Chapter 99
Towards a Stepwise Variability Management Process for Complex Systems:
A Robotics Perspective

Alex Lotz
University of Applied Sciences Ulm, Germany

Matthias Lutz
University of Applied Sciences Ulm, Germany

Juan F. Inglés-Romero
Universidad Politécnica de Cartagena, Spain

Cristina Vicente-Chicote
QSEG, Universidad de Extremadura, Spain

Dennis Stampfer
University of Applied Sciences Ulm, Germany

Christian Schlegel
University of Applied Sciences Ulm, Germany

ABSTRACT

Complex systems are executed in environments with a huge number of potential situations and contingencies, therefore a mechanism is required to express dynamic variability at design-time that can be efficiently resolved in the application at run-time based on the then available information. We present an approach for dynamic variability modeling and its exploitation at run-time. It supports different developer roles and allows the separation of two different kinds of dynamic variability at design-time: (i) variability related to the system operation, and (ii) variability associated with QoS. The former provides robustness to contingencies, maintaining a high success rate in task fulfillment. The latter focuses on the quality of the application execution (defined in terms of non-functional properties like safety or task efficiency) under changing situations and limited resources. The authors also discuss different alternatives for the run-time integration of the two variability management mechanisms, and show real-world robotic examples to illustrate them.

1. INTRODUCTION

Service robots (e.g., companion, elder care, home health care, or co-worker robots) are expected to robustly and efficiently fulfill different tasks in complex environments (such as domestic, outdoor or crowded public spaces). Real-world environments are inherently open-ended and show a huge number of variants and contingencies. Thus, service robots need to know how to flexibly plan their sequence of actions in order to fulfill their tasks, taking into account changes in the environment, noisy perception and execution failures. For instance, a robot navigating in a building needs to reactively avoid dynamic obstacles in its local surrounding. Besides, it might need to reconsider its plan of how to get to its destination, e.g., in case the planned route is blocked. The management of this *variability in operation* provides the robot with a high degree of robustness and allows it to maintain a high success rate in task fulfillment. On the other hand, there is typically a wide range of possibilities to succeed in the same task. Among these possibilities, some might be better than others according to quality criteria defined by the designer (e.g., in terms of resource consumption, safety, performance, etc.). For instance, since robots are equipped with limited resources, they need to know how to spend them in the most appropriate way. Thus, if a robot is running out of battery, it might be a good idea to prioritize power consumption over task efficiency. The management of this *variability in quality* improves the overall robot execution performance.

Addressing dynamic variability in robotics software is nothing new. However, it has been traditionally managed in an ad-hoc way, that is, developers try to predict future execution conditions and implement specific mechanisms to deal with each particular situation, spreading the variability management rationale throughout their application code. Among other issues, this usually leads to increased complexity, poor reuse, and it hinders the extensibility and maintenance of the applications. Explicitly addressing variability modeling in the development processes allows expressing which of the involved stakeholders should be exposed to which variability and which stakeholder should be allowed to exploit which variability. Thus, support for variability modeling in the development process is an essential ingredient for achieving a separation of roles and concerns between the various stakeholders and also enables model reuse. Finally, when it comes to a systematic design and stepwise refinement, it is important that the decision space is defined at design-time and handed over to the run-time system. The purposefully left open variability can then be exploited by the system at run-time. This motivates our approach (Lotz et al., 2013): (i) we need to make it as simple as possible for the designer to express variability at design-time, and (ii) we need the robot to be able to autonomously bind variability at run-time, based on the then available information.

Figure 1. Design-time and run-time variability management in complex systems

Related Content

Biological Traits in Artificial Self-Reproducing Systems
Eleonora Bilotta and Pietro Pantano (2012). *International Journal of Signs and Semiotic Systems (pp. 69-83).*
www.igi-global.com/article/biological-traits-in-artificial-self-reproducing-systems/101252?camid=4v1a

Privacy-Friendly Wi-Fi-Based Occupancy Estimation with Minimal Resources
E. Makri, J. ten Brinke, R. Evers, P. Man and H. Olthof (2018). *International Journal of Ambient Computing and Intelligence (pp. 34-51).*
www.igi-global.com/article/privacy-friendly-wi-fi-based-occupancy-estimation-with-minimal-resources/211171?camid=4v1a

Time-Slot Based Intelligent Music Recommender in Indian Music
Sudipta Chakrabarty, Samarjit Roy and Debashis De (2017). *Intelligent Analysis of Multimedia Information (pp. 319-351).*
www.igi-global.com/chapter/time-slot-based-intelligent-music-recommender-in-indian-music/159442?camid=4v1a

Role of Distance Metric in Goal Geometric Programming Problem (G^2 P^2) Under Imprecise Environment
Payel Ghosh and Tapan Kumar Roy (2019). *International Journal of Fuzzy System Applications (pp. 65-82).*
www.igi-global.com/article/role-of-distance-metric-in-goal-geometric-programming-problem-g2-p2-under-imprecise-environment/214940?camid=4v1a