From Logic Specification to $\gamma$-Calculus:
A Method for Designing Multiagent Systems

Hong Lin, University of Houston-Downtown, USA

ABSTRACT

A program construction method based on $\gamma$-calculus is proposed: The problem to be solved is specified by first-order predicate logic, and a semantic verification program is constructed directly from the specification. We exploit this method in synthesizing the architectural specifications of multiagent systems (MAS) in $\gamma$-calculus based on the logic specifications of the MAS. By enabling the transformation from the logic specifications to operational specifications of MAS, this method allows the design of the MAS to be focused on the architectural definition level. It benefits the development of MAS by enabling logic deduction on behaviors of the MAS, and a design methodology in an incremental fashion.

Keywords: chemical reaction model; logic specifications; multiagent systems; program synthesis; program transformation; software architectures

INTRODUCTION

The modeling issue in the abstract computing machine level has been studied in Banâtre, Fradet, and Radenac (2004), where the chemical reaction model (Banâtre, Fradet, & Radenac, 2005b; Banâtre & Le Metayer, 1990, 1993; Le Metayer, 1994) is used to model an autonomic system. Given the dynamic and concurrent nature of multiagent systems (MASs), we find that the chemical reaction metaphor (CRM) provides a mechanism for describing the overall architecture of the distributed multiagent
systems precisely and concisely while giving the design of the real system a solid starting point, and allowing step-by-step refinement of the system using transformational methods (H. Lin, 2004; H. Lin & Yang, 2006).

Although CRM is suitable for modeling MAS, it serves as an operational specification language for MAS and requires the designers of MAS to have an understanding of chemistry-inspired computational models. As a matter of fact, logic specification for MAS is better suited as a specification method in the current understanding because logic specification focuses on behavioral properties of the systems without concern for the underlying computational model. We propose a method for generating a MAS’s specifications in \( \gamma \)-calculus from its logic specifications. We use the “generate-and-test” method to design the rewrite process. Generally speaking, this process generates data in the domain of logical specification and creates a verification program in \( \gamma \)-calculus to verify the logical specification with the generated data.

When applying this method to multi-agent systems, there are some problems to solve. The architectural specification of a multiagent system is different from that of a normal program because for a multiagent system, we need to consider a collection of aspects, including distribution, security, performance, and so forth. This will cause a much more complex synthesizing process. For example, the distribution aspect will cause the communication pattern to be considered in the synthesizing process. In our approach, the communications are defined by the logical specifications of the interfaces of the system in terms of either message passing or shared memory. The practicability of this method is further strengthened by a transformation method we have proposed to implement CRM specifications on realistic computational models (H. Lin, 2004; H. Lin & Yang, 2006).

Multi-agent systems are considered complex systems whose design issues are difficult to be handled by logic systems. By bridging logic specifications and operational specifications of multiagent systems, our study opens a path to introducing derivative methods in the higher level architectural design of multiagent systems. This work will help formalize the design processes and promote the current research endeavor to end the state of MAS design on case-by-case fashion.

**Modeling Multiagent Systems by Chemical Reaction Models**

Gamma (Banâtre & Le Metayer, 1990, 1993) is a kernel language in which programs are described in terms of multiset transformations. In the Gamma programming paradigm, programmers can concentrate on the logic of problem solving based on an abstract machine and are free from considering any particular execution environment. It has seeded follow-up elaborations, such as the chemical abstract machine (Cham; Berry & Boudol, 1992), higher order Gamma (Le Metayer, 1994), and...
Related Content

Ubiquitous Mediation and Critical Interventions: Reflections on the Function of Signs and the Purposes of Peirce’s Semeiotic
[www.igi-global.com/article/ubiquitous-mediation-critical-interventions/56444?camid=4v1a](www.igi-global.com/article/ubiquitous-mediation-critical-interventions/56444?camid=4v1a)

Investigation of ANFIS and FFBNN Recognition Methods Performance in Tamil Speech Word Recognition
Timing Matters: Dynamic Interactions Create Sensitive Periods for Word Learning
[www.igi-global.com/chapter/timing-matters-dynamic-interactions-create/74888?camid=4v1a](www.igi-global.com/chapter/timing-matters-dynamic-interactions-create/74888?camid=4v1a)

Planning a Fourth Industrial Revolution Organization: Alternative Process Considerations
(2018). *Organizational Leadership for the Fourth Industrial Revolution: Emerging Research and Opportunities* (pp. 57-69).
[www.igi-global.com/chapter/planning-a-fourth-industrial-revolution-organization/198279?camid=4v1a](www.igi-global.com/chapter/planning-a-fourth-industrial-revolution-organization/198279?camid=4v1a)