Science of Computer Programming ••• (••••) •••–•••

Contents lists available at ScienceDirect

Science of Computer Programming

www.elsevier.com/locate/scico

Asynchronous synthesis techniques for coordinating autonomic managers in the cloud

Rim Abid, Gwen Salaün *, Noel De Palma

University of Grenoble Alpes, LIG, CNRS, France

ARTICLE INFO

Article history:
Received 1 April 2016
Received in revised form 24 May 2017
Accepted 24 May 2017
Available online xxxx

Keywords:
Asynchronous coordination
Autonomic managers
Distributed cloud applications
Synthesis techniques

ABSTRACT

Cloud computing allows the delivery of on-demand computing resources over the Internet on a pay-for-use basis. From a technical point of view, cloud applications usually consist of several software components deployed on remote virtual machines. Managing such applications is a challenging problem because manual administration is no longer realistic for these complex distributed systems. Thus, autonomic computing is a promising solution for monitoring and updating these applications automatically. This is achieved through the automation of administration functions and the use of control loops called autonomic managers. An autonomic manager observes the environment, detects changes, and reconfigures dynamically the application. Multiple autonomic managers can be deployed in the same system and must make consistent decisions. Using them without coordination may lead to inconsistencies and error-prone situations. In this article, we first present a simple language for expressing coordination constraints given a set of autonomic managers. Second, given a coordination expression written with that language, we propose new synthesis techniques for automatically generating an asynchronous controller. These synthesis techniques work in two steps by successively generating a model of the controller and a Java object corresponding to this model. This Java code is finally used for deploying the generated controller. As far as evaluation is concerned, we validated our approach by using it for coordinating real-world cloud applications.

© 2017 Elsevier B.V. All rights reserved.

1. Introduction

Managing complex distributed applications is a challenging problem because manual administration is no longer realistic for complex systems. Autonomic computing is a promising solution for automating the administration functions, which focus particularly on replicating virtual machines, destroying or adding them, and handling virtual machine failures in the cloud. These operations are executed by different autonomic managers considered as control loops. Each manager observes the application execution, ensures a continuous monitoring, and immediately reacts to changes by automatically executing reconfiguration tasks. Several managers can be deployed to supervise the same application and must make consistent decisions. Nonetheless, using them without coordination may lead the system into inconsistencies and error-prone situations (*e.g.*, removing a server that is necessary). As a consequence, the use of multiple managers (*e.g.*, self-repair and self-sizing managers) implemented in the same system requires taking globally consistent decisions. Hence, a manager should be aware of decisions of all managers before reacting.

* Corresponding author.
E-mail address: gwen.salaun@inria.fr (G. Salaün).

http://dx.doi.org/10.1016/j.scico.2017.05.005
0167-6423/© 2017 Elsevier B.V. All rights reserved.

Please cite this article in press as: R. Abid et al., Asynchronous synthesis techniques for coordinating autonomic managers in the cloud, Sci. Comput. Program. (2017), http://dx.doi.org/10.1016/j.scico.2017.05.005

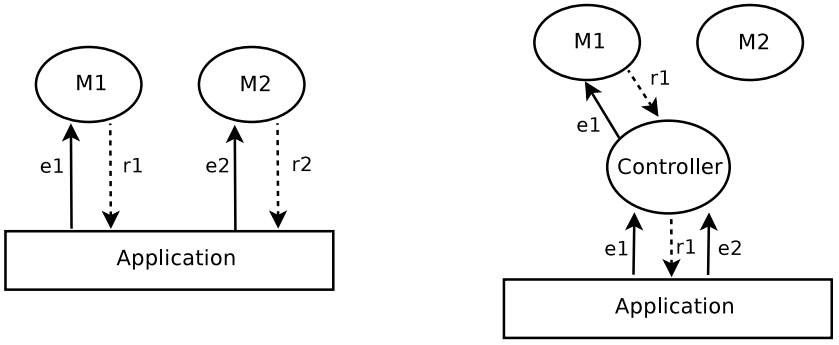

Fig. 1. Administration tasks without or with coordination.

