The SynchAADL2Maude Tool

Kyungmin Bae[1], Peter Csaba Ölveczky[2],
José Meseguer[1], and Abdullah Al-Nayeem[1]

[1] University of Illinois at Urbana-Champaign
[2] University of Oslo

Abstract. SynchAADL2Maude is an Eclipse plug-in that uses Real-Time Maude to simulate and model check Synchronous AADL models. Synchronous AADL is a variant of the industrial modeling standard AADL that supports the modeling of synchronous embedded systems. In particular, Synchronous AADL can be used to define in AADL the synchronous models in the PALS methodology, in which the very hard tasks of modeling and verifying an asynchronous distributed real-time system that should be virtually synchronous can be reduced to the much simpler tasks of modeling and verifying the underlying synchronous design.

1 Introduction

The *Architecture Analysis & Design Language* (AADL) [6] is an industrial modeling standard used in avionics, aerospace, automotive, medical devices, and robotics communities—including Honeywell, Rockwell-Collins, Lockheed Martin, General Dynamics, Airbus, the European Space Agency, Dassault, EADS, Ford, and Toyota—to describe an embedded real-time system as an assembly of software components mapped onto an execution platform.

A number of tools support the formal analysis of different aspects of models in various fragments of AADL. However, since the components in AADL models interact asynchronously, their model checking becomes unfeasible even for fairly small models due to the state space explosion caused by the interleavings.

We therefore define in [1] a variant of AADL, called *Synchronous AADL*, for modeling *synchronous* real-time systems in AADL. This effort was motivated by the observation that many automotive and avionics systems should be *virtually synchronous*—that is, conceptually, there is a logical period during which all components perform a transition and send messages to each other—that must be realized in a distributed environment with network delays, skewed local clocks, etc. Together with colleagues at UIUC and Rockwell-Collins, we have proposed the PALS transformation [3,4], whose key idea is that one can model and verify the much simpler synchronous design, and PALS then provides a correct-by-construction distributed asynchronous model. There are also other transformations relating synchronous and asynchronous systems for distributed real-time architectures, such as the time-triggered architecture (TTA) [2]. Synchronous AADL makes it possible to define such synchronous models in AADL.

The *SynchAADL2Maude* OSATE[1] plug-in is a recent simulation and linear temporal logic (LTL) model checking tool for Synchronous AADL. The tool automatically synthesizes a Real-Time Maude [5] model from a Synchronous AADL model, provides support to conveniently define LTL properties of the Synchronous AADL model, and performs the Real-Time Maude model checking *within* OSATE. This enables a model-engineering process for important classes of distributed real-time systems that combines the convenience of AADL modeling, the complexity reduction of PALS and TTA, and formal verification in Real-Time Maude. We illustrate the use of SynchAADL2Maude in Section 3 with a virtually synchronous avionics system, whose distributed asynchronous version (even in very simple settings) has millions of reachable states and cannot be feasibly model checked, but where the Synchronous AADL model of the corresponding synchronous PALS design can be verified by our tool in less than a second.

The tool, together with related papers and technical reports, is available at `http://www.cs.illinois.edu/~kbae4/SynchAADL/`.

2 Background: Real-Time Maude and Synchronous AADL

Real-Time Maude [5] is a rewriting-logic-based formal specification language and analysis tool for real-time systems. Real-Time Maude provides simulation capabilities, as well as (unbounded and time-bounded) explicit-state reachability analysis and LTL and timed CTL model checking.

The *Synchronous AADL* modeling language [1] supports the modeling of synchronous designs in AADL, including both synchronous PALS designs and other synchronous designs that can be mapped onto different distributed real-time architectures. Synchronous AADL is an annotated sublanguage of AADL, identifying a set of AADL models that can be considered as synchronous, and adding a *property set* `SynchAADL` to declare Synchronous AADL-specific properties. Since Synchronous AADL is intended to model synchronous *designs*, it disregards the hardware and scheduling features of AADL and focuses on the behavioral and structural subset of AADL, namely, hierarchical system, process, and thread components, ports and connections, and thread behaviors defined in the *behavior annex* standard. The formal Real-Time Maude semantics of Synchronous AADL is defined in [1].

