elements to be found in the original ADLs: components, connectors, configurations and constraints, giving first-class status to connectors;
- Provides appropriate time-related properties for both the connectors and a number of the components — additionally it supports more system wide concepts such as end-to-end timings;
- Attempts to capture more of the behaviour of the system through the introduction of concepts such as modes;
- Imposes no constraints on how the components are implemented, MetaH assumed that they would be implemented in Ada or C, AADL acknowledges that there are many approaches including software languages, specialised modelling languages (e.g. Matlab/Simulink), or hardware description languages (e.g. VHDL).

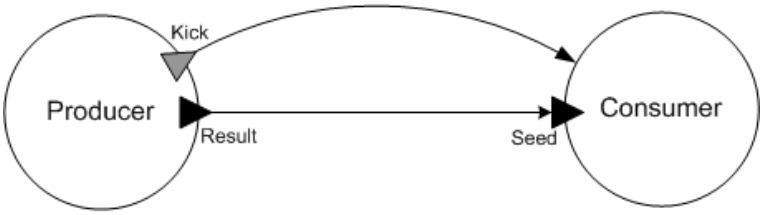
In short it is quite different from, and more capable than MetaH. Perhaps its only real limitation is that it makes a strong distinction between the platform and the application, assuming that the platform is a given, having been defined prior to the AADL "layer" of the architecture is developed.

A possible MetaH solution is given in Figure A.2, again both the graphical and textual forms are given.[8] As is clear from the figure that the connections between the two processes is only apparent by the association of input and output ports and the connection of the event. The need for two interfaces between the components stems from the limited interactions that are possible between components. The absence of an explicit entity (connector) between the processes means that attributes may only be associated with the

[8]The syntax used follows that given in [103].

processes (and their interfaces) — identifying the complete interaction between the processes can only be accomplished by examination of both processes. The attributes (in this case defined by the language) provide a mechanism to allow other elements of the toolset to perform a number of analyses.

Whilst this small example makes use of a number of the constructs available in MetaH there are others that support the description of platform components, and the mapping of processes to these components.




```
periodic process Producer is
      Result   : out port Integer;
      Kick     : out event;
end Producer;

aperiodic process Consumer is
      Seed     : in port Integer;
end Consumer;

periodic process implementation Producer.Imp is
attributes
      self'sourcetime := 100us;
end Producer.Imp;

aperiodic process implementation Consumer.Imp is
attributes
      self'sourcetime := 200us;
end Consumer.Imp;

macro M is  -- presents no interface
end M;

macro implementation M.Imp is
      PA : periodic process Producer.Imp;
      PB : aperiodic process Consumer.Imp;
      PA.Kick is signal event;
connections
      PB.Seed <<- PA.Result; -- produced when ready
      PB <- PA.Kick;
attributes
      PA'period := 400us;
end M.Imp;
```

Figure A.2: MetaH solution to the producer–consumer architecture example

Ptolemy

Ptolemy II [47] is the latest development in the Ptolemy "family"; it is derived from Ptolemy Classic [9]. Its focus is on embedded software and the embedded systems in which it executes, and is, as noted earlier, described by its developers as an Architecture *Design* Language. They distinguish this from an Architecture *Description* Language on the grounds that its goal is to produce coherent software architectures by imposing structure on the interactions between the elements, whereas the goal of a "traditional" ADL is to describe interactions between essentially independent, and possibly pre-existing, elements. The structure is constrained, at least in part, by the model(s) of computation chosen for the system; the developers consider that the choice of computation model strongly affects the quality of the system design.

Given its use of specific computation models it is in essence an *implementation constraining* language; a number of computation models are provided (Communicating Sequential Processes, Continuous Time, Discrete Events, Process Networks and a number of others) giving some degree of flexibility, but only from a relatively narrow perspective. It is not designed to support the descriptions of architectures created from elements defined in diverse manners, but relies on the framework it imposes to ensure the architecture is consistent. Tool support is extensive; however extensibility is limited to that possible by modifying the source code of the associated tools — this is outside of the scope of extensibility as defined for this comparison. There are some examples of real world usage, including the use of the Giotto domain to describe some robotic systems [41].

