Journal of Ambient Intelligence and Smart Environments 6 (2014) 107–109
DOI 10.3233/AIS-140245
IOS Press

Thesis

Efficient runtime adaptability to support context-awareness in a robotic framework

Miguel Garcia[*] and Francisco Ortin
Department of Computer Science, University of Oviedo, Spain

Keywords: Context-awareness, runtime adaptation, computational reflection, runtime performance

On June 25, 2013, Miguel Garcia defended his PhD thesis on efficient runtime adaptability for context-aware systems at the University of Oviedo. He presented his dissertation in a publicly open presentation held in the School of Computer Engineering (Fig. 1). After commenting on every question raised by the assessing committee, he was awarded the highest grade possible (*cum laude*).

Miguel Garcia obtained his PhD with International Doctorate honorable mention. His work was previously accessed by Dr. Jörg Thomaschewski (University of Applied Sciences, Emden, Germany) and Dr. Giner Alor (Technological Institute of Orizaba, Mexico). The thesis was supervised by Dr. Francisco Ortin, and the PhD dissertation Committee was composed of Professor Miguel A. Labrador (University of South Florida), Dr. J. Baltasar Garcia Perez-Schofield (University of Vigo), and Professor Juan Manuel Cueva (University of Oviedo).

Part the research work was carried out during a four-month research state at Liverpool John Moores University (UK), supervised by Professor Majdid Merabti and Dr. David Llewellyn-Jones. His work at Liverpool consisted in applying runtime adaptability to other scenarios such as distributed systems security [1] and dynamic aspect oriented development [2].

The research work was funded by the Spanish Ministry of Science and Innovation (project TIN2011-25978) and Microsoft Research. His four-year research fellowship grant was funded by the Ministry of Education and Science (including the research stay in John Moores University).

StaDyn uses a concrete type system (for var re
Type inference (reconstruction) has been imple
Developing a modification of the Hindley-Milner
INFORMÁTICA

Fig. 1. PhD dissertation defense.

Thesis summary

The thesis discusses the use of computational reflection to support context-awareness, applying it to a robotic framework [5] and proposing different runtime performance optimizations [6]. Service robots have to cope with many different situations emerged at runtime, while executing complex tasks. They should be programmed as context-aware systems, capable of adapting themselves to the execu-

[*] Corresponding author. E-mail: garciarmiguel@uniovi.es.

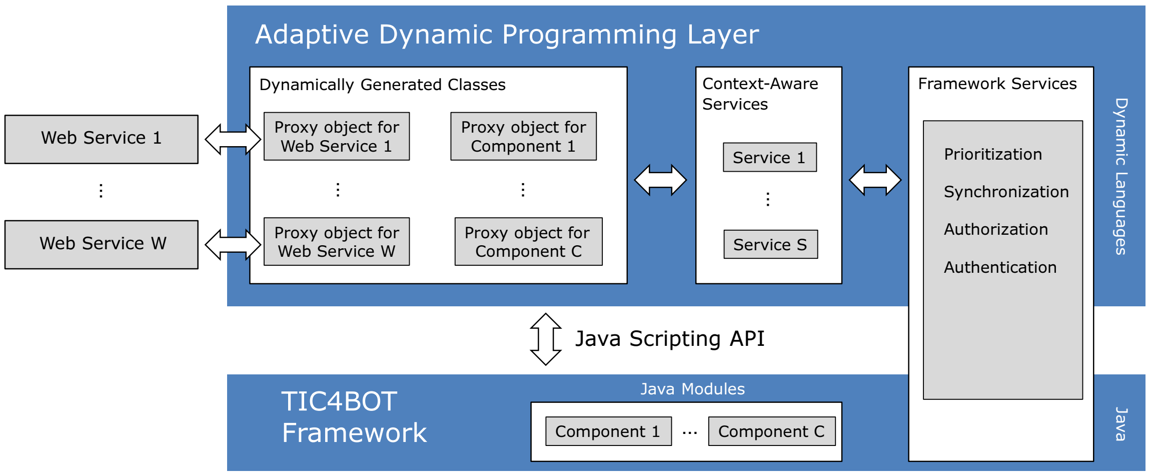

Fig. 2. Providing runtime-adaptable context-aware services over the TIC4BOT robotic framework.

tion environment. Since computational reflection is a programming language feature that offers a high level of runtime adaptability, we analyzed the suitability of reflection to fulfill the dynamism requirements of context-aware robotic systems.

Computational reflection is a language capability that allows reasoning about and acting upon the program structure and behavior, adjusting itself to changing conditions. This feature involves a great runtime adaptability to handle new situations without stopping the running applications. Although statically typed programming languages (such as Java and C#) support some level of reflection, those that provide the highest level are commonly dynamically typed.

Robotic systems have to manage the high dynamism complexity of real-world environment contexts. They must be context-aware, adapting themselves according to their location, the collection of nearby people and objects, and the changes to those objects over time. Because of the strong connection between highly runtime-adaptable robotic systems and computational reflection, we investigated how reflection could be used to facilitate the programming of runtime-adaptable context-aware robotic services. On the top of the TIC4BOT Java robotic framework, we developed an adaptive programming layer that supports adaptive context-awareness capabilities (Fig. 2) [3]. The framework provides the execution of Java components grouped into modules, together with basic prioritization, synchronization, authorization and authentication services [5]. The standard Java Scripting API (JSR 223) was used to allow programming the framework in any dynamic language. All the functionality provided by the framework in Java can be programmatically accessed at the adaptive layer, using the reflective services of any dynamic language.

Introspection (read-only reflection) was used to discover the functional modules and components of the framework. Structural reflection (modification of the structure of objects and classes) allowed the adaptation running services. Dynamic code generation permitted hot-reprogramming of the robot, and the generation of proxy classes perform a bidirectional communication with both local and remote devices. Finally, behavioral reflection allowed us to extend the semantics of the message passing mechanism, simulating methods that translate their invocations into SOAP message calls to remote Web services.

The dynamic type inference and type checks performed by dynamic languages commonly involve a runtime performance penalty. For this reason, we designed some performance optimizations based on type inference of dynamically typed code [4]. These optimizations involved a performance benefit from 141% to 880% [4].

References

[1] M. Garcia, D. Llewellyn-Jones, F. Ortin and M. Merabti, Applying dynamic separation of aspects to distributed systems security: A case study, *IET Software* **6** (2012), 231–248.
[2] M. Garcia, F. Ortin, D. Llewellyn-Jones and M. Merabti, A performance cost evaluation of aspect weaving, in: *36 Australian Computer Science Conference*, 2013, pp. 79–86.
[3] S. Mendez, F. Ortin, M. Garcia and V. García, Computational reflection in order to support context-awareness in a robotics framework, in: *International Conference on Software Engineering & Knowledge Engineering*, 2011, pp. 533–538.

[4] F. Ortin and M. Garcia, Union and intersection types to support both dynamic and static typing, *Information Processing Letters* **111** (2011), 278–286.

[5] F. Ortin, S. Mendez, V. Garcia-Diaz and M. Garcia, On the suitability of dynamic languages for hot-reprogramming a robotics framework: A Python case study, *Software: Practice and Experience* (2013), 1–28.

[6] F. Ortin, D. Zapico, J.B.G. Perez-Schofield and M. Garcia, Including both static and dynamic typing in the same programming language, *IET Software* **4** (2010), 268–282.