博士研究生学位论文

题目：**协调与组合：从 Reo 到 Mediator**

姓　　名： 李屹

学　　号： 1501110045

院　　系： 数学科学学院

专　　业： 应用数学

研究方向： 程序理论、软件形式化方法

导　　师： 孙猛教授

二〇一九年六月

版权声明

摘要

近年来，软件系统在我们日常生活中占据的比重不断增加。与此同时，软件自身的复杂性也在呈指数级增长。在这样的开发环境和迭代速度下，如何保持软件的可靠性成为了一项重要的课题。在学术界和工业界所提出的各种方法之中，基于组件的开发方法已经得到了广泛使用，成为了现代软件开发中的一种常用模式。基于组件的开发方法以组件重用为核心，将复杂系统拆分为简单功能单元，并分别对其进行实现。由于可重用的组件在开发过程中往往需要经过更为严格的验证和测试，并且在重用过程中组件的可能缺陷和错误可以被及时发现和排除，因此其质量能够得到更好的保证。除此之外，基于组件的开发过程可以有效地降低开发成本，大幅提升开发的效率。另一方面，形式化方法基于严格的数学基础，对计算机软硬件系统进行严格的描述、开发和验证，已经愈来愈多地应用到各个领域，并受到广泛关注。

值得注意的是，基于组件的开发方法与形式化方法有天然的契合性。通过将大型系统组件化，软件单元的规模得以降低，使单个组件的验证得到一定的简化；而通过形式化验证之后的组件则可以‘‘可信组件’’的形式在多个系统中被重用，更好地保证系统的正确性、可靠性。在本文中，我们围绕两种形式化语言及其背后的“协调”与“组合”概念，在形式化建模、验证与代码生成方面进行研究。

在本文的第一部分工作中，我们通过协调语言 Reo 对基于组件开发方法中“协调”概念进行刻画。首先，我们在高阶逻辑定理证明器 Coq 中定义了 Reo 基本信道的语义与连接件的组合算子，并通过交互式策略对连接件性质进行证明。为了使 Reo 能够刻画更多复杂场景下的协调行为，我们对 Reo 进行了概率与时间维度上的扩充。在此扩充中，我们增加了映射信道 (Map)，随机选择信道 (StochasticChoice) 以及带参数的计时信道 (pTimer)，使得扩展后的 Reo 语言能够表示带有随机行为和实时行为的连接件。扩充后的连接件形式语义由随机时间 Reo 自动机 STA_r(Stochastic Timed Automata *for Reo*) 刻画。我们同时也提供了基于 STA_r 的连接件组合算子的形式化定义并证明了其良定义性。我们通过一个 Python 工具包给出了扩充语言形式语义及组合算子的实现，该工具包同时还可以通过 PRISM 模型检查工具对构造的连接件进行验证。

由于 Reo 语言主要关注不同组件之间的协作，而在刻画参与协作的组件本身的行为方面较为薄弱，在本文的第二部分中我们提出了一个新的领域建模语言 *Mediator*。该语言从底层自动机角度和高层系统角度分别对组件的内部行为以及组件之间的组合方式进行描述。该语言的形式语义由带标签的状态转移系统进行定义，同时我们可以用 CTL* 来定义 *Mediator* 模型的性质 (同时支持计算树逻辑公式与线性时序逻辑公式)，并

对 *Mediator* 中定义的模型进行验证。在后续的工作中，我们对 *Mediator* 进行了分布式扩展，通过加入时间语义以及事务模型，可用 *Mediator* 语言描述分布式系统中的行为。

为了进一步将经过形式化验证的组件系统进行落地，我们开发了基于 *Mediator* 的代码自动生成系统，可以将经过验证的 *Mediator* 模型自动生成为 Arduino, System C 等平台上可以直接执行的程序代码。

关键词： 协调语言，基于组件的开发方法, Reo, Mediator, 形式化验证, 代码自动生成

Coordination and Composition : From Reo To Mediator

Yi Li (Applied Mathematics)
Directed by Prof. Meng Sun

ABSTRACT

Modern software systems are playing more and more important roles in our daily lives. To satisfy the requirements of the public, the complexity of software systems is also increasing rapidly. Under such situations, maintaining the reliability of software systems has become a challenging task. To deal with this problem, various approaches have been proposed from both the academic and the industrial worlds.