SystemC

SystemC, or the Open SystemC Initiative [81] is a collaboration between a number of industrial organisations whose normal business is focused on hardware component and system manufacture. It attempts to address the problems associated with the design and implementation of Systems-on-Chip (SoC) where the implementation is mixture of hardware and software (they believe that the latter accounting for 50-90% of the functionality); part of the aim is to provide a common environment in which "building blocks" may be easily incorporated across a range of products. It is based on the C++ language, using this for modelling both the hardware and software aspects of the system, thus allowing a more flexible solution to the partitioning between hardware and software. In terms of the comparison criteria it does not support (directly) the concepts of component, connector and compositions. Constraints and properties would be those imposed by the language

and the interfaces provided to the C++ classes. At the language level there is little scope for extensibility to address novel concepts, but clearly there is tool support.

Whilst originally intended to support mixed hardware/software solutions it now, with the standardisation to IEEE-1666, appears that the language has become more hardware centric and it seems doubtful that the language is capable of describing the architecture of large complex software intensive systems.

xADL

xADL[9] [23] is another ADL based on the xArch foundation described above, and is again *implementation independent*. Like xAcme it extends the core xArch, this time to include support for structure and types, mappings to implementations, and evolution. The developers of xADL, in [24], refer to it as an infrastructure for development of XML-based ADLs, but this term seems to be more appropriately applied to the xArch technology rather than to xADL itself. However xADL is designed to be extensible, and it has been shown that small extension schema can be used to extend xADL to express the architectural concerns of other ADLs. This extension capability has been demonstrated, in [21] for the product line ADLs Koala and Mae.

The implementation mappings are not bound directly to a specific platform or language; a two level approach is adopted where one level is abstract: it defines where the data should go in the architecture, but not what it should be. The second level is concrete: it defines the data that is required for a particular platform or programming language. A Java implementation XML schema is provided as part of xADL. To-date a purely software perspective has been taken, and no consideration has been given to the hardware mapping or concurrency issues.

The general style of the language is adopted from an earlier ADL — C2; and tool support for xADL (ArchStudio4) — these tools are based on the Eclipse framework.

In addition to the examples of modelling other ADLs noted above, the language has been used to describe an AWACS system, a system of in excess of 100 components with complex interconnections, and has been selected to be used in the definition of the architecture for a future Mars Lander, both identified [23]. Since these references there has been little new published on xADL.

[9]More correctly the reference should be to xADL 2.0, xADL 1.n and XADL are other examples of XML-based ADLs.

A.6 Comparison summary

The comments in the sections above regarding the "performance" against the criteria are summarised in Table A.2 below.

ADL	Entities	Constraints	Properties	Heterogeneity	Extensibility	Tool support	Real use
SysML	✔	✔	✔	✔	✔	✔	✔
MARTE	✔	✔	✔	✔	✔	?	×
AADL	✔	✔	✔	✔	?	✔	✔
Ptolemy	×	inherent		✔	×	✔	✔
SystemC	×	inherent		✔	×	✔	?
Acme	✔	✔	✔	✔	✔	✔	?
xADL	✔	✔	✔	✔	✔	✔	✔

Table A.2: Comparison summary

Notes:

1. The use of the '?' in the table indicates that there is a lack of clarity in the literature of the precise position with regard to these criteria, as is the case with MARTE;
2. In the case of Ptolemy and SystemC support for properties and constraints is provided but are inherent in the computation model chosen;
3. MetaH is considered as part of AADL.

