

CHARMY: A Framework for Designing and Validating Architectural Specifications

by Mauro Caporuscio, Paola Inverardi, Henry Muccini and Patrizio Pelliccione

CHARMY is a tool-supported framework, initially proposed in 2001 to *check the consistency between architectural models*. Since then the approach has evolved and can now be employed in a number of ways: to check the validity of architectural properties with respect to a software architecture specification; to incrementally create an architectural prototype; to use a compositional approach for the verification of middleware-based applications; to verify architectural patterns.

Software Architectures (SAs) emerged in the nineties and were used to structure complex software systems, exploiting commonalities in the organization of specific domains and providing a high-level system description. Nowadays SA is an independent discipline focusing on the overall organization of a large software system and using *abstractions* to express the logical coordination structure of complex distributed systems. The emphasis in SA specification is on capturing the *system structure* (i.e., the architecture topology) by identifying architectural components and connectors and the *system behavior* (i.e., the architecture dynamics) by identifying how components and connectors interact.

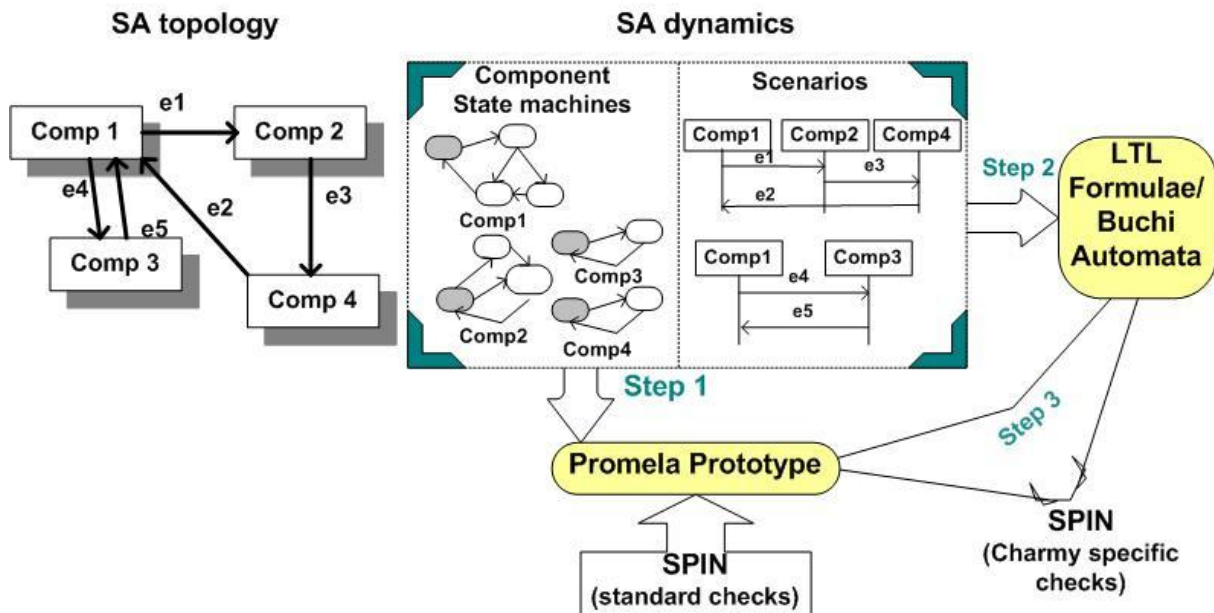


Fig. 1: The CHARMY Framework

To specify SAs, informal box-and-line notations have now been replaced/complemented by formal and rigorous Architecture Description Languages (ADLs). Many methods and tools, developed on the basis of these ADLs, have been proposed for SA-level testing, analysis and model checking, deadlock analysis, performance analysis and so on. In other terms, many studies have proved the suitability of these formal languages for analysis purposes.

However, we note that formal languages are not commonly used in industrial applications which tend to prefer model-based, semi-formal notations. In fact, in current industrial practice, *state-based machines* and *scenarios* are the most commonly used tools to model behavioral aspects, even at the architectural level: state diagrams describe component behavior while scenarios (e.g., Message

Sequence Charts or Sequence diagrams) identify their interaction. The introduction of the Unified Modelling Language (UML) as the de-facto standard to model software systems has increased the use of modeling notations to describe software systems. Furthermore, the introduction of UML extensions to model SAs makes UML diagrams more suitable for SA modeling.

The CHARMY framework, introduced in the early stages of the software development process, aims at assisting the software architect in the design and validation phases. CHARMY enables the formulation of the SA through model-based specifications (extensively used in industrial projects), which are automatically translated into a formal prototype and validated against selected properties.

In particular, the state diagrams used to specify how the architectural components should behave are automatically interpreted in order to synthesize a formal prototype (see Fig. 1). Properties to be validated are modeled using scenarios. Through the CHARMY engine, the model checker SPIN is used to check the conformance of the formal prototype with respect to the behavioral properties.

In order to make CHARMY useful in an industrial context, we have hidden the complexity of the approach, providing the software engineer with an automated easy-to-use tool that takes the architectural model as input, creates a prototype and automatically analyzes it, reducing human intervention to the minimum. The tool is a plug-in system and structured to allow an easy and quick evolution and integration with other tools.

By using CHARMY, the software architect can save time and improve space efficiency. CHARMY provides guidelines on how to model the system and automatically generates an optimized system prototype.

In the future, we are planning several interesting extensions. Currently, the user runs CHARMY to produce the formal architectural prototype and eventually uses SPIN for its verification. With the aim of enabling the verification of the architecture without the need for a good familiarity with model checking processes, we are also developing a plug-in component which will integrate SPIN in CHARMY, thus hiding the complexity of the model checker. Another interesting idea for future work is to use CHARMY to manage the counter example output produced by SPIN. SPIN is already able to represent counter examples graphically, but the visualized counter example often has an abstraction level which differs from the system specification. However, we are not restricted to the use of SPIN as our model checker. We are also investigating the possibility of using the SMV or Bogor model checkers. For example, it could be interesting to exploit the plug-in structure of Bogor to define a customized algorithm search.

Link:

<http://www.di.univaq.it/charmym>

E-mail: charmym@di.univaq.it

Please contact:

Project Advisor: Paola Inverardi – University of L’Aquila

E-mail: inverardi@di.univaq.it

<http://www.di.univaq.it>