一种嵌入式软件构件模型和构件库*)

李 涛 董云卫
(西北工业大学计算机学院 西安 710072)

摘 要 嵌入式系统的快速增长和嵌入式软件复杂度的增长,对嵌入式软件开发技术的提高提出了迫切的要求。软件开发技术正在向半自动化代码生成以及由构件生成系统的方向发展。基于构件的软件开发(CBSD)技术能够显著地减少软件开发的时间和成本。本文首先讨论了一种嵌入式软件构件模型——CMES,该模型定义了嵌入式领域中构件的使用。为了方便构件的管理和查找,本文还设计并实现了一个基于 Web 的嵌入式软件构件库——WRES。WRES 使用刻面分类法提高构件库的浏览和查询效率。

关键词 嵌入式软件,构件,CMES,WRES

A Component Model and Component Repository for Embedded Software

LI Tao DONG Yun-Wei
(Northwestern Polytechnic University, Xi'an 710072)

Abstract With the rapid growth of application demands and the increasing complexity of embedded software, the embedded software development urgently needs some advanced developing techniques. Software technology is shifting toward semi-automated code generation and integration of system from components. It's a good ideal to introduce CBSD into embedded system development, which can significantly reduce the time and cost for software systems development. However, there are some difficulties with the CBSD approach such as component model and component retrieval as well as component composition. In this paper, a Component Model for Embedded Software (CMES) is discussed to specify the components used in embedded domain, and a Web-based Repository for Embedded Software (WRES) to facilitate component management and retrieval. WRES adopts faceted classification scheme to facilitate repository browsing and effective search.

Keywords Embedded software, Component, CBSD, CMES, WRES

1 引言

如今,嵌入式产品的市场非常巨大,预计在今后几年还会快速增长。同时,由于应用的增长,嵌入式软件正变得越来越复杂。嵌入式软件开发迫切需要更加高效的软件重用手段,基于构件的软件开发(CBSD)被证明是最有效的途径。软件构件是一个具有规范接口和确定的上下文依赖的组装单元,它能够被独立部署和被第三方组装[1]。基于构件的软件开发(CBSD)就是将不同的构件组装成软件系统的一种技术。

软件工程在嵌入式系统领域的发展要远远落后于其他应用领域。嵌入式软件是典型的硬件平台相关系统,这些系统具有难于维护、更新、定制和移植等特点。基于构件开发将给嵌入式系统的开发带来巨大的好处,例如:减少开发时间,通过重用已有的构件和继承领域内专家的知识经验,将构件改造并组装成复杂的嵌入式系统软件。

在嵌入式领域中有以下几种基于构件的嵌入式软件开发的方法。文[8]中提出的 Koala 构件模型用于消费电子设备上的嵌入式软件开发。文[9]中提出了一种动态可重配置实时软件开发的框架,该框架基于端口对象(Port Based Objects)。文[10,11]提出了一种名为 PECOS 的构件模型。PECOS 模型旨在将基于构件开发应用于特定类型的嵌入式系统,这类系统被称为领域设备。文[12,13]提出了一种基于 Java 的嵌入式系统开发构件模型 SEESCOA。Koala 和 PBO 模型都缺少对 QoS 的考虑,而 PECOS 和 SEESCOA 模型则只考虑了 1 到 2 个 QoS 属性。

对嵌入式软件开发提供支持的构件库研究也有所进展:文[14]提出了一种在线嵌入式软件库 ORES,它通过一种基于存在论的方法对构件进行检索,方便了对构件的管理。文[15]提出了一种基于刻面分类模式的针对工业控制领域的嵌入式软件构件库。

本文探索将基于构件开发引入嵌入式领域的方法,提出了一种嵌入式构件模型 CMES 和一个基于 Web 的嵌入式软件构件库 WRES。首先介绍介绍了基于构件开发嵌入式软件系统的发展状况;在第 2 节提出了嵌入式软件构件模型,并结合模型介绍了嵌入式软件构件模型的各种属性和接口;第 3 节介绍了一种基于 Web 的嵌入式软件构件库的设计及实现功能;第 4 节结合电子化样机系统的开发,介绍了本文提出的嵌入式构件模型的应用开发情况,最后对本文的工作进行总结。