Whilst this is a far from exhaustive survey it identifies the basic capabilities across a range of languages. With the exception of Ptolemy and SystemC the languages are of broadly similar capabilities, with perhaps the most distinguishing feature being their level of real usage, the industrially and commercially based languages being, unsurprisingly, dominant in this regard. Ptolemy and SystemC can be seen as being more focused on their particular domains ("system" implementation), and therefore perhaps less suited to a general architectural role.

A.7 Summary

The study and development of ADLs, and the related languages, parallels that of software architecture more generally — both have their origins in the realisation that simple design methods were inadequate for the successful development of large software-intensive systems. Over the past 20 years or so, many languages have been developed, largely from

an academic perspective. It has been common for these languages to be domain specific, and for them to lack the necessary breadth of usage and support to bring them into mainstream use. In general, these languages did not support non-functional properties; at best they would address some specific properties for the domain to which they were targeted, real-time aspects were only, at best, weakly addressed — the focus being more on the structure and function of the system rather than its behaviour.

Whilst most of the early ADLs did not evolve to mainstream use, there is perhaps one notable exception — MetaH is the basis for AADL, and this latter language has certainly found it way into industrial use. As noted above one of the fundamental reasons why most of the languages failed to make this transition was that there was inadequate tool support. Most of the ADLs were developed independently of the tool vendors and in an era where there were no mainstream tool frameworks or infrastructures, hence their originators needed to develop and maintain all aspects of the tools, resulting in tools that had little prospect of long term development and support.

In parallel with the development of ADLs in academia, the OMG has developed the UML, which, unlike these other ADLs, has received widespread support across both the tool developers and end-users in industry. Whilst its initial adoption was focused on the design (*cf.* architecture) of certain types of software system, it has been continued to be developed by the OMG to a point where, with the emergence of SysML, it is now used to model the architectures of the higher level non-software aspects of large systems using frameworks such as DoDAF and MoDAF. More generally there is a trend of languages needing to have some relationship to UML, as an example: AADL itself is not a UML derivative, however one of the annexes of its standard defines a UML profile allowing AADL architectures to be developed using UML tools.

Looking to the future, the creation of MARTE is an interesting development. This is perhaps the first language to consider time as an independent aspect of the architecture, and its explicit recognition of the need to address non-functional properties is similarly original; however it is not entirely clear how independent this view of time will be in practice. Whilst its definition is due to be finalised in the near future, it remains to be seen as to what degree (and how quickly) it becomes supported by tool vendors, and hence is exploited by users in the wider community.

It appears that there is a degree of inevitability that these UML based languages will become the *defacto* standards in industry, to the detriment of perhaps “better” solutions developed through more fundamental research. There appears to be an analogy here with programming languages where C is commonly used for real-time embedded systems, but

is widely accepted to be poor for the development and implementation of large-scale or high-dependability systems.

A final observation — whilst the original ADLs were relatively simple, comprising the four basic elements, with the more recent languages such as AADL, SysML and MARTE, there is a trend towards larger and more complex languages. The reasons for this trend are not clear, however it is not unreasonable to suggest that in these cases the need to accommodate a wide base of user "opinion"[10] has caused this expansion. Whether or not such languages can result in better system solutions remains to be seen.

[10]All of these languages have been through some form of standardisation process and that appears to require that all diverse views are accommodated.

Abbreviations

AADL	Originally Avionics Architecture Description Language, now Architecture Analysis & Design Language
ADL	Architecture Description/Definition/Design Language — generally, and throughout this report, the 'D' refers to Description
ADML	Architecture Description Markup Language
AML	Architecture Meta Language
ASIC	Application Specific Integrated Circuit
ATM	Automated Teller Machine (a bank cash point machine)
AWACS	Airborne Warning and Control System
BIT	Built In Test
BCET	Best Case Execution Time
C^2	Command and control
CMU	Carnegie Mellon University
COTS	Commercial Off The Shelf
CSP	Communicating Sequential Processes
DTD	Document Type Definition
DoD	Department of Defense
DoDAF	Department of Defense Architectural Framework
FPGA	Field Programmable Gate Array
HLA	High Level Architecture (for distributed simulations)

