

基于层次消息总线的软件构架动态模拟和演化研究^{*}

Study on Dynamic Simulation and Evolution of Hierarchical Message Bus Based Software Architecture

覃国蓉

张世琨

(北京大学计算机科学技术系 北京100871) (广东惠州大学计算中心 惠州 516015)

Abstract Software architecture has become one of the research hot spots in the field of software engineering. In [1], a hierarchical message bus based software architecture style has been proposed. With this style, components communicate with each other through message buses, and complicate components can be resolved and refined step by step. The whole system and individual components have unified structure. This style can be used to describe distributed systems based on message communication. On the basis of the above software architecture style, this paper studies the dynamic simulation and evolution of the systems corresponding with this style, and a tool named Jade Bird Software Architecture Simulating Execution Tool is also developed to support the system simulation and evolution.

Keywords Software architecture, Message bus, Dynamic simulation, Dynamic evolution

1. 引言

在软件工程实践中,人们越来越深刻地认识到,随着软件系统规模和复杂性的增加,系统总体结构设计的重要性已远远超过特定算法和数据结构的选择,良好的软件构架是保证系统成功的关键,软件构架已成为目前软件工程研究的热点之一。

软件系统的构架定义了组成系统的构件和构件之间相互作用的关系^[1],表达构件之间相互作用关系的又称为连接件(connector)。在构架层次上,典型的构件包括客户、服务器、数据库、过滤器等,构件之间的交互可以是非常简单的,如过程调用和共享数据访问;也可以是复杂并具有丰富语义的,如客户-服务器协议、数据库访问协议、异步事件广播和管道等。软件构架规定了系统的基本结构和行为模式,是系统构造和演化的基础。良好的构架可以为软件开发和维护带来好处。

本文在文[1]提出的基于层次消息总线的软件构架风格的基础上,研究了符合这种软件构架风格的系统的动态模拟运行和演化,并设计了相应的青鸟软件构架动态模拟运行工具。

2. 基于层次消息总线的软件构架

文[1]提出了基于层次消息总线的软件构架风格,如图1所示。消息总线是系统的连接件,负责消息的传递、过滤和分派以及处理结果的返回;各个构件挂载在

消息总线上,向总线登记感兴趣的消息类型;构件根据需要发出消息,由消息总线负责把该消息分派到系统中所有对此消息感兴趣的构件,消息是构件之间通讯的唯一方式;构件接收到消息后,根据自身状态对消息进行响应,并通过总线返回处理结果。由于构件通过总线进行连接,并不要求各个构件具有相同的地址空间或局限在一台机器上。系统具有并发和分布的特点,可以较好地刻画分布式并发系统,以及基于 CORBA、COM 和 EJB 规范的系统。

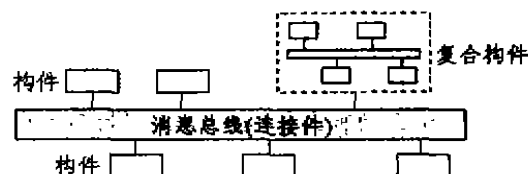


图1 基于层次消息总线的软件构架

系统中的复杂构件可以分解为比较简单的子构件,这些子构件通过局部消息总线进行连接,这种复杂的构件称为复合构件。如果子构件仍然比较复杂,可以进一步分解。如此分解下去,整个系统形成了树状的拓扑结构,树结构的末端结点称为叶结点,它们是系统的原子构件,不再包含子构件。另外,整个系统也可以作为一个构件,通过更高层的消息总线,集成到更大的系统中,于是,整个系统和组成系统的单个构件就可以采用统一的方式进行刻画。

^{*} 本文得到国家“九五”重点科技攻关项目基金(No. 98-780-01)资助。

3. 基于层次消息总线的软件构架的动态模拟

对于传统的软件开发,尽快开发出原型系统,并通过原型系统的运行得到用户的反馈,进一步明确系统的需求,改进系统的设计,对提高软件质量,保证软件系统的开发成功具有重要意义。软件构架设计位于系统开发的前期阶段,是后续开发的基础,如果在软件构架设计阶段能够对未来的系统进行动态模拟,就可以对软件构架的设计进行验证,并完善软件构架。

在基于层次消息总线的软件构架中,所有构件是通过层次消息总线连接的,只要模拟出各个构件收发来的消息,构件对消息的响应,以及消息在层次消息总线上的分派情况,就可以完成该软件构架的模拟运行。基于层次消息总线的软件构架的动态模拟流程如图2所示。

