第30卷 第5期 计 算 机 工 程 2004年3月
Vol.30 №5 Computer Engineering March 2004
·基金项目论文· 文章编号：1000—3428(2004)05—0032—03 文献标识码：A 中图分类号：TP311.52

基于C2风格的COM构件组装

沈晨华，焦 桢，阳展飞，钱乐秋
（复旦大学计算机系软件工程实验室，上海200433）

摘 要：随着基于构件的软件工程（CBSE）的快速发展，开发人员已经被分成构件生产者和使用者两个群体。不同的构件标准正在被提出和制定，构件的使用正在变得越来越方便。因此，使用工具来组装构件已经成为可能。文章介绍了C2构架和COM标准的特性，并且针对组装过程中一些亟待解决的问题，提出了观点和想法。

关键词：构件；组装；C2风格；构件对象模型

COM Component Composition in C2-style Architectures

SHEN Chenhua, JIAO Zhen, YANG Zhanfei, QIAN Leqiu
(Lab of Software Engineering, Department of Computer, Fudan University, Shanghai 200433)

【Abstract】 With the development of the component-based software engineering (CBSE), the developers have been divided into two groups, component producer and customer. Various standards of component have been suggested and established. Using components is becoming more and more easy. It is possible to compose components with a tool. This article covers the characters of C2-style architecture and COM, and suggests some ideas for solving the problems occurred while composing components.

【Key words】 Component; Composition; C2-style; Component object model(COM)

1 背景

1.1 C2构架

C2是一种基于分层结构、消息驱动的软件构架风格。图1是一个典型的C2风格构架。C2构架中的基本元素是构件、连接器。每个构件定义有一个顶端接口和一个底端接口，构件通过这两个接口连接到构架中，这使得构架中构件的增加、删除、重组更为简单方便。每个连接器也定义有顶端接口和底端接口，但接口的数量与连接在其上的构件和连接器的数量有关，这也有利于实现在运行时的动态绑定。构件之间不存在直接的通信手段，构架中各元素构件、连接器之间的通信只有通过连接器传递消息来实现。处于低层的构件向高层的构件发出服务请求消息，消息经由连接器送到相应的构件。处理完成后由该构件将结果信息经连接器送到低层相应的构件[1]。

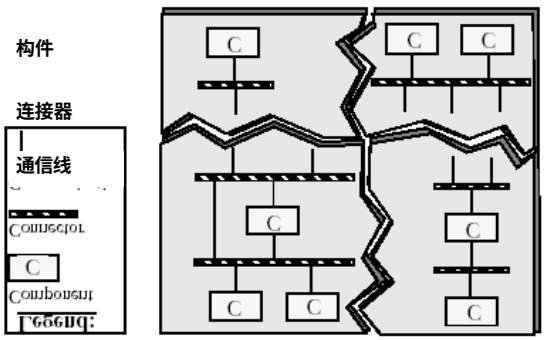

图1 示例C2构架

C2构架风格具有以下优点：

(1) 基于构件的风格：软件构件可以用不同的语言来实现，通过一个构架，它们之间可以更好和更方便地进行交互。

(2) 可伸缩性：构件可以有不同的粒度，还可能运行在分布的、异质的平台上。未必存在共享的内存空间，每个构件可以有自己的控制线程。

(3) 灵活性：系统中可能存在多个用户。有多个对话框以各种形式显示。使用了多种工具包或媒体。构架可能会动态改变。概念构架与实现构架不同，多种实现构架可能体现了同一概念构架。

1.2 COM

构件对象模型（Component Object Model，COM），就是一个可以用于构造软件构件的模型。COM是一个二进制构件标准，允许异构型构件无缝地相互协同工作，允许在没有任何源代码的情况下复用构件。还制定了一系列规范和要求，用以构造软件构件。

对于开发人员来说，COM技术的核心就是接口，COM完美地实现了接口与实现的分离。对于外部使用者来说，一个COM构件的接口分成两类：

(1) 服务接口：定义了构件所能提供的服务，使用者通过接口中的某个方法来获取服务。

(2) 请求接口：虽然并不常见，但是一些COM构件可能会触发一些事件或者需要别的构件（代码）的协助才能完成某项功能。这些功能就被定义在请求接口内，COM构件会调用这些方法而不是实现这些方法。

