Chapter 15
Fault-Recovery and Coherence in Internet of Things Choreographies

Sylvain Cherrier
University of Marne-la-Vallée, France

Yacine M. Ghamri-Doudane
University of La Rochelle, France

ABSTRACT

Facilitating the creation of applications for the Internet of Things (IoT) is a major concern to increase its development. D-LITe, our previous work, introduces genericity by providing a universal programming interface. Objects are dynamically configured to have their own behaviour, and their actions/reactions are considered as part of a whole application. D-LITeful Objects describe their capabilities, receives the application logic to be executed, and interacts with other. The application is seen as a choreography of behaviours. But the main issue of choreographies is the loss of coherence. Because of their unreliability, some networks used in IoT may introduce de-synchronization, leading to interactions errors. We propose to reintroduce coherence in order to keep the advantages of choreography while dealing with this main issue. An overlay of logical check-points defines the dependence between the coherent states of objects and triggers re-synchronizations. This paper ends with a discussion on the trade-off between the checking cost and the reliability improvement.

INTRODUCTION

The Internet of Things (IoT) is a rising domain that gives Internet widespread connectivity to real world objects. IoT focuses research interest because it re-uses well-known proto-cols. The success of IoT will come from the ability to easily create applications. In our previous work, we have presented D-LITe (Cherrier et al., 2011), a framework for IoT applications creation and deployment, based on services choreographies. The main advantage of services choreographies is based on the distribution of the different parts of the application across the network. The absence of central point leads to a better

DOI: 10.4018/978-1-5225-9866-4.ch015

dissemination of the uses of the resources, and a more efficient use of energy, especially in constrained networks (Cherrier et al., 2012).

However, services choreographies are subject to a major issue. Unlike orchestrations which are under the control of a unique central point, the spread of the logic may lead to desynchronisations. For example, some stakeholders may miss steps in their choreography and the whole application becomes incoherent. As it integrates WSAN (Wireless Sensors and Actuators Network), IoT uses wireless links to make Objects communicate. These wireless links are characterized by their unreliability. Indeed, tests on our testbed show that we are still facing desynchronisations, in spite of controls made at lower layers.

In this paper, we propose a new mechanism allowing IoT programmers to introduce coherence controls in order to correct the logical state of Objects involved in a global application. This coherence checking is deployed along with the services choreography. To respect the decentralized approach chosen for the framework, the coherence mechanism is organized in cascade. Some Objects start a check by sending an order to a list of "followers" Objects. Receiving this order, these "followers" check their own state, make correction if needed, then transmit in their turn the coherence control orders to their own list of "followers" Objects. By doing so, we show in this paper that we are able to reintroduce coherence. Finally, the cost of introducing such coherence is analyzed in terms of communication overhead.

This paper is organized as follow: Section II presents the background composed of related works and an introduction to our platform D-LITe. The coherence checking architecture is described in Section III. Section IV contains our experimental study and our results. Finally, concluding remarks and future research directions are given in Section V.

BACKGROUND

Related Work

Our approach of Internet of Things applications (Cherrier et al., 2011) is part of the Services Oriented Architecture(SOA) realm. SOA offers a decentralized composition of programs in a "loosely coupled platform-independent model" (Zhou et al., 2010). Canfora and Di Penta present the specificity of SOA as "radical changing (in) the development perspective" (Canfora et al., 2009). Implication of that change leads to "lack of observability of the service code" and "lack of control" because of the infrastructure independence and the absence of access to the running code. The "cost of testing" for such a decentralized software composition over heterogeneous hardware may lead to denial-of-service if too many checks are performed.

Testing Web services collaboration can be done at design time. Huang et al. (Huang et al., 2005) use language translations from OWL-S (Ontology Web Language for Services) into a model checkers compatible one, in order to test some services composition. This translation is used to generate test cases in an a priori check. To specifically test web services choreographies, Zhou et al. (Zhou et al., 2010) propose the use of assertions that "express the intention of the program by designers". A simulator processes each web services and builds the complete interaction.

All paths are checked and assertions are verified. This tool can detect design errors. However, in our case, we are more concerned by global fault checking of running choreographies, which is not necessary and as such not addressed for web applications.