2 嵌入式构件模型

2.1 通用构件模型的特点

目前在通用计算机应用程序开发中有几种广泛使用的构件模型,几乎都无法适用于嵌入式软件开发。通用的软件构

*)本技术成果受"陕西省自然科学基金"(项目编号:2005F46)和"西安科技攻关计划"(项目编号:GG05022)项目资助。李 涛 硕士研究生,研究方向为网络与分布式计算、嵌入式计算、软件工程;董云卫 博士,副教授,研究方向为软件工程和中间件技术。

件技术包括 CORBA 构件模型(CCM)[2]，JavaBeans[3]，COM[4,5]和.NET[6]。由于 CCM 的面向企业级系统，它会给嵌入式系统开发商带来无法接受的巨大开销。JavaBeans 和.NET 都是虚拟机模型，它们适合更高层的应用软件，但不适合在嵌入式软件开发中占主导地位的底层系统软件的开发，而且.NET 和 COM 都和微软操作系统平台紧密相关。

适用于嵌入式软件系统开发的构件模型必须具备以下特性：

• 轻载性：同其他通用构件模型相比，嵌入式构件模型必须是轻载的。这是由于嵌入式构件目标系统的多样化程度远远超过企业级系统，而且系统的资源非常有限。

• QoS 需求：嵌入式构件模型必须强调 QoS 需求，例如：最坏情况下的执行时间、内存消耗以及可靠性等。这是因为一个工作良好的嵌入式系统不仅依赖于正确的功能实现，而且依赖 QoS 属性。当有若干满足功能需求的构件可供选择时，最符合 QoS 需求的那个将被选择。