1.3 基于C2风格的COM构件组装概述

构件组装流程大致如下：

(1) 用户定义构架描述，可能是通过一个可视化界面完成的；

(2) 对构架中所有的构件进行包装，使得包装后的构件之间能够通过消息进行交互；

(3) 产生一些连接器，来负责构件间消息的转发和处理。

2 目标系统

2.1 系统构架

为了更好地复用现有的构件，要尽可能消除构件间的不匹配。这种不匹配性主要体现在功能上的不匹配。即构件提

基金项目：国家“863”计划基金资助项目（2001AA113070；2002AA114010）；上海市科技发展基金资助项目（025115014）
作者简介：沈晨华（1978—），男，硕士生，研究方向为软件工程；焦 桢、阳展飞，硕士生；钱乐秋，教授、博导
收稿日期：2003-02-28 E-mail：huahuashen@hotmail.com

供的功能不能完全满足需要。分成两种情况：(1) 调用一个构件中的多个方法才能实现功能；(2) 多个构件合作才能实现功能。因此，组装时必须考虑这些问题。

在系统中，用户为每一个构件和连接器定义端口。构件和连接器通过端口来发送和接收消息。在这里，一个端口只能对应于一个消息。根据所处的角色不同，把端口分成两类：请求端口和服务端口。

构件与连接器（连接器与连接器）之间的通信线路包含了一组端口映射关系，即处于下方的构件（连接器）的请求端口与处于上方的构件（连接器）的服务端口之间的对应关系。这样，消息才能被明确地传递给正确的接收者。消息的传递可以是本地的，也可以是通过网络的。

接下来，通过一个简单的例子来说明构件间是如何进行交互的，如图2所示。

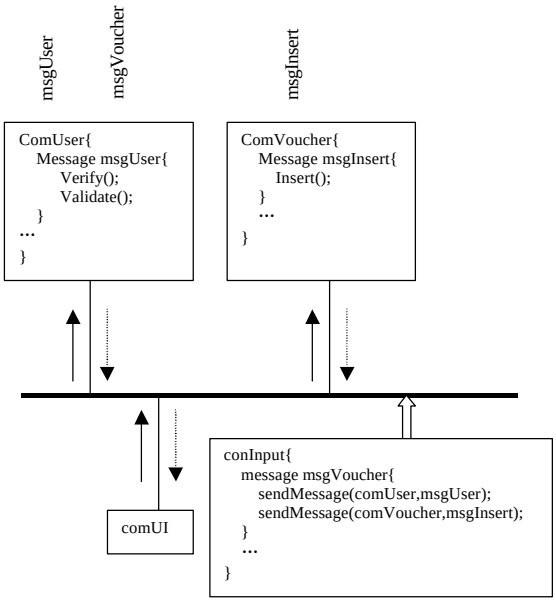

图2 单据录入交互示意图

考虑如下情况，有3个构件： comUI， comUser，comVoucher；还有一个连接器conInput。

当用户希望录入一个单据的时候：

comUI构件发送消息给conInput；

conInput发送消息给comUser；

comUser首先调用verify()，确认用户身份，接着调用validate()，验证用户的权限，然后再发送通知给conInput；

conInput发送消息给comVoucher；

comVoucher调用insert方法，录入一张单据，完成之后，发送通知给conInput；

conInput发送通知给comUI；

comUI输出一些信息，通知用户操作已经顺利完成。

至此，整个交互过程完成。

2.2 对COM构件的包装

正如前面所提到的，目标系统是消息驱动的，而COM构件是通过方法调用来提供服务的，因此需要对COM构件进行适当的包装，使得发送给包装后的构件的消息转化成为对于所包装的COM构件中某个方法的调用。

(1) 在COM中实现事件机制

在COM中通过连接点来实现事件[2]，这就需要一对合作者：源点和汇点。对某一个事件感兴趣的客户（汇点）将自己的意向告诉对象（源点）。源点发出事件给注册过的汇点，如图3。