On one hand, *component-based development* has been widely used in practice. In component-based development, developers encapsulate the reusable features into components. To ensure reliability, such components are usually strictly tested or verified. Moreover, as reused in many larger projects, their vulnerabilities and errors are more likely to be spotted and corrected. Consequently, component-based development has successfully improved the developing efficiency and reliability of various software projects. On the other hand, *formal methods* focus on precisely specifying, developing and verifying software and hardware systems based on mathematical foundations. In recent years, formal methods have attracted extensive attention from different areas.

We have noticed that component-based development and formal methods are born cooperatively. Component-based development significantly simplifies the development of large-scale software projects by splitting them into small components which are easier to formalize and verify. Meanwhile, the formally verified components can be reused in various projects, improving their reliability without extra cost. In this thesis, our work on modeling and verifying component-based software mainly focuses on two formal languages, inspiring two different but highly related aspects in component-based software development: *Reo* $\mapsto$ coordination, and *Mediator* $\mapsto$ composition. Multiple aspects in formal engineering are investigated, including formal modeling, verification and code generation.

First, we formalize the coordination behavior in component-based development using Reo, a channel-based exogenous coordination language where concurrency protocols are manifested as connectors. The primitive channels and the composition operators in Reo are defined in the

Coq proof assistant in the beginning. With the help of interactive pattern and tactic searching, Coq is able to prove various properties of the connectors. Moreover, we extend Reo in order to capture stochastic and timed behavior through a mapping channel, a stochastic choice channel and a parametric timed channel. STA_r is proposed as the semantics model of both original Reo channels and the extended channels, and $PTCTL^*$ is used to describe the properties of stochastic timed connectors. To bring this framework into practice, we have implemented a Python toolkit that can build stochastic and timed connectors, automatically generate the semantics based on the composition operators and verify their properties through the PRISM model checker.

Second, we inherit the basic ideas from Reo and propose a new language called *Mediator*. Reo is aimed at coordination between components, and pays little attention to the behavior inside the components. *Mediator* provides proper formalism for both high-level *system* layouts and low-level *automata*-based behavior units. With its semantics defined by labeled transition systems and properties formalized by CTL*, we can verify the properties of *Mediator* models by using model checking. Then we extend *Mediator* to capture complex behavior in distributed computing. Clocks and transactions are used to describe the communication between different entities in this case. We also provide a formal semantics for the extension for further verification attempts.

Finally, to reduce the software errors imported in the implementation phase, we propose a code-generation framework based on *Mediator*. Currently, the framework supports the open-source hardware platform *Arduino C*. With *Mediator* models provided, our algorithm automatically schedules and generates the controller (as *Mediator* entity) to a runnable source-code archive.

KEYWORDS: Coordination, Composition, Reo, Mediator, Formal Verification, Code Generation

