Science of Computer Programming 89 (2014) 41–68

Contents lists available at ScienceDirect

Science of Computer Programming

www.elsevier.com/locate/scico

Modelling and simulation of asynchronous real-time systems using Timed Rebeca

Arni Hermann Reynisson, Marjan Sirjani *, Luca Aceto, Matteo Cimini, Ali Jafari, Anna Ingolfsdottir, Steinar Hugi Sigurdarson

School of Computer Science, Reykjavik University, Reykjavik, Iceland

HIGHLIGHTS

- Timed Rebeca as an actor-based modelling language extended with time constraints.
- The formal semantics of Timed Rebeca using Structural Operational Semantics (SOS).
- A tool for mapping Timed Rebeca models to Erlang.
- Examples of applications of Timed Rebeca to different small and medium sized case studies.
- Experimental results from the simulation of the resulting Timed Rebeca models using McErlang.

ARTICLE INFO

Article history:
Received 31 January 2012
Received in revised form 20 January 2014
Accepted 21 January 2014
Available online 24 January 2014

Keywords:
Concurrent system
Timing constraints
Distributed systems
Actor modelling
Analysis

ABSTRACT

In this paper we propose Timed Rebeca as an extension of the Rebeca language that can be used to model distributed and asynchronous systems with timing constraints. Timed Rebeca restricts the modeller to a pure asynchronous actor-based paradigm, where the structure of the model represents the service oriented architecture, while the computational model matches the network infrastructure. The modeller can specify both computational and network delay, and assign deadlines for serving a request. We provide the formal semantics of the language using Structural Operational Semantics, and show its expressiveness by means of examples. We developed a tool for automated translation from Timed Rebeca to the Erlang language, which provides a first implementation of Timed Rebeca. We can use the tool to set the parameters of Timed Rebeca models, which represent the environment and component variables, and use McErlang to run multiple simulations for different settings. The results of the simulations can then be employed to select the most appropriate values for the parameters in the model. Simulation is shown to be an effective analysis support, specially where model checking faces almost immediate state explosion in an asynchronous setting.

1. Introduction

This paper presents an extension of the actor-based Rebeca language [1,2] that can be used to model distributed and asynchronous systems with timing constraints. This extension of Rebeca is motivated by the ubiquitous presence of real-time computing systems, whose behaviour depends crucially on timing as well as functional requirements.

A well-established paradigm for modelling the functional behaviour of distributed and asynchronous systems is the actor model. This model was originally introduced by Hewitt [3] as an agent-based language, and is a mathematical model of concurrent computation that treats *actors* as the universal primitives of concurrent computation [4]. In response to a message that it receives, an actor can make local decisions, create more actors, send more messages, and determine how

* Corresponding author.

0167-6423/$ – see front matter
http://dx.doi.org/10.1016/j.scico.2014.01.008

to respond to the next message it receives. Actors have encapsulated states and behaviour, and are capable of redirecting communication links through the exchange of actor identities. Different interpretations, dialects and extensions of actor models have been proposed in several domains and are claimed to be the most suitable modells of computation for some of the dominating applications, such as multi-core programming and web services [5].

Reactive Objects Language, *Rebeca* [1], is an operational interpretation of the actor model with formal semantics and model-checking tools. Rebeca is designed to bridge the gap between formal methods and software engineers. The formal semantics of Rebeca is a solid basis for its formal verification. Compositional and modular verification, abstraction, symmetry and partial-order reduction have been investigated for verifying Rebeca models. The theory underlying these verification methods is already established and is embodied in verification tools [6,7,1]. With its simple, message-driven and object-based computational model, Java-like syntax, and accompanying set of verification tools, Rebeca is an interesting and easy-to-learn model for practitioners.

Motivation and contribution Although actors are attracting more and more attention both in academia and industry, little work has been done on timed actors and even less on analyzing timed actor-based models. In this work we present

- Timed Rebeca by extending Rebeca with time constraints,
- the formal semantics of Timed Rebeca using Structural Operational Semantics (SOS) [8],
- a tool for mapping Timed Rebeca models to Erlang,
- examples of applications of Timed Rebeca to different small and medium sized case studies, and
- experimental results from the simulation of the resulting Timed Rebeca models using McErlang [9].

The main contribution of this work is offering a pure asynchronous actor-based modelling language with timing primitives and analysis support. Timed Rebeca can be used in a model-driven methodology in which the designer builds an abstract model where each component is a reactive object communicating through non-blocking asynchronous messages. The structure of the model can very well represent service oriented architectures, while the computational model matches the network infrastructure. Hence the model captures faithfully the behaviour of the system in a distributed and asynchronous world.

This paper is an extended version of [10]. The main extensions are presenting a formal mapping as a syntax-directed translation from Timed Rebeca to Erlang, and a medium-sized case study where we model and simulate the BitTorrent [11] protocol using Timed Rebeca and McErlang [9].

Comparison with other timed models Comparing with the well-established timed models, like timed automata [12], TCCS [13], and real-time Maude [14], Timed Rebeca offers an actor-based syntax and a built-in actor-based computational model, which restricts the style of modelling to an event-based concurrent object-based paradigm. Modelling time-related features in computational models has been studied for a long time [15,12]; while we have no claims of improving the expressiveness of timed models, we believe that our model is highly usable due to its actor-based nature and Java-like syntax. The usability is due to the one to one correspondence between the entities of the real world and the objects in the model, and the events and actions of the real world and the computational model. Moreover, the syntax of the language is familiar for software engineers and practitioners.

Comparison with other timed actor models We know of a few other timed actor-based modelling languages [16–18] that we will discuss in more detail in Section 6, where we discuss further related work. In [16] a central synchronizer acts like a coordinator and enforces the real-time and synchronization constraints (called interaction constraints). A language for the coordinated actors is briefly proposed in [17]; however, the main focus is having reusable real-time actors without hardwired interaction constraints. The constraints declared within the central synchronizer in this line of work can be seen as the required global properties of a Timed Rebeca model. We capture the architecture and configuration of a system via a Timed Rebeca model and then we can check whether the global constraints are satisfied. The language primitives that we use to extend Rebeca are consistent with the proposal in [17]. The primitives proposed in [18] are different from ours; that paper introduced an *await* primitive whereas we keep the asynchronous nature of the model.

Analysis support In order to analyze Timed Rebeca models, we developed a tool to facilitate their simulation. In a parallel project [19], a mapping from Timed Rebeca to timed automata is developed and UPPAAL [20] is used for model checking. The asynchronous nature of Rebeca models causes state explosion while model checking even for small models. One solution is using a modular approach like in [21]. Here, we selected an alternative solution as a complementary tool for analysis. Using our tool we can translate a Timed Rebeca model to Erlang [22], set the parameters which represent the environment and component variables, and run McErlang [9] to simulate the model. The tool allows us to change the settings of different timing parameters and rerun the simulation in order to investigate different scenarios, find potential bugs and problems, and optimize the model by manipulating the settings. The parameters can be timing constraints on the local computations (e.g., deadlines for accomplishing a requested service), computation time for providing a service, and frequency of a periodic event. Parameters can also represent network configurations and delays. In our experiments we could find timing problems that caused missing a deadline, or an unstable state in the system.