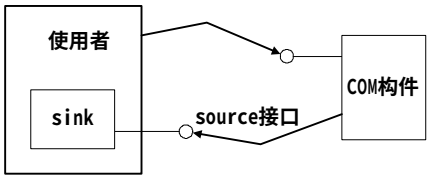

图3 COM的事件机制

(2) 包装

包装的主要想法就是给COM构件加上一个适配器，以复合构件的形式出现在系统中。对于COM构件来说，只有适配器这个唯一的使用者。包装后的复合构件结构如图4。

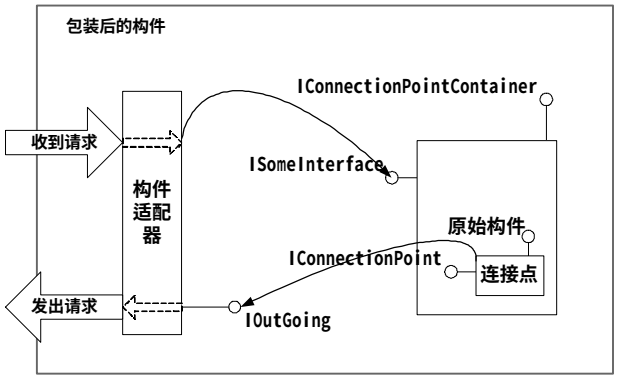

图4 包装后的构件

图4中，适配器中的虚线箭头表示在消息和方法调用之间的转化。

1) 对于外部的请求，适配器会根据预先设定的顺序调用一组构件的方法以及一些中间结果的处理，来提供给外界一个完整的功能。当所有的步骤都被执行完毕后，通过请求所在的端口发出通知，确认服务完成。

2) 对构件将要产生的事件，适配器首先生成一个该事件的汇点并在构件上注册，当事件被触发时，适配器将产生一个相应的消息向外发出，如果消息有返回值，还需要等待。当收到消息的通知后，整个事件的处理才算完成。

为了使得构件能够被正确地包装，需要在构件描述中明确的分类给出COM构件的接口——服务接口和请求接口。这些工作必须由构件生产者来完成。

3 构架描述

为了便于保存和交流，需要有一套符号来对构架进行描述。借助于XML的力量，可以得到一个标准化的、跨平台的和格式严谨的构架描述[3,4]。

(1) 构件

<C2Component>::=<ComponentBasicInfo><InPorts><CompOutPorts>

<InPorts>::=<InPort>*

<CompOutPorts>::=(<OutPort><Event>)*

(2) 连接器

<C2Connector>::=<ConnectorBasicInfo><InPorts><ConnOutPorts>

<ConnOutPorts>::=<OutPort>*

(3) 端口

<InPort>::=<PortName><MsgID><AcitionSequence>

<OutPort>::=<PortName><MsgID>

(4) 消息

<C2Message>::=<MsgID><Parameters>

<Parameters>::=<Paremeter>

<Paremeter>::=[in|out]<ParamType><ParamName>

(5) 连接

<C2Link>::=<TopName><BottomName><PortMaps>

<PortMaps>::=(<OutPortName><InPortName>)+

(6) 构架

<Architecture>: := <C2Components><C2Connectors><C2Messages><Topology>

<C2Components>::=<C2Component>*

<C2Connectors>::=<C2Connector>*

<C2Messages>::=<C2Message>*

<Topology>::=<C2Link>*

对于上述的构架描述语言定义，有以下说明：

(1) 对于构件和连接器的请求端口采取了不同的处理方式：构件中的请求端口，是由于COM构件的事件引起的，而连接器中的则是由于服务端口的调用引起的，因此需要分开处理。而在服务端口上，两者就比较相似，但各自的动作序列相去甚远。

(2) 服务端口中包含动作序列：为了尽可能地复用构件，通过动作序列来解决构件功能上的不匹配性，动作序列中的每个步骤，根据端口所在位置（构件或连接器）的不同，有不同的取值范围。如果在构件中，则可以调用构件提供的方法，以提供服务；如果在连接器中，则可以激活请求端口，以达到消息的路由。当然，动作序列中还可以包含一些辅助动作，如中间变量的处理等。另外，为了提供尽可能强的表达能力，动作序列中还可以使用控制结构（顺序、分支和循环）。

(3) 通过消息传递数据：消息本身没有返回值，如果有数据需要回馈给发送者，可以通过输出参数来实现。

(4) 异步消息：在构件中传播的消息都是以异步方式进行的，发送者发出消息后，立刻继续下面的处理，如果后面需要用到消息的返回值，则等待。当该消息被完成后，处理者就会发送通知给发送者。

(5) 块与块的连接：两个块之间只能有一个连接。在一个连接中,可以实现多个交互，这通过请求端口-服务端口映射对来表示。

(6) 构架约束：必须验证构架描述是否满足一些条件。例如最上层和最下层必须是构件、块与块之间的连接是否遵循C2规范、对象名称的唯一性以及动作序列的合法性等。

4 结论

COM构件是目前使用最广泛的构件标准，如何最大限度地复用现存的构件是一个亟待解决的问题。目前我们已经制定了面向组装的构件和构架描述规范，并开发出了相应的可视化系统，支持对COM构件进行组装。