针对以上特点本文提出了一个嵌入式构件模型——CMES。以下介绍该模型的关键元素。

2.2 CMES 的关键元素

在 CMES 中，构件是一个可封装和重用的单元。每一个构件都有 2 个关键元素，属性和接口。图 1 和图 2 是 CMES 的示意图。

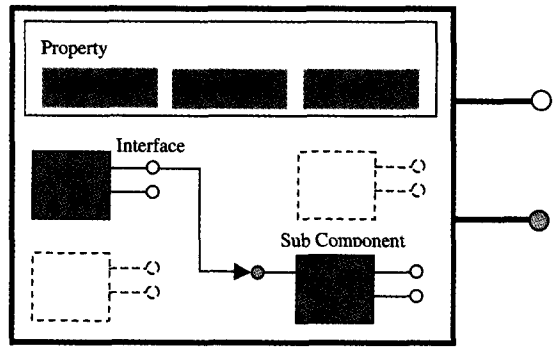

图 1 CMES 结构图

如图 1 所示，属性是构件的内部视图，接口是构件的外部视图。一个构件可能包含多个子构件，子构件之间通过接口进行通信，这种关系用 UML 表示，如图 2 所示。

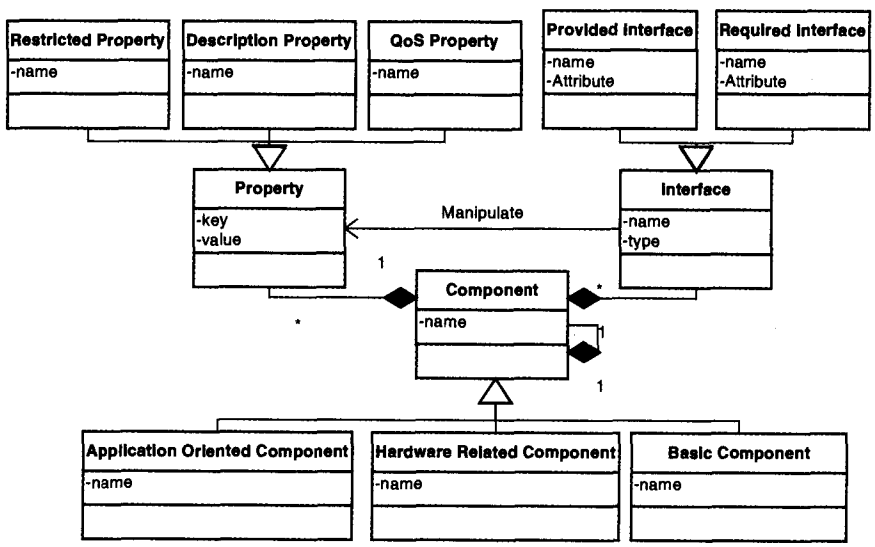

图 2 CMES 的 UML 图

2.2.1 嵌入式软件构件的属性

属性是描述构件某一方面特征的元数据。属性通常表示为一个有值的标识符。根据属性对构件特征描述的类型不同，我们将嵌入式软件构件的属性分为三种：描述属性、约束属性和服务质量(QoS)属性。

• 描述属性：描述属性指的是一般构件都具有的公共属性。常见的描述属性有：

名称 每个构件都有一个名字作为标识。

标识 每个构件都有一个唯一的 ID 号作为与其他构件区别的标志。

类型 每个构件都属于某一个应用类别。

功能 该属性描述构件所能提供的功能。

版本号 每个构件都有它的版本号。

层次 可重用的构件不一定是源代码，它们可以分为 4 个层次：分析件、设计件、代码件和测试件，每个层次都对应软件开发的一个阶段。事实上，软件重用存在于软件开发的每一个阶段。很多能够重用的经验都可以用来指导一个新系统的开发，如：需求规约，设计图，测试计划和测试用例，它们都可以封装到构件中。本文中，我们只关注代码件。

粒度 构件可以分为以下两类：原子构件(也称叶子构件)和复合构件。原子构件不能包含其他构件。复合构件是由若干构件(原子构件或者复合构件)连接组成。构件的连接只能发生在双方的提供接口和需求接口相兼容的情况下。

编程语言 一个构件可以是由 java，C 或者 C++ 等编程语言实现。

表示 一个构件可以是源代码形式，动态链接库形式，静态链接库形式，以及文档，图表等形式。

• 约束属性：该类属性描述构件所依赖的环境。约束属性分为两类：描述绝大多数构件都具有的公有约束属性和描述单独构件的特殊需求的特专有约束属性。公有约束属性包括 CPU 类型、操作系统、依赖的类库等。专有约束属性包括特殊硬件需求、时间约束和内存需求等。

• QoS 属性：QoS 属性是嵌入式软件构件最重要的要素。为了选取合适的构件来开发嵌入式系统，我们必须考虑构件的 QoS 属性，以评估它所提供的性能指标。常见的 QoS 属性

包括平均执行时间、最差执行时间、最优执行时间、内存消耗、功耗、健壮性、可靠性和安全性等。

2.2.2 接口

接口是最低级别的设计和重用单元，它定义了一个构件所能提供的服务。构件通过接口与外界环境进行通信。在CMES中，接口包括一个名称和一组语义相关的操作。根据接口之间的生产—消费关系，我们把接口分为两类：提供接口和需求接口。一个构件必须具有提供接口，而需求接口不是构件所必须具有的。一个接口不能既是提供接口也是需求接口。

接口是构件之间通信的桥梁。两个构件通过一对接口（一个提供接口和一个需求接口）进行通信。

下面以一个电子花样机控制系统的例子详细介绍CMES。

2.3 一个CMES的例子

图3给出了一个电子花样机控制系统中的3个构件。构件MainProcess用于捕捉用户事件。花样机缝纫的过程是：首先，构件MainProcess从构件Pattern中读取并调整花样图案文件；然后，MainProcess发送控制命令到构件Motor执行缝纫操作。

