

上海大学出版社

2005年上海大学博士学位论文 100



基于组件的分布式软件 动态配置模型的研究

- 作者：曹 旻
- 专业：控制理论与控制工程
- 导师：吴耿锋 曹建农



G643/138

上海大学出版社

001280750

2005年上海大学博士学位论文 100



基于组件的分布式软件 动态配置模型的研究

- 作者：曹建农
- 专业：控制理论与控制工程
- 导师：吴耿锋 曹建农



贵阳学院图书馆



图书在版编目(CIP)数据

2005 年上海大学博士学位论文. 第 2 辑/博士学位编辑部编. —上海: 上海大学出版社, 2009. 6

ISBN 978-7-81118-367-2

I. 2… II. 博… III. 博士—学位论文—汇编—上海市—2005 IV. G643.8

中国版本图书馆 CIP 数据核字(2008)第 180878 号

2005 年上海大学博士学位论文

——第 2 辑

上海大学出版社出版发行

(上海市上大路 99 号 邮政编码 200444)

(<http://www.shangdapress.com> 发行热线 66135110)

出版人: 姚铁军

*

南京展望文化发展有限公司排版

上海华业装潢印刷厂印刷 各地新华书店经销

开本 890×1240 1/32 印张 274.25 字数 7641 千

2009 年 6 月第 1 版 2009 年 6 月第 1 次印刷

印数: 1~400

ISBN 978-7-81118-367-2/G·490 定价: 980.00 元(49 册)

Shanghai University Doctoral Dissertation (2005)

The Research on the Dynamic Reconfiguration Model of Component-based Distributed Software

Candidate: Cao Min

Major: Control Engineering and Control Theory

Supervisor: Wu Gengfeng Cao Jiannong

Shanghai University Press

• Shanghai •

答辩委员会评语

上海大学

本论文经答辩委员会全体委员审查,确认符合上海大学博士学位论文质量要求。

答辩委员会名单:

主任:	龙晋元	教授,上海交通大学	200030
委员:	顾 宇	教授,复旦大学计算机系	200030
	乐嘉锦	教授,东华大学计算机系	200062
	邵志清	教授,华东理工大学	200052
	刘宗田	教授,上海大学计算机学院	200031
导师:	吴耿锋	教授,上海大学	200072
	曹建农	教授,香港理工大学	200072

(3) 实现了一个在 CORBA 之上的可视化的集成开发环境的原型,该原型实现了 GOFMCBS 模型的各项功能。

论文思路清晰,结构合理,内容翔实,论述清楚,有创新性,表明作者掌握了较宽范围的基础理论和系统深入专门知识,科学研究和实验工作能力强。

曾昊同学在答辩过程中,叙述清楚,回答问题正确。

评阅人名单:

顾 宇	教授, 复旦大学计算机学院	200433
苗夺谦	教授, 同济大学计算机系	200092
李国微	教授, 华中科技大学计算机	430074

评议人名单:

尤晋元	教授, 上海交通大学	200030
周佩玲	教授, 中国科技大学	230026
穆淮扣	教授, 上海大学计算机学院	200072
童维勤	教授, 上海大学计算机学院	200072

答辩委员会对论文的评语

曹旻同学的博士学位论文“基于组件的分布式软件动态配置模型的研究”结合软件动态配置方法和目前流行的对象中间件技术,对基于组件的分布式软件 CBDS 的动态配置模型以及相关问题进行了研究和探讨,选题有理论意义和应用价值。

该论文的主要研究内容和成果包括:

(1) 提出了基于图的动态配置模型 GOPMCBS。该模型可以支持动态配置,还以统一的方法支持了软件的体系结构描述和较高级程序设计,缩短了软件从概念级到设计级的转换过程,达到了 CBDS 从设计到实现的连接。同时,该模型还部分支持容错,简化编程模型,利于软件体系结构的复用。

(2) 分析了组件间的动态依赖关系,提出一种描述组件间动态依赖关系的方法,提高了依赖关系的分析层次。

(3) 实现了一个在 CORBA 之上的可视化的集成开发环境的原型,该原型实现了 GOPMCBS 模型的各项功能。

论文思路清晰,结构合理,内容翔实,论述清楚,有创新性,表明作者掌握了坚实宽广的基础理论和系统深入的专门知识,科学研究和实践工作能力强。

曹旻同学在答辩过程中,叙述清楚,回答问题正确。

答辩委员会表决结果

答辩委员会经无记名投票,以 5 票赞成,0 票反对,一致同意通过曹旻同学的博士学位论文答辩,并一致建议授予其工学博士学位。

答辩委员会主席: **尤晋元**

2005 年 9 月 17 日

摘 要

分布式软件的动态配置一直是软件工程领域中的一个研究热点,而新一代的分布式软件通常由异构的组件组成,组件之间的交互更复杂一些,它们利用对象中间件技术以简化软件开发,但目前的中间件技术对分布式软件动态配置的支持不足,也缺乏很好地描述整个软件体系结构的工具,因此研究基于组件的分布式软件 CBDS (Component-Based Distributed Software) 的动态配置仍然有着重要的意义。