IEEE	Institute of Electrical and Electronic Engineers
ITEA	Information Technology for European Advancement
MARTE	Modeling and Analysis of Real-time and Embedded systems
MDA	Model Driven Architecture
MIL	Module interconnection Language
MoD	Ministry of Defence
MODAF	Ministry of Defence Architectural Framework
OCL	Object Constraint Language
OMG	Object Management Group
OO	Object Orientated
PL	Programming Language
RMA	Rate Monotonic Analysis
RTA	Response Time Analysis
RTES	Response Time Embedded System
SA	Simulated Annealing
SAE	Society of Automotive Engineers
SoC	System-on-Chip
TAR	Test Accuracy Ratio
UCI	University of California at Irvine
UML	The Unified Modeling Language
VHDL	VHISC (Very High Speed Integrated Circuit) Hardware Description Language
VLSI	Very Large Scale Integration
WCET	Worst Case Execution Time
XML	Extensible Markup Language

Bibliography

[1] Robert Allen and David Garlan. Formalising architectural connection. In *Proceedings of the Sixteenth International Conference on Software Engineering*, 1994.

[2] Mario R. Barbacci and Charles B. Weinstock. Mapping MetaH into Acme. Technical report, Software Engineering Institute, Carnegie Mellon University, 1996.

[3] Len Bass, Paul Clements, and Rick Kazman. *Software architecture in practice*. SEI series in software engineering. Addison Wesley, 1998.

[4] Len Bass, Paul Clements, and Rick Kazman. *Software architecture in practice*. SEI series in software engineering. Addison Wesley, 2nd edition, 2003.

[5] Iain Bate, John McDermid, and Peter Nightingale. Establishing timing requirements for control loops in real-time systems. *Microprocessors and Microsystems*, 2003.

[6] Gordon Baxter, Alan Burns, and Kenneth Tan. Evaluating timebands as a tool for structuring the design of socio-technical system. In *Proceedings of the International Conference on Contemporary Ergonomics (CE2007)*, 2007.

[7] Claudio Bettini, X. Sean Wang, and Sushil Jajodia. A general framework for time granularity and its application to temporal reasoning. *Annals of Mathematics and Artificial Intelligence*, 1996.

[8] G. Booch. The irrelevance of architecture. *IEEE Software*, May – June 2007.

[9] Joseph Buck, Soonhoi Ha, Edward A. Lee, and David G. Messerschmitt. Ptolemy: A framework for simulating and prototyping heterogeneous systems. *International Journal of Computer Simulation*, 1992.

[10] Alan Burns and Gordon Baxter. Time bands in system structure. In *Structure for dependability: Computer based systems from an interdisciplinary perspective*. Springer, 2006.

[11] Alan Burns, Gordon Baxter, Colin Fidge, and Ian Hayes. Modelling temporal behaviour in complex socio-technical systems. Technical Report YCS-2005-390, Department of Computer Science, University of York, 2005.

[12] Alan Burns and Andy Wellings. Delivering real-time behaviour. In *Domain Modelling and the Duration Calculus*, LNCS 4710. Springer-Verlag, 2007.

[13] Carlos Canal, Ernesto Pimentel, and José M. Troya. Specification and refinement of dynamic software architectures. In *WICSA1: Proceedings of the TC2 First Working IFIP Conference on Software Architecture (WICSA1)*, 1999.

[14] E. Ciapessoni, E. Corsetti, A Montanari, and P. San Pietro. Embedding time granularity in a logical specification language for synchronous real-time systems. *Science of Computer Programming*, 20, 1993.

[15] Paul Clements, Felix Bachmann, Len Bass, David Garlan, James Ivers, Reed Little, Robert Nord, and Judith Stafford. *Documenting software architectures, Views and Beyond*. SEI series in software engineering. Addison Wesley, 2003.