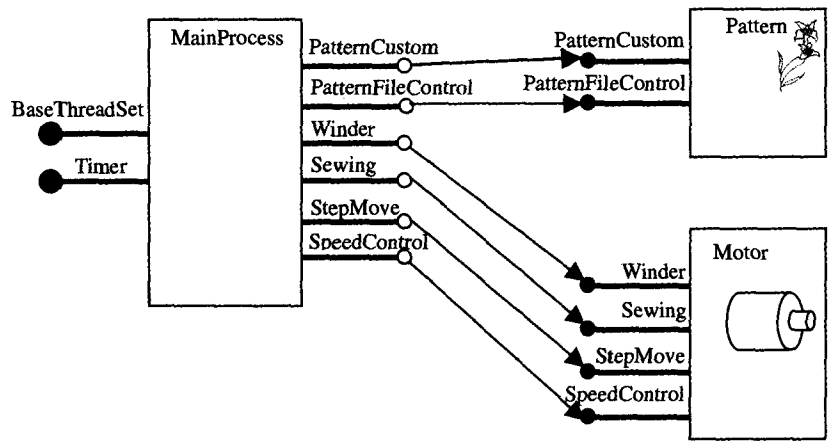

图3 电子花样机控制系统

构件Motor负责控制针的移动速度和方向。它为构件MainProcess提供相关接口，并从MainProcess处捕捉事件。

构件Pattern为外界环境提供对花样文件进行操作的接口。

3 基于Web的嵌入式软件构件库

3.1 构件库的需求

CBSD技术已经广泛用于提高复杂应用程序的开发效率。构件库方便软件开发过程中对构件的重用，以满足降低开发成本和提高生产率的目的[7]。为了将CBSD引入嵌入式开发领域，除了上面讨论的构件模型外，建立嵌入式软件构件库也是必不可少的。

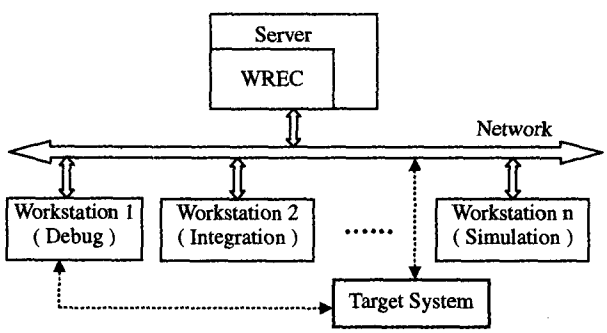

图4 基于网络的嵌入式集成开发环境

目前，Web技术正变得越来越流行，大量的公共库通过Internet以在线的方式向用户提供服务。我们提出了一种基于Web的嵌入式软件构件库WREC（Web-based Repository for Embedded Component），实现对构件的管理和检索。图4描述了WREC在基于网络的嵌入式集成开发环境中的角色，嵌入应用开发者可充分利用集成开发环境的嵌入式软件构件库WREC，WREC根据应用需求（如特定硬件平台需求、时间约束需求等）提取相应的嵌入式软件构件。嵌入应用开发者配置嵌入软构件，并通过嵌入应用框架集成为特定的嵌入式应用系统，附加新开发代码，然后下载到目标机进行在线调试。该开发环境直接面向开发团队，使他们可以通过网络协同进行远程开发调试、项目管理等，大大提高开发效率。

3.2 刻面分类法

当构件库具有了大量的构件后，高效的查找和检索系统是必不可少的。构件库必须有一套对构件进行分类的机制，以便用户可以快速地找到他们想要的构件。

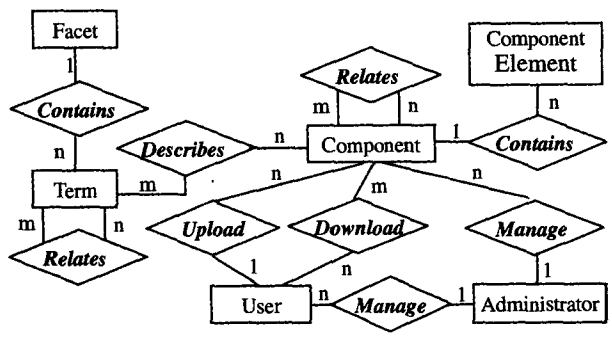

图5 WREC的E-R模型

构件分类法是一个{刻面，术语}对（又叫描述子）的集合[16]。刻面分类模式将关键字置于上下文环境中。更重要的是，刻面分类模式强调按照从不同的观点（或者叫刻面）出发的原则，来进行更为精确的分类。刻面分类法已经被应用于很多不同的工程领域，所以WREC选择它作为主要的分类模式。图5描述了WREC的E-R模型。