Contents

在读期间论文发表情况

在本部分列出的所有论文均按照出版（接受）时间排序。

已发表论文

1. 张琦，李屹，孙猛. Mediator 的 SystemC 代码自动生成.《计算机工程与科学》, 41(5)，835-842，2019.
2. Yi Li, Xiyue Zhang, Yuanyi Ji and Meng Sun. A Formal Framework Capturing Real-Time and Stochastic Behavior in Connectors. Science of Computer Programming，vol. 177, pages 21-40, 2019.
3. Xiyue Zhang, Weijiang Hong, Yi Li and Meng Sun. Reasoning about Connectors Using Coq and Z3. Science of Computer Programming, vol. 170, pages 27-44, 2019.
4. Ai Liu, Shun Wang, Yi Li and Meng Sun. On Semantics for Mediator: A Coalgebraic Perspective. in Proceedings of SOFL+MSVL 2018, LNCS 11392, pages 146-165, Springer, 2019.
5. Yi Li. Developing Reliable Component-Based Software in Mediator. In Proceedings of ICFEM 2018, pages 432-435, LNCS 11232, Springer, 2018.
6. Hongfei Fu, Yi Li and Jianlin Li. Verifying Probabilistic Timed Automata Against Omega-Regular Dense-Time Properties. In Proceedings of QEST 2018, pages 122-139, LNCS 11024, Springer, 2018.
7. Yi Li and Meng Sun. Generating Arduino C codes from Mediator. In It's All About Coordination: Essays to Celebrate the Lifelong Scientific Achievements of Farhad Arbab, pages 174-188, LNCS 10865, Springer, 2018.
8. Weijiang Hong, Saqib Nawaz, Xiyue Zhang, Yi Li and Meng Sun. Using Coq for Formal Modeling and Verification of Timed Connectors. In Proceedings of SEFM 2017 Collocated Workshops, pages 558-573, LNCS 10729, Springer, 2018.
9. Yi Li, Xiyue Zhang, Yuanyi Ji and Meng Sun. Capturing Stochastic and Real-time Behavior in Reo Connectors. In Proceedings of SBMF 2017, pages 287-304, LNCS 10623, Springer, 2017. (本论文获得会议最佳论文奖第一名)
10. Yi Li and Meng Sun. Component-based Modeling in Mediator. In Proceedings of FACS 2017, pages 1-19, LNCS 10487, Springer, 2017.
11. Xiyue Zhang, Weijiang Hong, Yi Li and Meng Sun. Reasoning about Connectors in

Coq. In Proceedings of FACS 2016, pages 172-190, LNCS 10231, Springer, 2017.

12. Yi Li, Meng Sun and Yiwu Wang. Active Learning from Blackbox to Timed Connectors. In Proceedings of TASE 2016, pages 132-135, IEEE, 2016.
13. Kong Pingfan, Yi Li, Xiaohong Chen, Jun Sun, Meng Sun and Wang Jingyi. Towards Concolic Testing for Hybrid Systems. In Proceedings of FM 2016, LNCS 9995, pages 460-478, Springer, 2016.
14. Yi Li and Meng Sun. Modeling and Verification of Component Connectors in Coq. In Science of Computer Programming. vol. 113(3), pages 285-301, 2015.
15. Yi Li and Meng Sun. Modeling and Analysis of Component Connectors in Coq. In Proceedings of FACS 2013, LNCS 8348, pages 273-290, Springer, 2014.
16. Ernst Moritz Hahn, Yi Li, Sven Schewe, Andrea Turrini, and Lijun Zhang. IscasMC: A web-based probabilistic model checker. In Proceedings of FM 2014, pp. 312-317. Springer, 2014.
17. Meng Sun and Yi Li. Formal Modeling and Verification of Complex Interactions in E-Government Applications. In Proceedings of ICEGOV 2014, pages 506-507, ACM, 2014.

已接受论文

1. Yi Li and Meng Sun. Distributed *Mediator*. Accepted by TASE 2019.
2. Xiyue Zhang, Yi Li, Weijiang Hong and Meng Sun. Using Recurrent Neural Network to Predict Tactics for Proving Component Connector Properties in Coq. Accepted by TASE 2019.