[16] Paul C. Clements. A survey of architecture description languages. In *Proceedings of the Eighth International Workshop on Software Specification and Design*, 1996.

[17] James Clifford and Ahobala Rao. A simple, general structure for temporal domains. In *Proceedings of Temporal Aspects of Infprmation Systems, Sophia-Antipolos, France*, 1987.

[18] Derek Coleman, Grady Booch, David Garlan, Sridhar Iyengar, Cris Kobryn, and Victoria Stavridou. Panel discussion: Is UML an Architectural Description Language? OOPLSA Conference – OOPSLA'99, 1999.

[19] Alan Paul Conn. Time affordances: The time factor in diagnostic usability heuristics. In *Proceedings of the Conference on Human Factors in Computing Systems CHI'95*, 1995.

[20] E. Corsetti, A Montanari, and E. Ratto. Dealing with different time granularities in formal specification of real-time systems. *Journal of Real-Time Systems*, 3(2), 1991.

[21] Eric M. Dashofy and Andre van der Hoek. Representing product family architectures in an extensible architecture description language. In *Proceedings of the International Conference on Product Family Engineering (PFE-4)*, 2001.

[22] Eric M. Dashofy, Andre van der Hoek, and Richard N. Taylor. A highly-extensible, XML-based architecture description language. Technical report, Dept. of Information and Computer Science, University of California, Irvine, 2000.

[23] Eric M. Dashofy, Andre van der Hoek, and Richard N. Taylor. A highly extensible, XML-based architecture description language. In *Proceedings of the Working IEEE/IFIP Conference on Software Architectures (WICSA 2001)*, 2001.

[24] Eric M. Dashofy, Andre van der Hoek, and Richard N. Taylor. An infrastructure for the rapid development of XML-based architecture description languages. In *Proceedings of the 24th International Conference on Software Engineering (ICSE'02)*, 2002.

[25] T. Dean. Using temporal hierarchies to efficiently maintain large temporal databases. *Journal of the Association for Computing Machinery*, 1989.

[26] Vincent Debruyne, Françoise Simonot-Lion, and Yvon Trinquet. EAST-ADL an architecture description language. In *IFIP TC-2 Workshop on Architecture Description Languages (WADL), World Computer Congress*, 2004.

[27] Mark Denford, Jogn Leaney, and Tim O'Neill. Non-functional refinement of computer based systems architecture. In *Proceedings of 11th IEEE Conference and workshop on the engineering of computer based systems ECBS'04*, 2004.

[28] Frank DeRemer and Hans H. Kron. Programming-in-the-large versus programming-in-the-small. *IEEE Transactions on Software Engineering*, 1976.

[29] E. W. Dijkstra. The structure of 'T.H.E.' multiprogramming system. *Communications of the ACM*, 18(8), August 1968.

[30] DOD. Test method standard microcircuits. Technical Report MIL-STD-883G, United States Department of Defense, 2006.

[31] DoD. The department of defense architecture framework version 1.5. Technical report, US Department of Defense, 2007.

[32] David Garlan, Robert J. Allen, and J. Ockerbloom. Exploiting style in architectural design environments. In *Proceedings of the ACM SIGSOFT International Symposium on the Foundations of Software Engineering*, 1994.

[33] David Garlan, R. Monroe, and D. Wile. Acme: An architecture description interchange language. In *Proceedings of CASCON'97*, 1997.

[34] David Garlan and Zhenya Wang. A case study in software architecture interchange. In *Proceedings of Coordination'99*, 1999.

[35] Akram Ghulam. Loitering munitions: Revolutionising indirect fire support. *RUSI Defence Systems*, 10(3), 2008.

[36] Ian Gorton. *Essential Software Architecture*. Springer, 2006.

[37] Open Group. The architecture description markup language (ADML). Technical report, The Open Group, 2000.

