RT-Simex: retro-analysis of execution traces

[Research Demonstration]

J. DeAntoni, F. Mallet  
I3S, UNSA, INRIA  
first.surname@unice.fr

F. Thomas, G. Reydet  
OBEO  
first.surname@obeo.fr

C. Mraidha  
CEA-List  
first.surname@cea.fr

L. Gauthier  
Atego  
first.surname@atego.com

L. Rioux, N. Sordon  
Thalès R&T  
first.surname@thalesgroup.com

ABSTRACT
This presentation demonstrates the early results from the French ANR project RT-Simex. RT-Simex proposes a set of tools to analyze timing of parallel embedded code and trace the simulation results back to the initial models from which the code was generated. The whole tool-set relies on standard formats (UML MARTE, Open Trace Format) to ensure a perennial use. This is achieved through the polychronous logical time model of MARTE.

Categories and Subject Descriptors
C.3 [Special-purpose and application-based systems]: real-time and embedded systems; C.4 [Performance of systems]: modeling techniques

1. INTRODUCTION
Models abstract away the irrelevant aspects of a system to focus on what is important for a given purpose. Model-driven engineering provides tools and techniques to deal with models as long as possible before deploying the code. The objective is to fix most bugs early in the design phase to avoid, or at least reduce, the costly debug phase afterwards. In embedded systems, the residual bugs, and specifically the timing bugs, are very difficult to identify because being embedded prevents the users from having a full access to the dynamic state of the resources (memory, registers, devices). Time debugging then requires the use of observers and instrumented code to export the relevant information and detect faults. Faults must be traced back to the initial models. The RT-Simex project (http://www.rtsimex.org) is funded by the French National Research Agency (ANR) to address this traceability issue between execution code and models in the specific domain of embedded systems. This presentation shows the early results and tools of the project that started in January 2009. The demonstration specifically focuses on the retro-analysis (confrontation of the executed code against the design model) of real-time constraints.

2. RT-SIMEX
RT-Simex gathers academic partners (CEA-List, INRIA, UBO), tool vendors (Obeo, Atego), and an industrial partner (Thalès R&T) who provides case studies. It addresses parallel applications that can execute on several platforms (Real-Time Linux, Real-Time Java, Accord). It relies on industry standards (UML MARTE [6], Open Trace Format [4]) and its foundational part is built on open-source tools. The initial models are captured as UML [7] models annotated with MARTE [6] stereotypes. MARTE is the recently adopted OMG UML profile for Modeling and Analysis of Real-Time and Embedded systems. It extends the UML by providing, amongst others, a very expressive time model [2]. Instrumented code is generated from the models to observe specific events. The code is then executed on the different platforms and produces an execution trace in the Open-Trace Format (OTF). Events in the OTF trace are analyzed and confronted to the initial constraint specification.

3. TIME CONSTRAINT SPECIFICATIONS
MARTE time model is inspired from the work on polychronous languages [3] and on the tagged systems [5]. It considers independent model elements, called clocks, that can be logical or physical. Occurrences of the clocks are called instants. A time structure is a set of clocks C and relations on instants. The basic relations are precedence (<), coincidence (≡), and exclusion (≠). For any instants i and j in a time structure, i < j means that the only acceptable execution traces are those where i occurs strictly before (precedes) j. i ≡ j imposes instants i and j to be coincident, whereas i ≠ j forbids the coincidence of the two instants. The clock constraint specification language (CCSL) complements to the MARTE time model by defining formally a set of kernel clock constraints, which apply infinitely many instant relations. The syntax and operational semantics of CCSL constraints is defined in a technical report [1]. Libraries of user-defined constraints can be built by composing existing constraints. A library of constraints relevant for RT-Simex has been built. Obeo Designer\footnote{http://www.obeo.fr} is used as a graphical front-end to easily apply the real-time CCSL constraints to the behav-
ioral models (activity, state, and interaction diagrams) depending on the analysis context. Contrary to most real-time constraint tools, we use a polychronous time model that allows the duration and time values to be expressed relatively to other clocks and not only relatively to a common chronometric clock (like second). For instance, a duration can be expressed relative to the clock cycle of a given processor core or bus. In low power models, the clock cycle varies according to some optimization criteria.

4. TIMING ANALYSIS
The formal operational semantics of CCSL constraints makes CCSL executable. From a MARTE model equipped with time constraints (by using Obeo Designer), it is possible simulate the model according to the CCSL semantics by using the tool we have built, named Timesquare.

4.1 Simulation and Animation
Timesquare checks that a CCSL specification is deadlock free. When a specification has a deterministic behavior, Timesquare provides the solution. Otherwise, Timesquare offers a set of simulation policies (Random, As soon as possible) to select one solution amongst the many possible. Simulation policies are a practical way to define a priority mechanism where priorities can be assigned dynamically to optimize one criterion or another. The selected solution can drive the animation of the model. The model can then be visualized by using any UML tool like the free and open-source UML tool Papyrus. The constraints are then used by Timesquare to animate the UML model in Papyrus and to provide a timing diagram or a sequence diagram corresponding to an execution that respect the constraints. Figure 1 shows an example of timing diagram produced in simulation by Timesquare (the video is available at http://www-sop.inria.fr/aoste/?r=9&s=30&tsq=6&open_SDF=1#SDF).

4.2 Execution and Retro-analysis
From the model, the code is generated so as to observe the selected events (model elements associated with clocks). The code is then executed directly on the target platform and an OTF trace is produced. An OTF trace is a fully ordered trace. Therefore, it can be transformed into a synchronous CCSL model. The clock calculus engine of Timesquare is used to verify that the generated trace is a refinement of the initial CCSL specification. If it does not conform to the specification, the violation is reported along with debug information (like the violated constraint, the step when it happened, the state of other constraints...). The two CCSL models involved do not have the same role. The one extracted from the trace is the actual behavior, the other one is not used to enforce a behavior (as usual CCSL specifications do) but only to verify that the two models are consistent with each other. CCSL constraints are, in that situation, used as assertions.

4.3 Reconciliation
When several programs execute concurrently on different platforms, each of them produce a trace. If the platform are completely unsynchronized, it is not possible to reconcile the traces together. However, the different parts usually interact with each other. The communications between the two parts are identified on the initial model and can be used as a loose synchronization information. For instance, a message is always sent before being received. This synchronization information can be expressed as a CCSL specification. Combining this specification with the CCSL models extracted from the traces produces a partial order that is analyzed once again with Timesquare.

5. CONCLUSIONS
RT-Simex addresses the key issue to trace simulation faults back to a UML MARTE model. The MARTE time model and CCSL are used to specify real-time constraints and to drive the instrumentation of the simulation code. Traces produced by the simulation are confronted to the specification and are used to enrich the model with operational information in an iterative process. Using logical time is key to select relevant events to be observed and to reconcile several simulation traces into a partial time model. To ensure a perennial use of the proposed tools, RT-Simex relies on standard formats including UML MARTE and the Open Trace Format.

6. REFERENCES