本文重点研究了 CBDS 的动态配置问题,提出了一个面向图形的、基于组件的软件编程模型 GOPMCBS (Graph-Oriented Programming Model of Component-Based Software),用以支持 CBDS 的组成和动态配置。围绕 GOPMCBS 模型,本文还重点研究了 CBDS 的体系结构描述和从设计到编程的无缝连接、组件间动态依赖关系的描述方法以及模型的可视化等问题。实践表明, GOPMCBS 提供了较高层次上的编程模型,使程序员摆脱了中间件底层的实现细节,简化了编程;它有利于描述整个 CBDS 的体系结构,支持 CBDS 的动态配置和软件体系结构的重用,并对整个软件系统的容错提供部分支持。

本文主要的贡献和创新点包括:

(1) 提出了基于图的动态配置模型。针对 CBDS 的动态配置,本文提出了一种新的面向图的模型 GOPMCBS,该模型把

CBDS 以逻辑图的方式来表示和存储,图的顶点表示组件,组件之间的关系用边来表示,边上的标识(label)表示组件之间的多样化的关系,比如关联、继承、调用等。该模型不仅提供了一个抽象的面向图的建模、编程和动态配置框架,而且扮演着软件组件对象之间的通信中间件的重要角色。

(2) 统一了 CBDS 的体系结构描述、动态配置和编程的过程,实现了 CBDS 从设计到编程的无缝连接。GOPMCBS 模型不仅可以支持动态配置,还以统一的方法支持了软件的体系结构描述和较高级程序设计,使动态配置和软件体系结构的描述相一致,并提供语言级的操作原语实现软件,缩短了软件从概念级到设计级的转换过程,真正达到 CBDS 从设计到实现的无缝连接。同时,该模型还部分支持容错,简化编程模型,利于软件体系结构的重用。

(3) 分析了组件间的动态依赖关系,提出一种描述组件间动态依赖关系的方法。使用 GOPMCBS 模型中提供的描述组件间动态依赖的方法,当一个组件被添加入或删除出系统的时候,系统会自动提示应用级程序员该组件跟哪些组件存在依赖关系,应当如何处理这些被影响的组件等等,并自动产生相应的代码。这样就提高了依赖关系的分析层次,使应用级程序员可以更多地关注各个组件之间的逻辑关系,而不是它们内部代码之间的联系。该方法基于图和 XML (Extensible Markup Language,可扩展标记语言)文件的分析,保证了这种依赖关系的分析可以被很容易地扩展到其他编程语言所实现的组件上,因而保持了 GOPMCBS 系统本身所具有的易扩展的特性。

(4) 支持 CORBA (Common Object Request Broker

Architecture, 公共对象请求代理结构) 组件模型 CCM(CORBA Component Model), 缩小了目前 CORBA 编程模型和 CCM 间的差距。CCM 目前尚无正式的商业或开源项目的实现, 它仅仅停留在模型上, 可以说 CCM 的核心规范还没有任何实现。GOPMCBS 模型在 CORBA 之上实现的原型能部分兼容 CCM 模型, 已经具有某些 CCM 欲达到的目标, 因此能支持 CCM, 或者说部分实现了 CCM。

(5) 实现了一个可视化的 IDE(Integrated Development Environment, 集成开发环境)。在 GOPMCBS 中, 我们在图上预定义了一组操作原语, 即给用户提供一个面向图的 API(Application Programming Interface, 应用编程接口), 使用 GOPMCBS API, 程序员可以利用 GOPMCBS 模型的抽象进行 CBDS 的编码工作, 这些编码都是采用面向图的方式进行。本文开发了一个在 CORBA 之上的 GOPMCBS API 的实现库——GOPMCBS LIB, 它实现了 GOPMCBS API 原语。

关键词 基于组件的分布式软件 CBDS, 动态配置, 面向图形的编程, GOPMCBS, 组件依赖

Abstract

Dynamic reconfiguration of distributed software has attracted much attention of research in recent years. Existing work on dynamic reconfiguration can be found in literature. However, how to compose and reconfigure a component-based distributed software (CBDS) remains a big challenge. Generally, distributed software components are homogeneous and components in the CBDS can be heterogeneous. The communication between the CBDS components is more complicated. Although the middleware technologies offer the support to work with heterogeneous and distributed components, they lack facilities to describe the global organization of a CBDS and dynamically reconfigure the CBDS. This paper mainly concerns with the dynamic reconfiguration of the CBDS.

The paper presents a novel, graph-oriented approach, named as the GOPMCBS (Graph-Oriented Programming Model of Component-Based Software), for composing and reconfiguring the CBDS. Based on the GOPMCBS, the paper also probes into the deep related issues, including the description of software architecture for the CBDS, the seamless connection between design and programming of the CBDS, the analysis of dynamic component dependencies and

dependencies description, the visualization of the GOPMCBS, and so on. The prototype implementation and experiments show that the GOPMCBS not only supports the composition and reconfiguration of the CBDS but also supports the powerful graph-oriented programming to make the development of the CBDS simpler and more visual. The GOPMCBS also partly supports the fault tolerance and the reuse of software architecture.