[38] D. Harel. Statecharts: A visual formalism for complex systems. *Science of computer programming*, 8(3), June 1987.

[39] Xudong He and Yi Deng. A framework for developing and analyzing software architecture specifications in SAM. *The Computer Journal*, 45(1), 2002.

[40] Kathryn L. Heninger, John W. Kallander, John E. Shore, and David L. Parnas. Software requirements for the A7e aircraft. Technical Report MR-3876, Naval Research Laboratory, Washington DC, 1978.

[41] Thomas A. Henzinger, Benjamin Horowitz, and Christoph Meyer Kirch. Embedded control systems development with Giotto. In *Proceedings of the ACM SIGPLAN Workshop on Languages, Compilers and Tools for Embedded Systems (LCTES)*, 2001.

[42] C. A. R. Hoare. *Communicating Sequential Processes*. Prentice Hall, 1985.

[43] C. Michael Holloway and Ricky W. Butler. Impediments to industrial use of formal methods. *IEEE Computer*, April 1996.

[44] S. G. Hutchesson and N. Hayes. Technology transfer and certification issues in safety critical real time systems. In *Digest of the IEE Colloquium on Real-Time Systems*, 1998.

[45] IEEE. IEEE-1471 — Recommended practice for architectural description of software-intensive systems. Technical report, Institute of Electrical and Electronic Engineers, 2000.

[46] IEEE. IEEE Std 1666 - IEEE standard SystemC language reference manual. Technical report, IEEE, 2006.

[47] John Davis II, Christopher Hylands, Jörn Janneck, Edward A. Lee, Jie Liu, Xiaojun Liu, Steve Nuendorffer, Sonia Sachs, Mary Stewart, Kees Vissers, Paul Whitaker, and

Yuhong Xiong. An overview of the Ptolemy project. Technical report, Department of Electrical Engineering and Computer Science, University of California, 2001.

[48] F. Jahanian and A. Mok. Safety analysis of timing properties in real-time systems. *IEEE Transactions on Software Engineering*, 1986.

[49] F. Jahanian and A. Mok. Modechart: A specification language for real-time systems. *IEEE Transactions on Software Engineering*, 20(12), December 1994.

[50] Mo Jamshidi. Systems of systems engineering — a definition. In *Proceedings of the IEEE Conference on Systems Man and Cybernetics SMC2005*, 2005.

[51] Geraint Jones and Michael Goldsmith, editors. *Programming in Occam 2*. Prentice Hall, 2nd edition, 1988.

[52] Philippe Kruchten, Henk Obbink, and Judith Stafford. The past, present and future of software architetcure. *IEEE Software*, March – April 2006.

[53] Philippe B. Kruchten. The 4+1 view model of architecture. *IEEE Software*, 1995.

[54] Robert Levine. *A Geography of Time — The Temporal Misadventures of a Social Psychologist* or *How Every Culture Keeps Time Just a Little Bit Differently*. Oneworld Publications, 2006.

[55] David C. Luckham, John J. Kenney, Lary M. Augustin, James Vera, Doug Bryan, and Walter Mann. Specification and analysis of system architecture using Rapide. *IEEE Transactions on Software Engineering*, 21(4), April 1995.

[56] David C. Luckham and James Vera. An event based architecture definition language. *IEEE Transactions on Software Engineering*, 21(9), September 1995.

[57] Jeff Magee, Naranker Dulay, Susan Eisenbach, and Jeff Kramer. Specifying distributed software architectures. In *Proceedings of Fifth European Software Engineering Conference (ESEC'95)*, 1995.

[58] G. Mealy. A method for synthesizing sequential circuits. *The Bell System Technical Journal*, 1955.

[59] Nenad Medvidovic. ADLs and dynamic architecture change. In *Proceedings of the ACM SIGSOFT'96 Workshop*, 1996.

