

Preface

In order to address today's societal challenges (e.g., environmental management, intelligent mobility, safety/security, sustainable urban living, etc.) monitoring and control of large-scale, complex, dynamical systems are essential technologies. These monitoring and control systems realize sophisticated signal processing algorithms (state estimation, pattern recognition, optimal control, decision-making, etc.) in distributed configurations and many times are deployed in difficult-to-access, hostile environments. Under these circumstances failures, changes in operational conditions and changes in user needs are not exceptions but should be considered as nominal operation. Runtime reconfiguration capabilities (covering self-organization, self-optimization) have to be realized to extend the operational envelope of the system. Runtime reconfiguration enables the system to cope with changing system configurations without re-programming the existing setup (i.e., supporting easy deployment), to support mobile subsystems (dynamic sensor management), to cope with environmental changes (e.g., changing network capacities) and to adapt to changing system goals (user needs) during operation. The realization and operation of runtime reconfigurable systems poses new challenges in every stage of the system's life cycle.

In the book the design and testing of large-scale, distributed signal processing systems are considered with special emphasis on systems architecture, tooling, and best practices. Distinguishing features of the targeted systems are serious resource constraints (e.g., processing capabilities, communication, energy supply), and the presence of demanding nonfunctional requirements such as timeliness, robustness, lifetime, and the capability of handling system "evolution". All along a strict model-based process is followed and thus the architecture modeling, model checking, model-based evaluation, and model-based design optimization play a central role in the book. After summarizing the conceptual/theoretical foundation of the methodology, a systems engineering process and a matching tool suite is described. In order to assure the landing of the abstract concepts, we shall use a number of nontrivial examples, taken from real-world cases, to illustrate the application of theory to practice.

The book is intended to be used as a cookbook for designers and practitioners of complex embedded systems: though it covers the relevant fundamentals for designing and testing of large-scale networked embedded systems, it also proposes systems engineering processes, architectural templates (design patterns), and practical testing/validation approaches and thus it serves as a guideline for the system designer.

The book captures the expertise of several authors who have worked for years at the interface between theory and engineering in a variety of topical issues of embedded systems. Since 2012 these authors have come to work together under the umbrella of the European project ARTEMIS DEMANES (Design, Monitoring and Operation of Adaptive Networked Embedded Systems) that has at its heart the ambition to define a methodology and to realize tool chain for high-level support for designing, implementing, testing, and deploying real-world runtime reconfigurable applications. This book collects the most relevant developments by DEMANES members in the area of self-configurable embedded networks and systems. This manuscript is presenting seven complementary topics relevant to the requirements specification, the off-line design methodology, the services and tooling used, the validation and verification framework, the validation tools and, finally, the applicability to real-world pilots of large-scale runtime-adaptive networked embedded systems.

In Chap. 1, a multi-aspect modeling language is introduced that allows system designers to model the architecture of large-scale networked systems from different aspects and contains concepts to model runtime reconfiguration at design-time. The proposed architecture for modeling runtime reconfiguration consists of primary tasks in one layer and secondary management tasks in another layer. Special reconfiguration primitives allow the description of four types of reconfiguration: re-parameterization, re-instantiation, rewiring, and relocation. The modeling language is accompanied by a modeling methodology based on MAPE-K and uses feedback loops in the system model to model runtime reconfiguration. System-level characteristics, i.e., KPIs are introduced allowing system designers to evaluate system designs and pick the most promising one. Special attention is paid to the fact that the availability of a runtime redesign capability in a system requires KPIs to be derived and evaluated at runtime. Only then can the redesign be guided properly.

Chapter 2 discusses design methodologies that are specific for adaptive networked embedded systems. This chapter discusses design-time/runtime trade-offs, design patterns for reconfigurable real-time monitoring and control, runtime design space exploration of adaptive systems and a systems engineering process for runtime reconfigurable systems. We draw guidelines for all stages of the architectural process and help system and software designers in choosing wisely specific algorithms and techniques. In summary, we showcase state-of-the-art design techniques that are directly addressing the adaptivity.

Chapter 3 discusses how to build up a model-oriented tool chain and its most important activity flows: model editing, model visualization and transformation; model validation and evaluation; and finally, (semi-)automatic system implementation based on the system model. In addition, for each activity emphasis goes also

to the specific tool support which is necessary for the design of adaptive/reconfigurable systems.

Chapter 4 recognizes and analyzes the need for runtime validation and verification of large-scale adaptive networked embedded systems (ANES). It is not feasible, during the development stages, to anticipate all the possible operating conditions that the system may face in a real environment. This is because some information about the execution context and the system itself can be available only once the system has been deployed. Thus, in order to correctly assess the effectiveness, efficiency and robustness of ANES, it is required to verify that the system correctly adopts the proper adaptation mechanisms in response to the context changes as well as to check the quality of such adaptations. The focus of the chapter is to discuss about the needs for employing runtime verification and validation of ANES and the main challenges and requirements for its implementation. Moreover, it presents a reference framework that supports developers in testing adaptive systems at runtime. One of its key features is the capability to emulate certain realistic conditions through synthetic data, which is useful to check the system's behavior under specific and controlled situations.

Chapter 5, symmetrically to Chap. 3, is a thorough analysis of the state-of-the-art tooling available for validating and verifying at runtime large-scale networked embedded systems. In cases where the long-term functioning of the systems is of interest, the systems quality should prove very high and, therefore, proper validation and verification practices are required. Given the lack of relevant tools that deal with testing runtime self-adaptive systems, this chapter proposes the implementation of a V&V framework introduced by Chap. 4, by merging several already known tools. First, it gives an understanding of ways to quantify and predict the reliability of large-scale distributed systems. Second, key performance indicators of the self-adaptive systems are identified from monitoring techniques and third, the test cases are formalized in a structured form. Finally, the chapter presents two test cases as examples of a system working under normal operation conditions as well as under induced conditions, based on real-life implementations. Execution of the test is led by a test coordinator for which JSON notation is used, and then the interpretation and testing is carried out in a C++ toolbox where the monitoring and testing algorithms reside.

This last chapter describes a real self-adaptive system carried out using the DEMANES tool chain based on the foundations of the DEMANES tool chain described from previous chapters. This chapter focuses on the design and implementation stages of a real use case development, a pilot. The use case under study is a subsystem called cargo monitoring system (CMS), which monitors the state of the container cargo and push all the data to a back office infrastructure for further processing. The containers can be on a truck, a train, or any other appropriate transportation means, or stacked in a container terminal or a cargo ship. A WSN will measure physical magnitudes (temperature, humidity and so on) inside a container and will forward data to others processing nodes in the CMS network. The CMS has to meet several self-adaptive requirements. For instance, the parameters of the CMS elements that monitor the container cargo state are

reconfigured accordingly to adapt to internal or external changes (e.g., a low battery level or a container temperature out of the adequate bounds), and the CMS adapts the WSN nodes power transmission to save energy while providing an acceptable quality of service.

Delft, The Netherlands
Eindhoven, The Netherlands
June 2015

Zoltan Papp
George Exarchakos

Runtime Reconfiguration in Networked Embedded
Systems

Design and Testing Practices

Papp, Z.; Exarchakos, G. (Eds.)

2016, XXII, 171 p. 85 illus., 62 illus. in color., Hardcover

ISBN: 978-981-10-0714-9