WREC中定义了6个刻面，分别描述如下：

1）类型：嵌入式软件构件可以分为以下3种类型：硬件

相关构件(嵌入式内核、设备驱动等)、面向应用构件(与特定应用领域相关)和基础构件(嵌入式中间件等)。

2) 应用领域:嵌入式领域涵盖了很多子领域,为了确定一个构件可能的应用领域,我们必须把它限定在一个特定的子领域。该刻面有以下术语:工业控制、消费电子和移动设备等。

3) 应用环境:该刻面有以下常见术语:操作系统、CPU类型、编译器、数据库和运行时库等。

4) 功能:该刻面描述构件所能提供的服务。常见的术语有:通信、图像处理、音频处理、用户接口、输入输出设备和信息安全等。

5) 层次:该刻面与构件的层次属性相对应,用户可以了解到一个构件在系统开发的哪一个阶段会被用到。

6) 表示:该刻面定义了构件模型 CMES 中的表示属性。

3.3 WREC 的系统结构

图 6 给出了 WREC 基于 Web 的三层结构图。最低层是一个存放可重用嵌入式构件的数据库,我们使用 Oracle 作为数据库管理系统。最上层为用户接口(Web 浏览器),中间层提供一个工具集。工具分为两部分:管理工具和服务工具。管理工具提供给系统管理员,服务工具提供给普通用户。以下分别描述各工具。

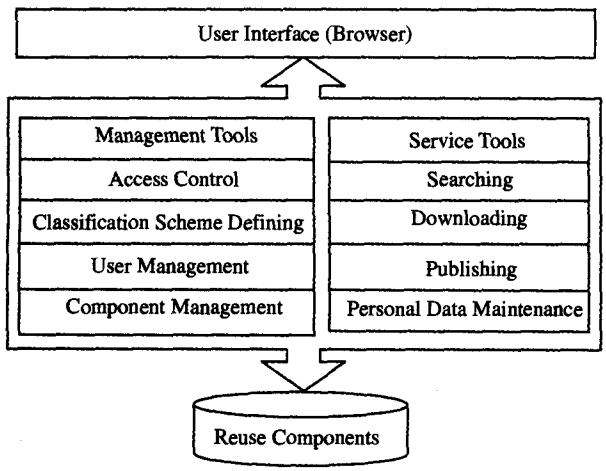

图 6 WREC 的体系结构

查询工具:帮助用户找出合适的构件。系统提供了多种查询方法,如:按照关键字查询和基于刻面分类法的查询。

下载工具:当用户选择了所需要的构件,可以通过下载工具以在线的方式从 WREC 的服务器上将构件下载下来。

发布工具:用户可以在线发布构件,构件的实体被上传到 WREC 的服务器上。为了方便对所发布构件进行分类,用户在发布时必须填写一些记录刻面分类信息的表单。除了构件之外,用户还可以发布对构件的需求信息。这样,用户能够通过发布工具描述他们所需要的构件,并在线征集合适的构件。

个人信息维护工具:用户使用该工具维护个人信息,如:用户名、密码、电话号码、email 地址等。

访问控制工具:系统管理员使用该工具限制用户浏览、查找、删除、添加、发布和下载构件的权限。访问控制工具提供登录控制系其他安全措施,如:一个用户处于非活跃状态的时间超过 30 分钟即视为退出系统。

分类模式定义工具:只有系统管理员才有权定义和修改构件的分类模式。

用户管理工具:系统管理员使用该工具添加、删除用户,以及改变用户的角色。

构件管理工具:系统管理员使用该工具对构件执行添加、删除和编辑操作。

4 应用分析

我们以西安标准缝纫工业集团的智能化电子花样机嵌入式软件系统项目为应用实例,通过对原有软件源代码的可重用性的分析,从中提取并封装了 50 多个有复用价值且可配置性强的嵌入式软件构件。在将基于构件的开发引入该项目后,显著提高了开发效率,缩短了开发周期,节约了开发成本。实践证明我们的嵌入式构件具有如下特点:

1) 应用程序能够在运行时动态加载构件;

2) 用户能够根据应用需求对构件进行定制和配置;

3) 构件版本的升级对用户透明,不影响应用程序的正常运行。

总结 本文提出了一种用于嵌入式开发的嵌入式软件构件模型 CMES,该模型具有以下特性:

