Chapter 19
The Formal Design Model of a Lift Dispatching System (LDS)

Yingxu Wang
University of Calgary, Canada

Cyprian F. Ngolah
University of Calgary, Canada

Hadi Ahmadi
University of Calgary, Canada

Philip Sheu
Univ. of California, Irvine, USA

Shi Ying
Wuhan University, China

ABSTRACT

A Lift Dispatching System (LDS) is a typical real-time system that is highly complicated in design and implementation. This article presents the formal design, specification, and modeling of the LDS system using a denotational mathematics known as Real-Time Process Algebra (RTPA). The conceptual model of the LDS system is introduced as the initial requirements for the system. The architectural model of the LDS system is created using RTPA architectural modeling methodologies and refined by a set of Unified Data Models (UDMs). The static behaviors of the LDS system are specified and refined by a set of Unified Process Models (UPMs) for the lift dispatching and serving processes. The dynamic behaviors of the LDS system are specified and refined by process priority allocation and process deployment models. Based on the formal design models of the LDS system, code can be automatically generated using the RTPA Code Generator (RTPA-CG), or be seamlessly transferred into programs by programmers. The formal models of LDS may not only serve as a formal design paradigm of real-time software systems, but also a test bench of the expressive power and modeling capability of exiting formal methods in software engineering.

DOI: 10.4018/978-1-4666-0261-8.ch019

INTRODUCTION

A real-time system is characterized by event-/time-/interrupt-driven behaviours that are constrained by both its logic correctness and timing correctness. Although nonreal-time transaction processing systems may only consider the logical correctness, real-time systems have to put emphases on dynamic timing constraints with system control logics. A Lift Dispatching System (LDS) is a typical real-time control system characterized by its high degree of complexity, intricate interactions with hardware devices and users, and necessary requirements for domain knowledge (Hayes, 1985; McDermid, 1991; Chenais & Weinberger, 1992; Liu, 2000; Wang, 2002, 2007; Ngolah et al., 2004). All these factors warrant an LDS system as a complex but ideal design paradigm in large-scale software system design in general and in real-time system modeling in particular.

The lift scheduling problem has been studied as a real-time system in Chenais and Weinberger (1992) and Hamdi et al. (1995) to be NP-complete, because for n lifts, if there are p requests, then there would be upto n^p possible dispatching strategies. Further, the problem is dynamic, i.e., during executing a given dispatching plan, new requests presented inside the cabins of lifts and from the floors may often interrupt and change the current dispatching strategy. The request scheduler therefore must be able to find a suitable dispatching mechanism in order to ensure there is no request to wait for a long period before being served.

There is no systematical and detailed repository and formal documentation of design knowledge and modeling prototypes of an LDS system nor a formal model of it in denotational mathematics and formal notation systems (Wang, 2008d). This article presents the formal design, specification, and modeling of the LDS system using a denotational mathematics known as Real-Time Process Algebra (RTPA) (Wang, 2002, 2003, 2007, 2008a, 2008b). RTPA introduces only 17 meta-processes and 17 process relations to describe software system architectures and behaviors with a stepwise refinement methodology (Wang, 2007, 2008a, 2008c). According to the RTPA methodology for system modeling and refinement, a software system can be specified as a set of *architectural* and *operational components* as well as their interactions. The former is modeled by *Unified Data Models* (UDMs, also known as the component logical model (CLM)) (Wang, 2007), which is an abstract model of the system hardware interface, an internal logic model of hardware, and/or a control structure of a system. The latter is modeled by *static* and *dynamic processes* using the *Unified Process Models* (UPMs) (Hoare, 1978, 1985; Bjorner & Jones, 1982; Corsetti & Ratto, 1991; Wang, 2007, 2008a; Wang & King, 2000; Wang & Ngolah, 2002).

This article develops a formal design model of the LDS system in a top-down approach on the basis of the RTPA methodology. In the remainder of this article, the conceptual model of the LDS system is described as the initial requirements of the system. The architectural model of the LDS system is created based on the conceptual model using the RTPA architectural modeling methodologies and refined by a set of UDMs. Then, the static behaviors of the LDS system are specified and refined by a set of processes (UPMs). The dynamic behaviors of the LDS system are specified and refined by process priority allocation, process deployment, and process dispatching models. With the formal and rigorous models of the LDS system, code can be automatically generated by the RTPA Code Generator (RTPA-CG) (Wang, 2007), or be seamlessly transferred into program code manually. The formal models of LDS may not only serve as a formal design paradigm of real-time software systems, but also a test bench of the expressive power and modeling capability of exiting formal methods in software engineering.