在本文提出的构架描述规范中，仍保留了相当的表达能力，在一定程度上解决了构件功能的不匹配性。

参考文献

1 Medvidovic N, Oreizy P, Taylor R N. Reuse of Off-the-shelf Components in C2-Style Architectures. Boston, MA: SSR'97, 17-19, 1997-05:17-19

2 Eddon G, Eddon H. Inside Com+ Base Services. MS Press, 1999

3 唐振云，齐克科，钱乐秋. XML技术在CBSE中的应用. 小型微型计算机系统, 2001,(7)

4 赵文耘, 张 志. 基于XML的构架描述语言XBA的研究. 电子学报, 2002,(12A)

※※

（上接第31页）

数目是lgN，因此算法复杂度是$O\left(\log N\cdot 2^{\frac{N}{2}\lg\frac{4N}{N+2}}\right)$，与完全搜索相比算法复杂度减小，但仍然有指数级别的复杂度，实际的应用仍旧有问题。

这里提出一种最大公因数贪心算法，首先将告警表示为所有可以引起这个告警的故障的集合，告警之间的集合作如下的代数运算：

$$(a_1\cap a_2)\cup(a_1\cap a_3)\cup\ldots\cup(a_{k-2}\cap a_k)\cup(a_{k-1}\cap a_k)=S' \quad (4)$$

这个最大公因数过程将得到包含N'个故障的集合S'，$N'\le N$，然后对集合S'执行Katzela的贪心算法可以得出需要的故障集合，如果存在多个故障集合则应用式(3)得出至少有一个实体出现故障的概率最大的故障集合。式(4)所描述的最大公因数算法复杂度可以表示为$O\binom{N}{2}=O\left(\frac{N(N-1)}{2!}\right)$，因此最大公因数贪心算法的复杂度为

$$O\left(\log N'\cdot 2^{\frac{N'}{2}\lg\frac{4N'}{N'+2}}+\frac{n(n-1)}{2!}\right) \quad (5)$$

这个算法的复杂度取决于告警集合A中所有元素告警域重叠度的情况，如果系统中的故障彼此相关性较小，那么它们的告警域将重叠元素必然很少，式(4)所得集合的$N'\ll N$，式(5)的第1项就可以忽略不计，化简为$O(n^2)$，算法不再具有指数复杂性。

4 结论

时间关联和聚类关联是告警关联的两个最重要的部分，综合的告警关联系统必须侧重这两个方面。考虑网络事件发生的固有的模糊性，可以利用模糊逻辑进行告警的时间关联，根据给出的一个新的告警关联窗口选择算法，避免了故障关联过程中将一个故障的告警集合割裂在两个关联窗口中。聚类关联采用依赖关系图模型，根据对其他算法的分析提出了针对此模型的一种最大公因数贪心算法，用于解决故障的告警域关联度较小问题。未来的工作主要包括：考虑分布环境下告警关联系统的设计；将神经网络技术引入告警关联中; 研究选择故障的告警域的方法, 令其重叠度尽可能小。

参考文献

1 Becraft W R, Lee P L. An Integrated Neural Network/Expert System Approach for Fault Diagnosis [J]. Computers Chem. Engng, 1993,17 (10): 1001-1014

2 Jakobson G, Weissman M. Real-time Telecommunication Network Management: Extending Event Correlation with Temporal Constraints [A]. Fourth International Symposium on Integrated Network Management[C]. Santa Barbara, CA, 1995: 290-302

3 Aboelela E, Douligeris C. Fuzzy Temporal Reasoning Model for Event Correlation in Network Management [A]. 24th Conference on Local Computer Networks, LCN'99[C], Lowell, Massachusetts, USA, 1999: 150-159

4 Chao C, Yang D, Liu A. A Time-aware Fault Diagnosis System in LAN [A]. Integrated Network Management Proceedings[C], IEEE/IFIP International Symposium, 2001: 499-512

5 Katzela I, Schwartz M. Schemes for Fault Identification in Communication Networks [J]. Networking, IEEE/ACM Transactions, 1995,3 (6): 753-764

6 Chi-Chun Lo, Shing-Hong Chen, Bon-Yeh Lin. Coding-based Schemes for Fault Identification in Communication Networks [A]. Military Communications Conference Proceedings[C], IEEE, 1999,2: 915-919