致谢

“纸上纵横莫笑痴，余生无复少年时。
春风秋叶无穷意，半寄湖光半作诗。”

恍惚之间，踏进这个园子已有十载。

四年本科，六年硕博，一生中任性驰骋而无所挂碍的岁月。即便今日的北大早已不是那个蔡元培时代的象牙塔，然而对于我等活在其中的人，仍然是包容放浪之形骸，肆意之批判与广博之知识的安乐窝。未名湖安谧的波光之上，数不清的快乐与闲愁弥集于此。然而韶光易逝，离开它的那一刻也终于将要来临。

在这六年之中，我的导师孙猛教授当可谓是对我助益最多且影响最深的人。孙老师于我亦师亦友，前三年师多于友，后三年友多于师。如此良师益友，实为不多得之机缘。学术之路如行于漫漫长夜，从导师处所学，让我渐渐能领会到科研工作中自然的乐趣，以及走在这条道路上踏实的态度。

我要感谢所有与我合作过的老师们，如中科院软件研究所的张立军老师，符鸿飞老师，Ernst Moritz Hahn 以及 Andrea Turrini；新加坡科技设计大学的孙军老师等。在与你们合作进行的研究工作中，我逐渐掌握了进行科研的方法和必需的经验，而你们的学术精神和人格更令我颇有受益。

我要感谢曾经在讨论班，课程等各处曾经对我进行指导的其他老师们：中科院软件研究所的詹乃军老师，王淑灵老师；北京大学数学科学学院的裘宗燕老师，夏壁灿老师，马尽文老师，林作铨老师，杨建生老师，牟克典老师，毛珩老师，甘锐老师；以及在德国萨尔大学交流期间结识的 Holger Hermanns 与 Arnd Hartmanns。你们课上课下的讲授，有意或无意的思想流露，都是我研究工作中重要的组成部分。

此外我还要感谢北京大学数学科学学院的张树义老师。虽然张老师与我所做科研工作关系并不密切，但某种意义上，在张老师带领的网络小组的工作经历让我熟悉了现实工作中的程序设计而非算法领域的程序设计，让我得以拥有“一星期熟悉任何一门常规程序语言”的能力。

我要感谢与我合作或交流的同学们。北京大学数学科学学院的陈霄泓，张喜悦，洪伟疆，冀元祎，刘艾，王顺，张琦，王译梧以及 Muhammad Saqib Nawaz；中科院软件研究所的李建霖，李勇，付辰以及新加坡科技设计大学的王竟亦，孔屏藩等。与你们的合作令人愉悦，而与你们的探讨使我受益匪浅。

我要感谢我的爱人，她塑造了我现在的许多性格，让我成为更好的自己；我要感谢我的父母，他们坚定地相信我并支持我做出的任何选择，从而让我能够集中主要精

力专注于学术而非各种琐事。

最后，无论社会上对于博士看法如何，这六年是我个人生命中非常开心的时光。虽然也会累，也会烦，论文写不出来时会感到抑郁。可是我有限的人生经验告诉我，任何一次当我以为“明年就可以轻松一点”的时候，生活都无情地否定了我的所有侥幸。因此，无论现实如何困难，今天总比明天要更加闲适。而相应的，在这样的紧张，无定而充斥着黑色幽默的社会面前，身后这六年的生活是何等美好而令人流连。我感谢在这六年中交谈过，合作过，相遇过或擦肩而过的每个人，你们组成了这六年来独一无二的，不可重复的人生体验。

谢谢！

北京大学学位论文原创性声明和使用授权说明

原创性声明

本人郑重声明：所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不含任何其他个人或集体已经发表或撰写过的作品或成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本声明的法律结果由本人承担。

论文作者签名:　　　　　　日期:　　　年　　月　　日

学位论文使用授权说明

（必须装订在提交学校图书馆的印刷本）

本人完全了解北京大学关于收集、保存、使用学位论文的规定，即:

- 按照学校要求提交学位论文的印刷本和电子版本;
- 学校有权保存学位论文的印刷本和电子版，并提供目录检索与阅览服务，在校园网上提供服务;
- 学校可以采用影印、缩印、数字化或其它复制手段保存论文;
- 因某种特殊原因需要延迟发布学位论文电子版，授权学校在 □ 一年 / □ 两年 / □ 三年以后在校园网上全文发布。

（保密论文在解密后遵守此规定）

论文作者签名:　　　　　　导师签名:　　　　　　日期:　　　年　　月　　日