A Timed Model of *Circus* with the Reactive Design Miracle

Kun Wei, Jim Woodcock and Alan Burns

Computer Science, University of York

20 Jan 2009
Circus is a combination of CSP, Z and the refinement calculus.

The semantic model of UTP is based on the alphabetised relational calculus.

A relation is a pair \((\alpha P , P)\)

- \(\alpha P = in\alpha P \cup out\alpha P\)
- Conditional: \(P \triangleleft b \triangleright Q\)
- Composition: \(P;Q\)
- Assignment: \((x:=e)\)
- Non-determinism: \(P \sqcap Q\)
- The complete lattice of relations: \(\text{Abort(true)}\) and \(\text{Miracle(false)}\)
A theory is a collection of relations, containing three essential parts:

- an alphabet: a set of variable names, e.g., okay, tr;
- a signature: it gives the rules for syntax;
- healthiness conditions: a collection of some fundamental laws which rule out those relations we are not interested in.
A design in UTP is a relation that can be split into a precondition-postcondition pair with an observational variable.

\[ P \vdash Q \models \text{okay} \land P \Rightarrow \text{okay}' \land Q \]

To describe timed reactive processes and its behaviour with time, we introduce more observational variables:

- \text{wait} and \text{wait}': it is used to distinguish an intermediate observation from the observation made on termination.
- \text{t} and \text{t}': the start point and end point of a time interval.
- \text{tr} and \text{tr}': traces
- \text{ref} and \text{ref}': refusal sets
- \text{v}, \text{v}': values of a process's local variables
Healthiness conditions for reactive processes

- R1  \( P = P \land tr \leq tr' \): never change history
- R2  \( P(tr, tr') = P(\langle \rangle, tr' - tr) \): ignore history
- R3  \( P = \Pi_{rea} \triangleleft wait \triangleright P \): wait to start, preserving divergence
- R4  \( P = P \land t \leq t' \): never undo time

Then a reactive miracle is defined as follows:

\[ \top_R \equiv R(true \vdash false) = R(\neg okay) \]

So the new timed model is a complete lattice, rather than the complete partial orders of the standard model for CSP.
Primitive processes

- Chaos: the reactive abort
  \[ CHAOS \equiv R (\text{false} \vdash \text{true}) \]
- Stop: a deadlocked process
  \[ STOP \equiv R (\text{true} \vdash tr' = tr \land wait') \]
- Skip: terminates immediately
  \[ SKIP \equiv R (\text{true} \vdash tr' = tr \land v' = v \land \neg wait' \land t' = t) \]
Prefixing a miracle

\[ a \rightarrow SKIP \equiv R \begin{cases} \text{true} \vdash \langle \text{wait}' \rangle \land v' = v \\ tr' = tr \land a \notin ref' \land wait' \land v' = v \end{cases} \]

\[ a \rightarrow \top_R = R (\text{true} \vdash (tr' = tr \land a \notin ref' \land wait' \land v' = v)) \]

This is a miraculous process that is waiting for interaction with its environment, but never actually performing it even if the event \( a \) has been offered.
\[(a \rightarrow SKIP) R \]

This is another very strange process that states, if its predecessor successfully terminates, it performs the event \(a\) and terminates immediately. Note that, since there is not the case that \(wait'\) is true, the process has no state that it is waiting for the interaction of its environment.

- Internal choice: \(P \sqcap R = P\)
- Parallel composition: \(P \parallel R = R\)
Ordered simultaneity

- In timed CSP, a process, $a \rightarrow b \rightarrow \text{SKIP}$, may have a trace like $\langle \{t, a\}, \{t, b\} \rangle$.
- True concurrency is allowed in CSPP in which multiple events can occur simultaneously but without any order.
- The most important contribution of reactive miracles to Circus is we can define ordinary \textit{urgent} events.
- In timed CSP, internal events are \textit{urgent} because they cannot be blocked.
- In our model, an urgent event means it cannot be blocked under some circumstances.
Urgent events

\[
((a \rightarrow SKIP) \parallel \top_R) \parallel STOP = \top_R
\]

\[
c \rightarrow (a \rightarrow (b \rightarrow SKIP \parallel \top_R)) \parallel STOP
\]

\[
= c \rightarrow \top_R
\]

\[
c \rightarrow (a \rightarrow (b \rightarrow SKIP \parallel \top_R)) \parallel STOP
\]

\[
= c \rightarrow (a \rightarrow \top_R)
\]

\[
\hat{a} \rightarrow SKIP \equiv (a \rightarrow SKIP) \parallel \top_R
\]
Radio time announcement

- In a hour band,
  \[ Time(i) = \text{hour}.i; \quad \text{WAIT} \; 1; \quad Time((i + 1) \mod 24) \]

- In a second band,
  \[ Ticks(i) = 
      \text{beep} \rightarrow \text{beep} \rightarrow \text{beep} \rightarrow \text{beep} \rightarrow \text{beeeep} \rightarrow \] 
  announce\_i \rightarrow \text{SKIP} \]

- To maintain the consistency of mapping, we define
  \[ UTicks(i) = 
      \text{beep} \rightarrow \ddagger\text{beep} \rightarrow \ddagger\text{beep} \rightarrow \ddagger\text{beep} \rightarrow \ddagger\text{beeeep} \rightarrow \] 
  \ddagger\text{announce}.i \rightarrow \text{SKIP} \]
To maintain the coordination, we define \textit{hour}.i is urgent:

\[
\begin{align*}
& (Time(i) || UTicks(i)) \\
& \text{beeeep, hour}.i
\end{align*}
\]

\[
\text{beeeep} \rightarrow \frac{1}{2} \text{hour}.i \rightarrow \text{SKIP}
\]
Strick deadline operator:

\[ P \triangleright d \triangleq P \triangleright_d R \]

- \[ a \rightarrow ((b \rightarrow \text{SKIP}) \triangleright d) \]
- \[ (a \rightarrow b \rightarrow \text{SKIP}) \triangleright d \]

\[ (((a \rightarrow \text{SKIP}) \triangleright d) \parallel ((b \rightarrow \text{SKIP}) \triangleright d)) \parallel \{a,b\} \]

\[ a \rightarrow b \rightarrow \text{SKIP} \]
In this work we present a timed model for *Circus* involving the reactive miracle.

The model seems to be a promising contribution to the timebands model.

We need to further discuss the refinement of systems in detail.

We are also about to explore a hybrid system that effectively uses discrete time to express concurrency and naturally uses continuous time to describe the variables that are changing continuously.