1) 强调 QoS 属性。嵌入式软件构件的一个独特之处在于其 QoS 属性。这是嵌入式构件与通用软件构件的最大区别。

2) 将构件的接口分为提供接口和需求接口。接口是构件最基本的复用单元。构件之间的通信实际上是发生在由提供接口和需求接口之间。构件的组合也是发生在这两种类型的接口之间。

此外,为了在嵌入式领域支持基于构件的开发,本文还设计并实现了一个基于 Web 的嵌入式软件构件库系统 WREC。WREC 具有如下优点:

1) 访问简单。WREC 是一个基于 Web 的在线系统,用户可以通过浏览器,从任何地方访问该系统。

2) 强大的刻面分类模式。用户能够使用刻面分类模式,精确、高效地查询到合适的构件。

3) 详细的构件复用信息。系统为用户提供了详细的构件复用信息,如:API 手册、源代码和使用范例等。

下一步的工作方向是开发一个支持 CBSD 的可视化集成开发环境(IDE),使得用户在该环境中能够将可重用的嵌入式软件构件可视化地组装成应用程序,对构件进行组合的代码可以自动生成。

参 考 文 献

1 Szysperski C. Component Software: Beyond Object-Oriented Programming. Addison Wesley, 2002

2 The Object Management Group. CORBA Components. http://www.omg.org/docs/formal/ 02-06-65.pdf, June 2002

3 http://java.sun.com/products/javabeans/

4 Williams S, Kindel C. The component object model: A technical overview. http://msdn.microsoft.com/library/en-us/dncomg/html/msdn_comppr.asp, Oct. 1994

5 Box D. Essential COM. 1st edition. Addison-Wesley Professional. December 1997

6 Microsoft Corporation. NET homepage. http://microsoft.com/net/

7 Luqi, Guo Jiang. Toward Automated Retrieval for a Software Component Repository. In: Proceedings of IEEE International Conference and Workshop on the Engineering of Computer Based Systems (IEEE ECBS), Nashville, USA, March, 1999. 99～105

(下转第 271 页)

$\equiv D^{c}_{S\setminus\{i\},wait_k(TU\{i\}),boundary_k(TU\{i\})}$

若 $i=k$，则

$(\upsilon\vec{ba})(\bar{b_i}\mid\prod_{j\in S\setminus\{i\}}C_j^{wait}\mid\prod_{j\in T}\bar{b_j}\mid b_k\cdots b_n\cdot(C^{abort}\oplus C^{commit})\mid a\cdot C^{abort})\mid\prod_{j\in TU\{1,\cdots k-1,i\}}P_j^{wait}$

$\xrightarrow{\tau}(\upsilon\vec{ba})(\prod_{j\in S\setminus\{i\}}C_j^{wait}\mid\prod_{j\in T}\bar{b_j}\mid b_{k+1}\cdots b_n\cdot(C^{abort}\oplus C^{commit})\mid a\cdot C^{abort})\mid\prod_{j\in TU\{1,\cdots,k\}}P_j^{wait}$

$\xrightarrow{\hat{\tau}}D^{c}_{S\setminus\{i\},wait_k(TU\{i\}),boundary_k(TU\{i\})}$

综上所述，$D^{c}_{S,T,k}\mid P_i^{commit}\approx D^{a}_{S\setminus\{i\},wait_k(TU\{i\}),boundary_k(TU\{i\})}$

命题 3 $D^{c}_{S,T}\mid P_i^{abort}\approx D^{a}_{S\setminus\{i\},TU\{i\}}\mid C_i^{abort}$

证明：

$D^{c}_{S,T}\mid P_i^{abort}$

$\equiv\prod_{j\in S}d_j(x)\mid\prod_{j\in S}C_j^{abort}\mid\prod_{j\in T}\overline{abort_j}\mid\bar{d_i}\langle NO\rangle\mid\overline{abort_i}$

$\xrightarrow{\tau}\prod_{j\in S\setminus\{i\}}d_j(x)\mid\prod_{j\in S}C_j^{abort}\mid\prod_{j\in TU\{i\}}\overline{abort_j}$