3 Using the SynchAADL2Maude Tool

We exemplify the use of the SynchAADL2Maude tool with an avionics system developed by Steve Miller and Darren Cofer at Rockwell-Collins [4]. In *integrated modular avionics*, there are multiple physically separated *cabinets* on the aircraft so that physical damage does not take out the computer system. The *active standby* system considers the case of two cabinets and focuses on the logic for deciding which side is *active*. The architecture of the system is shown in Figure 1.

[1] The OSATE modeling environment provides a set of Eclipse plug-ins for AADL.

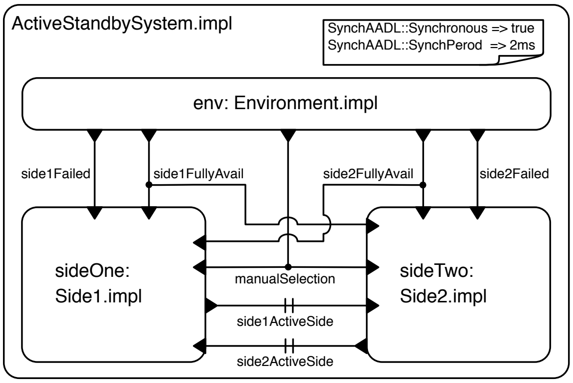


Fig. 1. The architecture of the active standby system

In SynchAADL2Maude, the properties to be verified are managed by an XML file. One important property that the system should satisfy is that *if a side is failed, the other side should become active.* Side *i* has failed if it has received the value `true` in its `side`*i*`Failed` port. Using the predefined proposition `value of` *port* `in component` *thread* `is` *v*, the formula `side`*i*`Failed` can be defined as follows:

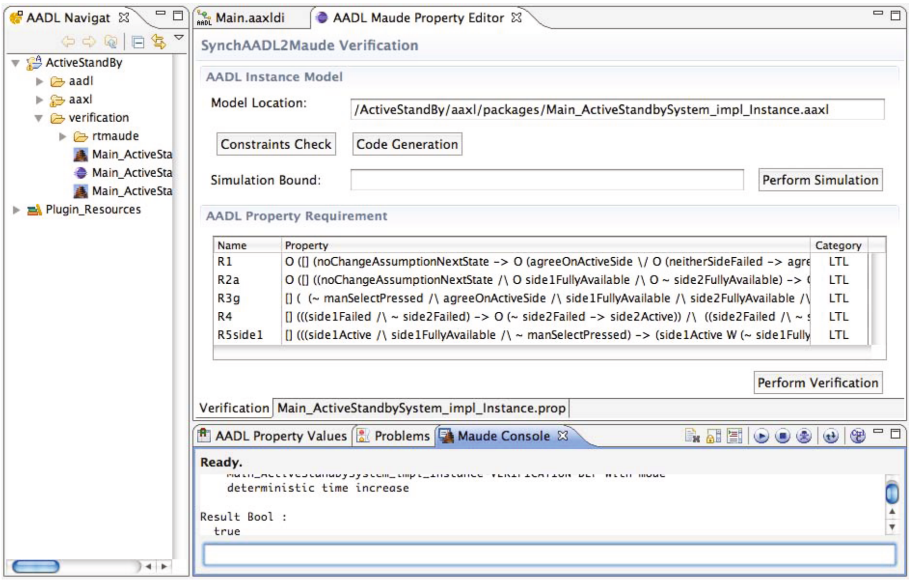
```
<definition> <name> side1Failed </name>
 <value> value of side1Failed in component MAIN -> sideOne -> sideProcess -> sideThread is true
 </value>
</definition>
```