[60] Nenad Medvidovic, David E. Rosenblum, and Richard N. Taylor. A language and environment for architecture-based software development and evolution. In *Proceedings of the 21st International Conference on Software Engineering (ICSE'99)*, 1999.

[61] Nenad Medvidovic and Richard N. Taylor. Separating fact from fiction in software architecture. In *Proceedings of the Third Software Architecture Workshop (ISAW-3)*, 1998.

[62] Nenad Medvidovic and Richard N. Taylor. A classification and comparison framework for software architecture description languages. *IEEE Transactions on Software Engineering*, 26(1), January 2000.

[63] Nikunj R. Metha, Nenad Medvidovic, and Sandeep Phadke. Towards a taxonomy of software connectors. In *Proceedings of the 22nd International Conference on Software Engineering (ICSE 2000)*, 2000.

[64] MOD. The joint services manual of the law of armed conflict. Technical Report JSP-383, UK Ministry of Defence, 2007.

[65] MOD. The MOD architecture framework version 1.2. Technical report, UK Ministry of Defence, 2008.

[66] R. T. Monroe. *Rapid development of Custom Software architecture design environments*. PhD thesis, CMU-CS-99-161, Dept. of Computer Science, Carnegie Mellon University, 1999.

[67] Angelo Montanari, Enrico Maim, Emanuele Ciapessoni, and Elena Ratto. Dealing with time granularity in formal in the event calculus. In *Proceedings of the international conference on fifth generation computing systems*, 1992.

[68] E. F. Moore. Gedanken — experiments on sequential machines. *Automata Studies, Annals of Mathematics Studies*, 1956.

[69] M. Moriconi, X. Qian, and R. A. Riemenschneider. Correct architectural refinement. *IEEE Transactions on Software Engineering*, 21(4), April 1995.

[70] Allen Newell. *Unified Theories of Cognition*. Harvard University Press, 1990.

[71] Elisabetta Di Nitto and David E. Rosenblum. Exploiting ADLs to specify architectural styles induced by middleware infrastructures. In *Proceedings of the 21st International Conference on Software Engineering (ICSE'99)*, 1999.

[72] Christer Norström, Kristian Sandström, Jukka Mäki-Turja, Mikael Gustafsson, and Nils-Erik Bankestad. Experiences from introducing state-of-the-art real-time techniques in the automotive industry. In *Proceedings of the Eighth IEEE International and Workshop on the Engineering of Computer-Based Systems, IEEE International Conference on the*. IEEE Computer Society, 2001.

[73] H. Nyquist. Certain topics in telegraph transmission theory. *Transactions of the A. I. E. E*, February 1928.

[74] OMG. Unified modelling language (UML2). Technical report, Object Management Group, 2004.

[75] OMG. UML profile for schedulability, performance and time (v1.1). Technical report, Object Management Group, 2005.

[76] OMG. Systems modelling language (SysML). Technical report, Object Management Group, 2007.

[77] OMG. Unified modelling language, version 2.1.1. Technical report, Object Management Group, 2007.

[78] OMG. Unified modelling language, version 2.1.2. Technical report, Object Management Group, 2007.

[79] OMG. A UML profile for MARTE: Modeling and analysis of real-time embedded systems (Beta2). Technical report, Object Management Group, 2008.

[80] Flavio Oquendo. π-ADL: An architecture description language based on the higher-order typed π-calculus for specifying dynamic and mobile architectures. *ACM SIGSOFT Software Engineering Notes*, 2004.

[81] Preeti Ranjan Panda. SystemC: A modeling platform supporting multiple design abstractions. In *Proceedings of the International Symposium on System Synthesis, Volume 14*, 2001.

[82] Cécile Péraire, R.A. Riemenschneider, and Victoria Stavridou. Integrating the Unified Modeling Language with an architecture description language. In *Proceedings of OOPSLA'99*, 1999.

