Chapter 8
Development of Controllers Using Simulink and Contract-Based Design

ABSTRACT

This chapter describes the application of contract-based design in Simulink to the development of a digital hydraulic controller. A semi-formal approach to the development is used, where contracts are used to aid the structuring of the system and for creating interface specifications for components that can later be used for testing. An analysis of the influence of the contracts on the development process and system quality is provided to evaluate the effectiveness of the method. It was concluded that contract-based design can significantly aid the development of high quality Simulink models.

INTRODUCTION

Economical development of reliable control software is becoming more important as software is becoming a more integrated part of many mechanical applications. This chapter describes a method for the development of control systems, as well as an evaluation of the method on the development of a part of a controller for a digital hydraulic system. Digital hydraulics requires sophisticated controllers to achieve the best possible performance and to implement advanced features, such as energy saving. The software of the controllers is therefore highly complex. To handle the complexity and to
ensure that the developed system is reliable and functions correctly, adequate software development techniques need to be used.

One approach that has been developed to meet the challenges of embedded control system design is model-based design. In this approach, a simulation model of the system to be controlled is created. This enables fast and cheap evaluation and validation of the controller. Simulink (Mathworks, 2010) is a graphical language that has become one of the most popular languages for model-based design of control systems. This is mainly due to user friendly modelling notation, large library of ready-made components and good simulation tools for testing and validation. Therefore Simulink has also been used as a tool to develop digital hydraulic systems for several years. However, Simulink lacks tools and concepts for stepwise design and modular techniques to analyse system correctness. These features would be desirable to better reason about the correctness of the constructed system in a more manageable way and thereby increase the reliability of the constructed systems. We have taken a contract-based approach to address the deficiencies above.

Contracts here refer to pre- and post-conditions for programs or program fragments (Meyer, 1997; Morgan, 1990). They provide rules for dividing functionality into components and rules for analysis of correctness. Contract-based design is a well-known technique for object-oriented software development (Barnett, Leino, & Schulte, 2004; Burdy, et al., 2005; Meyer, 1997). We have proposed a stepwise development method and verification techniques based on contracts for Simulink (Boström, Linjama, Morel, Siivonen, & Waldén, 2007; Boström, 2008). To analyse Simulink models for correctness with respect to contracts, a formal notion of correctness is needed. The analysis methods we have developed are based on a translation of the models to action systems (Back & Sere, 1991; Back & von Wright, 1994). Using this approach, the well known formal reasoning techniques for action systems, which are based on the refinement calculus (Back & von Wright, 1998), can be used for analysis (Boström et al., 2007; Boström, 2008; Boström, Grönblo, Huotari, & Wiik 2010). Formalisations of Simulink have also been investigated in other notations. There is a translation to Lustre (Tripakis, Sofronis, Caspi, & Curic, 2005) that can handle a large subset of Simulink. Another formalisation is a translation to Circus (Cavalcanti, Clayton, & O’Halloran, 2005). Neither of those works considers contracts as an integrated part of the formalisation. However, contracts could be introduced there also. Simulink itself also has several toolboxes for verification and validation (Mathworks, 2010). Even formal verification is now supported via the Simulink design verifier toolbox. Compositional reasoning about system correctness as done using contracts is not directly supported there. However, the contracts together with our formal analysis methods give correctness conditions that can be checked or validated using also those tools.

This chapter presents a practical application of contracts in Simulink to the development of a part of a controller for a digital hydraulic system. Contracts are used as an aid for structuring the system, as well as for analysis in the form of testing and design reviews. The control engineers benefited from the contract-based design by increasing their understanding of the developed system and by getting clear correctness conditions to test the system against. The contribution of this chapter is a demonstration of how contracts can be used to specify and document a fairly large and complex control system developed using a dataflow language. The case study shows that a high quality of the resulting control software is achieved using this approach. As the case study was carried out by mechanical engineers, it also shows that the contract-based approach can be applied without an extensive training in computer science and formal methods. The chapter starts with a description of the case study. Then a short presentation of Simulink and contracts is given,
Related Content

Blackout Avoidance and Energy Saving Services with Alert Aggregation
[www.igi-global.com/article/blackout-avoidance-and-energy-saving-services-with-alert-aggregation/89387?camid=4v1a](www.igi-global.com/article/blackout-avoidance-and-energy-saving-services-with-alert-aggregation/89387?camid=4v1a)

Towards Method Component Contextualization
[www.igi-global.com/article/towards-method-component-contextualization/58645?camid=4v1a](www.igi-global.com/article/towards-method-component-contextualization/58645?camid=4v1a)

Using Model-Driven Architecture Principles to Generate Applications based on Interconnecting Smart Objects and Sensors
Cristian González García and Jordán Pascual Espada (2014). *Advances and Applications in Model-Driven Engineering* (pp. 73-87).
[www.igi-global.com/chapter/using-model-driven-architecture-principles/78611?camid=4v1a](www.igi-global.com/chapter/using-model-driven-architecture-principles/78611?camid=4v1a)

Interactive Analysis of Agent-Goal Models in Enterprise Modeling
[www.igi-global.com/article/interactive-analysis-agent-goal-models/47383?camid=4v1a](www.igi-global.com/article/interactive-analysis-agent-goal-models/47383?camid=4v1a)