The formulas `side`*i*`Active` are defined in the same way. The LTL property to be verified is then declared by the `command` tag as follows (where '`~`', '`->`', '`[]`', and '`O`' denote, resp., negation, implication, and the "always" and "next" operators):

```
<command>  <name>R4</name>
 <value type = "ltl"> [] (((side1Failed /\ ~ side2Failed) -> O (~ side2Failed -> side2Active)) /\
                          ((side2Failed /\ ~ side1Failed) -> O (~ side1Failed -> side1Active)))
 </value>
</command>
```

Figure 2 shows the SynchAADL2Maude window for the active standby system. The `Constraints Check`, `Code Generation`, and `Perform Verification` buttons are used to, respectively, check whether a model is a valid Synchronous AADL model, generate the corresponding Real-Time Maude model, and model check the LTL properties given by the XML property file and shown in the "AADL Property Requirement" table. The results of the model checking are shown in the "Maude Console." Counterexamples from the LTL model checking are presented in a reasonably intuitive and concise way.

We have verified each requirement of the Synchronous AADL model of the active standby system, which has 203 reachable states, in 0.6 seconds on an Intel Xeon 2.93 GHz with 24GB RAM. As shown in [3], where we define directly in



SynchAADL2Maude Verification

AADL Instance Model

Model Location: /ActiveStandBy/aaxl/packages/Main_ActiveStandbySystem_impl_Instance.aaxl

Simulation Bound:

AADL Property Requirement

Name	Property	Category
R1	O ([] (noChangeAssumptionNextState -> O (agreeOnActiveSide \/ O (neitherSideFailed -> agre	LTL
R2a	O ([] ((noChangeAssumptionNextState /\ O side1FullyAvailable /\ O ~ side2FullyAvailable) -> (	LTL
R3g	[] ((~ manSelectPressed /\ agreeOnActiveSide /\ side1FullyAvailable /\ side2FullyAvailable /\	LTL
R4	[] (((side1Failed /\ ~ side2Failed) -> O (~ side2Failed -> side2Active)) /\ ((side2Failed /\ ~ s	LTL
R5side1	[] (((side1Active /\ side1FullyAvailable /\ ~ manSelectPressed) -> (side1Active W (~ side1Fully	LTL

Ready.

```
    deterministic time increase

Result Bool :
  true
```

Fig. 2. SynchAADL2Maude window in OSATE

Real-Time Maude models of both the synchronous and the asynchronous design of the active standby system, it is unfeasible to model check the corresponding *asynchronous* design: the *simplest possible* asynchronous model—with no message delays, no execution times, and perfect local clocks—has 3,047,832 reachable states and its model checking takes 1,249 seconds. If the message delay can be either 0 or 1 then no model checking terminates in reasonable time.

Acknowledgments. This work has been supported by Boeing Corporation under grant C8088, by The Research Council of Norway, and by the "Programa de Apoyo a la Investigación y Desarrollo" (PAID-02-11) of the Universitat Politécnica de València.

References

1. Bae, K., Ölveczky, P.C., Al-Nayeem, A., Meseguer, J.: Synchronous AADL and Its Formal Analysis in Real-Time Maude. In: Qin, S., Qiu, Z. (eds.) ICFEM 2011. LNCS, vol. 6991, pp. 651–667. Springer, Heidelberg (2011)
2. Kopetz, H., Bauer, G.: The time-triggered architecture. Proc. IEEE 91(1) (2003)
3. Meseguer, J., Ölveczky, P.C.: Formalization and Correctness of the PALS Architectural Pattern for Distributed Real-Time Systems. In: Dong, J.S., Zhu, H. (eds.) ICFEM 2010. LNCS, vol. 6447, pp. 303–320. Springer, Heidelberg (2010)
4. Miller, S.P., Cofer, D.D., Sha, L., Meseguer, J., Al-Nayeem, A.: Implementing logical synchrony in integrated modular avionics. In: Proc. DASC 2009. IEEE (2009)
5. Ölveczky, P.C., Meseguer, J.: Semantics and pragmatics of Real-Time Maude. Higher-Order and Symbolic Computation 20(1-2), 161–196 (2007)
6. SAE AADL Team: AADL homepage (2009), `http://www.aadl.info/`