[83] Dewayne E. Perry and Alexander L. Wolf. Foundations for the study of software architecture. *Software Engineering Notes*, 17(4), 1992.

[84] Ruben Prieto-Diaz and James N. Neighbors. Module interconnection languages. *Journal of Systems and Software*, 6(4), 1986.

[85] Steve Pruitt, Doug Stuart, Wonhee Sull, and T.W. Cook. The merit of XML as an architecture description language meta-language. Technical report, Microelectronics and Computer Technology Corp., 1998.

[86] B. Randell and J. N. Buxton. Software engineering techniques: Report on a conference. Technical report, Scientific Affairs Division, NATO, 1970.

[87] Binoy Ravindran. Engineering dynamic of real-time distributed systems: Architecture, systems description language, and middleware. *IEEE Transactions on Software Engineering*, 28(1), January 2002.

[88] James E. Robbins, Nenad Medvidovic, David F. Redmiles, and David E. Rosenblum. Integrating architectural description languages with a standard design method. In *Proceedings of the 20th International Conference on Software Engineering (ICSE'98)*, 1998.

[89] SAE. Avionics architecture description language (AADL). Technical report, The Society of Automotive Engineers, 2001.

[90] SAE. Architecture analysis and design language (AADL). Technical Report AS5506, The Society of Automotive Engineers, 2004.

[91] SAE. Architecture analysis and design language (AADL). Technical Report AS5506/1, The Society of Automotive Engineers, 2006.

[92] Andrew P. Sage and Christopher D. Cuppan. On the systems engineering and management of systems of systems and federation of systems. *Information Knowledge Systems Management*, 2, 2001.

[93] Bradley Schmerl. xAcme: CMU extensions to xArch. Technical report, Dept. of Computer Science, Carnegie Mellon University, 2001.

[94] Stephen C. Seow. *Designing and Engineering time. The Psychology of Time Perception in software.* Addison Wesley, 2008.

[95] Mary Shaw and Paul Clements. The golden age of software architetcure. *IEEE Software*, March – April 2006.

[96] Mary Shaw, Robert DeLine, Daniel V. Klein, Theodore L. Ross, David M. Young, and Gregory Zelesnik. Abstractions for software architecture and tools to support them. *IEEE Transactions on Software Engineering*, 21(4), April 1995.

[97] Mary Shaw and David Garlan. *Software architecture: perspectives on an emerging discipline.* Prentice Hall, 1996.

[98] Herbert A. Simon. The architecture of complexity. *Proceedings of the American Philosophical Society*, 106(6), 1962.

[99] J. M. Spivey. *The Z Notation, A Reference Manual*. International series in computer science. Prentice Hall, 2nd edition, 1992.

[100] S. Tucker Taft, Robert A. Duff, Randall L. Brukardt, Erhard Ploedereder, and Pascal Leroy, editors. *Ada 2005 Reference Manual. Language and Standard Libraries*. Springer, 2006.

[101] Richard N. Taylor, Nenad Medvidovic, Kenneth M. Anderson, E. James Whitehead Jr., and Jason E. Robbins. A component- and message-based architectural style for GUI software. In *Proceedings of the 17th International Conference on Software Engineering (ICSE'95)*, 1995.

[102] S. Vestal. A cursory overview and comparison of four architecture description languages. Technical report, Honeywell Technology Center, 1993.

[103] S. Vestal. MetaH programmers manual. Technical report, Honeywell Technology Center, 1996.

[104] David Wile. AML: an architecture meta language. In *14th IEEE International Conference on Automated Software Engineering*, 1999.

[105] David S. Wile. Architecture style-based calculi for non-functional properties. In *Proceedings of 18th IEEE Conference of Automated Software Engineering ASE'03*, 2003.

[106] Jianwen Zhu and Daniel D. Gajski. OpenJ: An extensible system level design language. In *Proceedings of the Conference on Design, Automation and Test in Europe*, 1999.