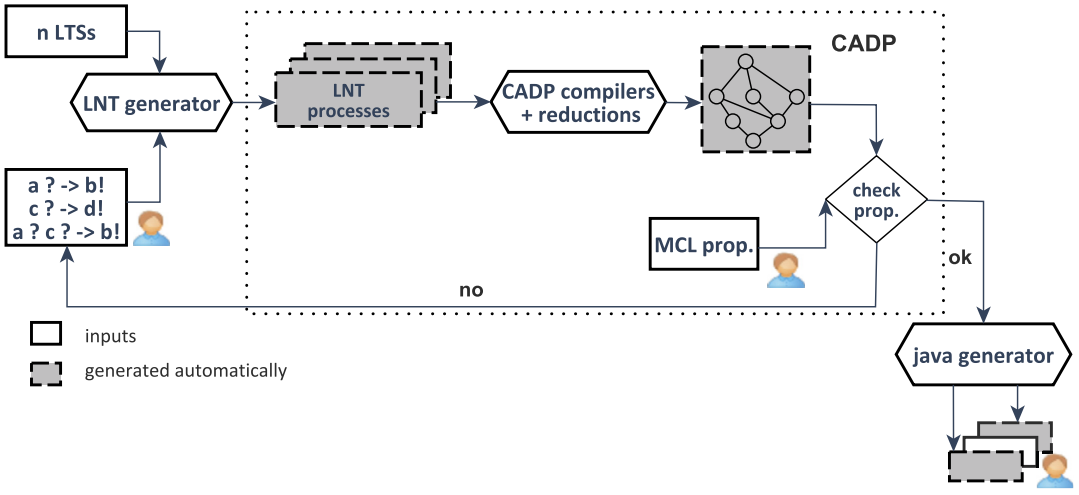

Fig. 2. Overview of our approach.

We present in this article our synthesis techniques for generating a controller, which aims at coordinating several managers. The generated controller prevents every manager from violating global objectives of all the managers. Fig. 1 shows an example with two managers (M1 and M2) administrating an application. The right hand part of this figure particularly illustrates the interest of the controller for taking globally consistent decisions (by filtering some event for instance as shown in this example).

Our controller synthesis techniques assume that all participants (managers and generated controller) interact using asynchronous communication semantics. This means that all the messages transmitted from/to the managers (controller, resp.) are stored/consumed into/from FIFO buffers. It is worth emphasizing that our approach is twice *asynchronous* in the sense that it applies on asynchronous systems (no global clock) and it relies on asynchronous communication semantics (communication via buffers).

Let us now present our solution with more details, as depicted in Fig. 2, which gives an overview of our approach. We consider as input a set of autonomic managers. Each manager is described using a formal model, namely a Labelled Transition System (LTS). As a first contribution, we define a set of reaction rules and regular expressions to specify the coordination requirements and interaction constraints. This simple language aims at expressing in an abstract way the relationship between the managers and the behaviour we expect from the controller to be generated. Given a set of manager LTSs and the coordination requirements, we propose synthesis techniques for generating an abstract model (LTS) for our controller. To do so, we rely on an encoding of our inputs (LTS models and coordination requirements) into the LNT specification language [7]. LNT is expressive enough for representing all the inputs and the way they interact together. Moreover, LNT is equipped with a rich toolbox, called CADP [16], that is used for automatically obtaining an LTS model from the LNT specification. The generated LTS corresponds to all possible executions of the controller. It is worth noting that since we rely on formal techniques and tools, all the verification techniques available in the CADP toolbox can be used for validating the generated controller. Once we have synthesized the controller LTS, a Java program is obtained using a code generator we developed. This Java program is necessary to finally deploy and use the synthesized controller for coordinating real applications. In this article, we present a typical example of an N-tier Web application for illustration purposes. We have validated our approach on several variants of this distributed application involving several instances of autonomic managers, such as

Please cite this article in press as: R. Abid et al., Asynchronous synthesis techniques for coordinating autonomic managers in the cloud, Sci. Comput. Program. (2017), http://dx.doi.org/10.1016/j.scico.2017.05.005

Download English Version:

https://daneshyari.com/en/article/4951793

Download Persian Version:

https://daneshyari.com/article/4951793

Daneshyari.com