Smart System Design Using Hybrid Models

Ravi Kapadia
Dept. of CS, Vanderbilt University,
Box 1679 Station B
Nashville, TN 37235.
ravi@vuse.vanderbilt.edu

"Smart" electromechanical systems are machines with intelligent, embedded computer control such as washing machines, car anti-skid braking systems and reprographic machines (e.g., printers, photocopiers, and fax machines). Conventional embedded controllers performed a pre-programmed set of actions, but as machines have become more versatile, controllers are being designed to generate and evaluate different plans to perform specified tasks, search for optimal solutions, and react to changing environments.

Embedded system design is a difficult problem because the designer must make decisions about allocating functionality between hardware and software, and ensure that the combined design functions correctly and optimally. When the control software must search a large space of possible actions to discover an optimal one in a very short amount of time, as may be the case with smart systems, the design task becomes even more challenging. Thus, the designer must carefully choose a hardware configuration, computational resources, and algorithmic strategies to develop an optimal design.

Traditionally, hardware design has preceded software design resulting in sub-optimal designs. With the increasing use of intelligent, embedded software, integrated hardware software design becomes critical. Hardware software codesign of embedded systems is prevalent in the electronics domain (Wolf, 1994) but the focus of design research in the AI community continues to be hardware oriented.

In order to codesign the hardware and software, we must develop representations of hardware and software at a level of abstraction that suits our design goals. (Fromherz and Carlsson, 1994) have developed discrete hardware models to schedule sheets in reprographic machines. We propose the development of a uniform modeling scheme that integrates discrete hardware and software models to aid in the task of generating and evaluating design alternatives for smart systems.

The Environment Relationship (ER) Net (Ghezzi et al., 1991) framework, an extension of basic Petri nets (Peterson, 1981), allows the explicit representation of time, which is necessary for modeling system optimization problems, and associates properties with tokens, which is essential for modeling systems with heterogenous job elements (sheets in the reprographic machine domain). (Kapadia, Biswas, and Fromherz, 1997) adapt the ER net formalism to model the subsystem that transports paper and images within a reprographic machine. Our goal is to use ER nets to build a combined model of hardware and software elements. A system designer may mix-and-match different hardware and software elements and simulate the integrated model to analyze its performance, cost, and reliability. Adopting an integrated approach over the present, sequential policy, will improve the quality and reduce the time taken to design complex, smart systems.

References

Fromherz, M. and Carlsson, B. 1994. Optimal Incremental and Anytime Scheduling. Proc. Workshop on Constraint Languages/Systems and their Use in Problem Modeling at ILPS'94, 45–59. ECRC, TR 94-38.

Ghezzi, C., Mandrioli, D., Morasca, S., and Pezze, M. 1991. A Unified High-level Petri Net Formalism for Time Critical Systems. In IEEE Transactions on Software Engineering 17(2),160–172.

Kapadia, R., Biswas, G., and Fromherz, M. 1997. Hybrid Modeling for Smart System Design. The 10th International FLAIRS Conference, Daytona Beach, FL.

Peterson, J. 1981. *Petri Net Theory and the Modeling of Systems.* Englewood Cliffs, NJ: Prentice Hall.

Wolf, W. 1994 Hardware-software Co-design of Embedded Systems. In Proc. of the IEEE, 82(7):965–989.