$\xrightarrow{\tau}\prod_{j\in S\setminus\{i\}}d_j(x)\mid\prod_{j\in S\setminus\{i\}}C_j^{abort}\mid\prod_{j\in TU\{i\}}\overline{abort_j}\mid C_i^{abort}$

$\equiv D^{a}_{S\setminus\{i\},TU\{i\}}\mid C_i^{abort}$

$\therefore D^{c}_{S,T}\mid P_i^{abort}\approx D^{a}_{S\setminus\{i\},TU\{i\}}\mid C_i^{abort}$

命题 4 $D^{c}_{S,T}\mid P_i^{commit}\approx D^{a}_{S\setminus\{i\},TU\{i\}}$

证明：

$D^{c}_{S,T}\mid P_i^{commit}$

$\equiv\prod_{j\in S}d_j(x)\mid\prod_{j\in S}C_j^{abort}\mid\prod_{j\in T}\overline{abort_j}\mid\bar{d_i}\langle YES\rangle\mid P_i^{wait}$

$\xrightarrow{\tau}\prod_{j\in S\setminus\{i\}}d_j(x)\mid\prod_{j\in S\setminus\{i\}}C_j^{abort}\mid\prod_{j\in TU\{i\}}\overline{abort_j}$

$\equiv D^{a}_{S\setminus\{i\},TU\{i\}}$

$\therefore D^{c}_{S,T}\mid P_i^{commit}\approx D^{a}_{S\setminus\{i\},TU\{i\}}$

当所有参与者都投票之后，即 $S=\Phi$，则

$D^{c}_{\Phi,\Phi,n+1}\equiv(\upsilon\vec{ba})((C^{abort}\oplus C^{commit})\mid a\cdot C^{abort})\mid\prod_{j\in\{1,\cdots,n\}}P_j^{wait}$

$D^{a}_{\Phi,\{1,\cdots,n\}}\equiv\prod_{j\in\{1,\cdots,n\}}\overline{abort_j}$

若协调者在 n 个参与者投票提交之后也决定提交，则

$D^{c}_{\Phi,\Phi,n+1}\xrightarrow{\tau}(\upsilon\vec{b})(C^{commit}\mid a\cdot C^{abort})\mid\prod_{j\in\{1,\cdots,n\}}P_j^{wait}$

$\equiv(\upsilon\vec{ba})(a\cdot C^{abort})\mid C^{commit}\mid\prod_{j\in\{1,\cdots,n\}}P_j^{wait}\equiv C^{commit}\mid\prod_{j\in\{1,\cdots,n\}}P_j^{wait}$

$\underbrace{\xrightarrow{\tau}\cdots\xrightarrow{\tau}}_{n}\prod_{j\in\{1,\cdots,n\}}\overline{commit_j}$

若协调者在 n 个参与者投票提交之后决定撤销，则

$D^{c}_{\Phi,\Phi,n+1}\xrightarrow{\tau}(\upsilon\vec{ba})(C^{abort}\mid a\cdot C^{abort})\mid\prod_{j\in\{1,\cdots,n\}}P_j^{wait}$

$\equiv(\upsilon\vec{ba})(a\cdot C^{abort})\mid C^{abort}\mid\prod_{j\in\{1,\cdots,n\}}P_j^{wait}$

$\equiv C^{abort}\mid\prod_{j\in\{1,\cdots,n\}}P_j^{wait}$

$\underbrace{\xrightarrow{\tau}\cdots\xrightarrow{\tau}}_{n}\prod_{j\in\{1,\cdots,n\}}\overline{abort_j}$

当所有进程的投票都是 Yes 时，$2PCP\approx(\upsilon\vec{c})(\prod_{j\in\{1,\cdots,n\}}\overline{commit_j})\equiv Commit$；当有一个进程投 No 票时，$2PCP\approx(\upsilon\vec{c})(\prod_{j\in\{1,\cdots,n\}}\overline{abort_j}\mid\prod_{i\in A}C_i^{abort})\equiv$ Abort，A 表示投 No 票的参与者的下标集合。因此，$2PCP\simeq$ Abort$\oplus$Commit。