The main contributions of the paper are as follows:

(1) A novel graph-based dynamic reconfiguration model GOPMCBS is proposed. In the GOPMCBS, the software architecture of the CBDS is described as a logical graph, of which nodes represent components and edges denote the relationship between components. Marking an edge with a label shows what kind of relationship is between the components, such as method invocation, inheritance, and relevancies, etc. The GOPMCBS plays several important roles in the development of a CBDS. It not only provides an abstract graph-oriented framework for modeling, programming and dynamically reconfiguring a CBDS, but also acts as the communication middleware among the components of the CBDS.

(2) The proposed model GOPMCBS unifies the whole process of software architecture description, dynamic reconfiguration and programming of the CBDS, thus really implements the seamless connection between the design and programming of the CBDS. Not only does the GOPMCBS

support dynamic reconfiguration of the CBDS, but also provides users with unified graph-based primitives to describe the software architecture of the CBDS and program the CBDS. The dynamic reconfiguration and fault tolerance of the CBDS are also implemented by executing the set of primitives. Therefore, the GOPMCBS tries to narrow the gap between concept-level design and detailed implementation of the CBDS and simplifies the programming of the CBDS. Moreover, software architecture reuse under the GOPMCBS is supported by the mechanism of sub-graph class deriving.

(3) The paper analyzes the dynamic component dependencies and presents an approach to describe the component dependencies of the CBDS. The GOPMCBS provides users with a graph-based and XML-based approach, which shows the interaction and reciprocity between components in a running CBDS, focusing on the CBDS logic instead of the detailed code. The GOPMCBS could dynamically reconfigure the CBDS better through parsing the XML (Extensible Markup Language) files which illustrates the dynamic component dependencies and describes the software architecture. The graph-based and XML-based approach also keeps the expansibility of the GOPMCBS.

(4) The prototype implementation of the GOPMCBS under the environment of the CORBA (Common Object Request Broker Architecture) partly archives what the CCM (CORBA Component Model) proposed, thus narrows the gap between the current CORBA programming model and the

CCM. There is currently a lack of implementations that are based on the CCM. Our prototype implementation of the GOPMCBS is CCM-compatible and archives part of what the CCM proposed, thus supports the CCM. Although the GOPMCBS has not achieved fully what the CCM is targeted at, it narrows the gap between the current CORBA programming model and the CCM. That is to say, not only have we implemented a prototype of the GOPMCBS under the environment of CORBA, but also we are enhancing our approach to support the concepts that are defined in the CCM.

(5) The paper implements a visual integrated development environment for the CBDS. The GOPMCBS provides developers with an intuitive abstraction of the software architecture and a set of predefined graph operations. Using these intuitive abstraction and graph-based primitive, named as the GOPMCBS API, users can write the code for their CBDS that makes use of the GOPMCBS abstractions. We have developed the GOPMCBS LIB, which implements the GOPMCBS API by providing a set of routines for converting and mapping the GOPMCBS communication primitives into the method invocation, and routines for querying and manipulating graphs.

Key words component-based distributed software, dynamic reconfiguration, graph-oriented programming, GOPMCBS, component dependencies

目 录

第一章 绪 论	1
1.1 研究方向和动机	1
1.2 国内外研究动态	2
1.3 本文研究的内容	7
第二章 概念和术语	9
2.1 配置和动态配置	9
2.2 动态配置原则和操作	10
2.3 动态配置过程中关键问题的分析和探讨	11
2.4 本文研究的内容	16
2.5 小结	17
第三章 面向图的基于组件的分布式软件动态配置模型 GOPMCBS	19
3.1 GOPMCBS 模型定义	19
3.2 面向图的接口定义 GOPMCBS API	24
3.3 GOPMCBS 框架	34
3.4 小结	36
第四章 GOPMCBS 模型对基于组件的分布式软件的支持	37
4.1 描述软件体系结构	37
4.2 对动态配置的支持	44
4.3 对重用软件体系结构的支持	48
4.4 对软件容错的支持	53

4.5	简化编程模型	57
4.6	小结	66
第五章 组件依赖及其在 GOPMCBS 中的实现		68
5.1	组件依赖概念及其研究现状	68
5.2	组件依赖在 GOPMCBS 中的描述方法	72
5.3	组件依赖在 GOPMCBS 中的实现	85
5.4	小结	88
第六章 GOPMCBS 在 CORBA 环境下对 CCM 的支持		90
6.1	CCM 概述	92
6.2	GOPMCBS 对 CCM 的支持	95
6.3	GOPMCBS 未实现的 CCM 功能及其改进方案	104
6.4	小结	107
第七章 GOPMCBS 实现原型和性能测试		110
7.1	GOPMCBS 系统设计	110
7.2	GOPMCBS 原型实现	119
7.3	GOPMCBS 原型性能测试	132
7.4	小结	141
第八章 总结与展望		142
8.1	总结	142
8.2	进一步的工作	144
参考文献		146
致 谢		158