结束语 本文的主要工作是使用 π-演算对两阶段提交协议进行了形式化的描述，并进行了验证。两阶段提交协议是一个被广泛使用的例子，但对它的正确性证明并不多，使用进程代数方法的证明尤为少见。Bernstain 在文[8]中对两阶段提交协议进行了详细的文字描述和讨论，但并没有对其正确性进行证明。Berger 在文[9]中使用了进程代数方法对两阶段提交协议进行了形式化的描述，但其使用的语言并不是标准的 π-演算语言，并且在其描述过程中对两阶段的理解有误，如事务的协调者应该是在收到所有参与者的投票之后自己才会投票，即协调者是没有不确定阶段的。Berger 在文中对两阶段提交协议的正确性也进行了证明，但是他只证明了两阶段提交协议的实现与规约是弱互模拟等价的，并且其证明过程并不严格。

我们对两阶段提交协议的描述参考了 Bernstain 和 Berger 的工作。与他们不同的是：1）我们使用了标准的异步 π-演算进行描述。2）我们给出了详细而准确的描述与证明过程。3）证明了两阶段提交协议的实现和规约是观察同余的。

本文只是无失效情况下对 2PCP 的描述，还没有考虑超时动作和失效恢复情况的描述。下一步的工作是能给出一个包括失效和恢复等情况在内的 2PCP 的描述和验证。

参 考 文 献

1 Clarke E M, Wing J M. Formal Methods: State of the Art and Future Directions. ACM Computing Surveys (CSUR), 1996, 28(4): 626～643

2 Milner R. Communication and Concurrency. Prentice Hall, 1989

3 李舟军. 传值 CCS 和 π-演算的验证理论和算法:[博士论文]. 国防科技大学, 1999

4 Milner R, Parrow J, Walker D. A Calculus of Mobile Processes, Part I, II. Information and Computation, 1992, 100(1): 1～77

5 Boudol G. Asynchrony and the π-Calculus: [Research Report]. 1702. INRIA, Sphia-Antipolis, 1992

6 Honda K, Tokoro M. On Asynchronous Communication Semantics. Object-based concurrent computing, Springer Lect Notes in Comp Sci 612, 1992

7 Amadio R M, Castellani I, Sangiorgi D. On bisimulations for the asynchronous π-calculus. Theoretical Computer Science, 1998

8 Bernstain P A. Concurrency Control and Recovery in Database Systems. Addison-Wesley Longman Publishing Co Inc Boston, MA. 1987

9 Berger M. Towards Abstractions for Distributed Systems: [PhD thesis]. the University of London, Revised Version, 2004

（上接第 262 页）

8 van Ommering R, van der Linden F, Kramer J, et al. The koala component model for consumer electronics software. IEEE Computer, 2000

9 Stewart D B, Volpe R A, Khosla P K. Design of dynamically reconfigurable real-time software using port-based objects. IEEE Trans Software Eng, 1997, 23(12): 759～776

10 Winter M, Genßer T, Christoph A, et al. Components for embedded software-The PECOS approach. In: Proc. Second International Workshop on Composition Languages, 2002

11 Nierstrasz O, Arvalo G, Ducasse S, et al. A component model for field devices. IFIP/ACM Conference on Component Deployment, Berlin, Germany, June 2002

12 Urting D, Van Baelen S, Holvoet T, et al. Embedded software development: components and contracts. In: Proceedings of the IASTED International Conference on Parallel and Distributed Computing and Systems. ACTA Press, 2001. 685～690

13 Urting D, Berbers Y, Van Baelen S, et al. A tool for component based design of embedded software. In: Proceedings of the Fortieth International Confernece on Tools Pacific: Objects for internet, mobile and embedded applications. February 2002, 10

14 Yen I-Ling, Goluguri J, Bastani F, et al. A Component-based Approach for Embedded Software Development. In: Proceedings of the Fifth IEEE International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC. 02)

15 de Lucena V F Jr. Facet-Based Classification Scheme for Industrial Automation Software Components

16 NATO Communications and Information Systems Agency. NATO Standard for Management of A Reusable Software Component Library. 1991, 2: 50～54

17 Yu Zhi-wen, Zhou Xing-she. Study of Development Techniques Based on Embedded Softwae Components. Application research of computers, 2003(4): 12～14