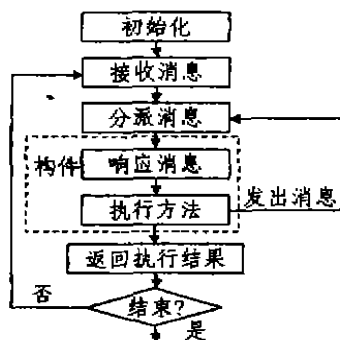


图2 动态模拟流程

1. **初始化** 把软件构架中所包含的各个构件的状态置为初始状态,并完成构件的登记工作,把相关信息填入构件-消息响应登记表和消息过滤表中。

2. **分派消息** 消息总线根据构件-消息响应登记表把该消息分派到系统中对该消息感兴趣的构件,如图3所示。

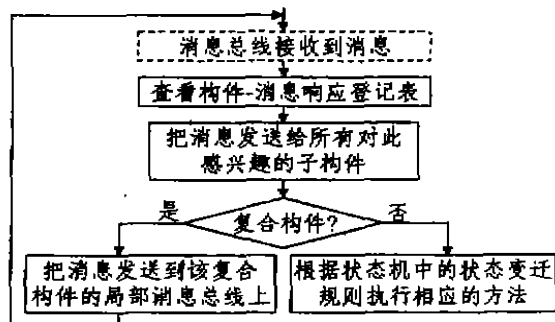


图3 消息分派流程

3. **响应消息和执行方法** 构件在接收到消息后,

根据当前所处的状态,对消息进行响应,激活相应的方法,随消息发送的参数被传给方法的实参,把这些参数压入运行栈,方法的运行环境包括构件局部变量和方法实参。逐条执行方法中的语句。在方法的执行过程中,可能需要系统中其它构件的协作,因此发出相应的消息,该消息首先在构件直接挂接的消息总线中进行广播,如果有同层构件进行响应,则执行相应的方法;否则,该消息则向上传递,在上--层的总线中进行广播,直到在某一层有构件对其进行响应,或者直到最上一层,于是该消息被发送到系统的外部,外部对该消息进行响应,处理结果沿着相反的路径传递给发出请求的构件。方法执行结束后,相应的构件状态发生变迁,并把结果数据返回消息发出者(因此在消息的传递过程中,消息发出者的标识是作为消息的隐式参数进行传递的,用于确定结果返回的目标。另外,将来可以做进一步的扩充,例如用做客户的身份验证),消息发出者的方法从调用断点处继续执行。

4. 基于层次消息总线的软件构架的动态演化

软件构架的动态演化是指在系统运行时刻进行功能上的修改、扩展和剪裁,其困难在于,系统如何适应修改所带来的变化,其中涉及了如何维护构件之间的关系,功能的分派,以及这一修改对整个软件构架产生的其它影响等。在基于层次消息总线的软件构架中,软件构架的动态演化体现为改变构件的实现、增加删除构件,增加删除消息响应登记和改变消息过滤规则。

1. 改变构件的实现

如果不改变构件的接口,只改变其实现,在这种情况下,由于与该构件相关的其它构件只关心其接口,并不关心构件是如何实现的,因此可以通过直接替换该构件实现,就可以完成动态演化。

2. 改变消息过滤

由于消息过滤的改变,并不影响系统中的构件及其连接,因此可通过在相应的消息总线上改变消息过滤规则,实现动态演化。

3. 增加删除构件和改变消息响应登记

增加删除构件和改变消息响应登记涉及到构件接口的改变,甚至有可能影响到该构件的所有父构件的接口。例如,假设复合构件C接收和发送的消息集合分别是 $\{R_1, R_2\}$ 和 $\{S_1, S_2\}$,它被两个子构件C₁和C₂所细化,C₁接收和发送的消息集合分别是 $\{R_1, M\}$ 和 $\{S_1\}$,C₂接收和发送的消息集合分别是 $\{R_2\}$ 和 $\{S_2, M\}$ 。现在需要在构件C的局部总线上增加一个构件C₃,它接收和发送的消息集合分别是 $\{R_3\}$ 和 $\{S_3\}$,但构件C₁和C₂都不会对S₃消息响应,因此必须在构件C的发送消息集合中增加S₃。

增加删除构件就是增加和删除相应(父)构件发送和接收的消息集合,因此可以归纳为四种不同的组合方式,即增加发送消息类型、删除发送消息类型、增加接收消息类型和删除接收消息类型,处理流程如下:

·增加发送消息类型 如果有同层的其它构件发送或接收同样的消息,处理流程结束;否则向上传递,在父构件的接口增加相应的发送消息类型,并考察父构件的同层构件是否发送或接收同样的消息。逐层向上传递,直到有某一层的构件发送或接收相应的消息,或到了最高层,这时应增加对外发送的消息类型。

·删除发送消息类型 如果有同层的其它构件发送或接收同样的消息,处理流程结束;否则向上传递,在父构件的接口删除相应的发送消息类型,并考察父构件的同层构件是否发送或接收同样的消息。逐层向上传递,直到有某一层的构件发送或接收相应的消息,或到了最高层,这时应减少对外发送的消息类型。

·增加接收消息类型 如果有同层的其它构件接收同样的消息,处理流程结束;否则向上传递,在父构件的接口增加相应的接收消息类型,并考察父构件的同层构件是否也接收同样的消息。逐层向上传递,直到有某一层的构件也接收同样的消息,或到了最高层,这时应增加相应的接收消息类型。

·删除接收消息类型 如果有同层的其它构件接收同样的消息,处理流程结束;否则向上传递,在父构件的接口删除相应的接收消息类型,并考察父构件的同层构件是否也接收同样的消息。逐层向上传递,直到有某一层的构件也接收同样的消息,或到了最高层,这时应删除相应的接收消息类型。

5. 青鸟软件构架动态模拟运行工具

在前面研究的基础上,我们开发了一个软件构架动态模拟运行工具——青鸟软件构架动态模拟运行工具^[2],支持系统的动态模拟运行和动态演化。为此,我们设计了青鸟软件构架命令控制语言(Jade Bird Software Architecture Command Control Language,简称JB/SACCL)。用户通过JB/SACCL命令同动态模拟运行工具进行交互,工具负责JB/SACCL命令的解释、执行,完成系统的动态模拟和动态演化。

图4给出了该工具的整体结构图。图形用户界面负责接收用户的命令,并显示执行的结果。用户接收到start命令后,进行初始化。然后,根据接收的命令类

型,分别由动态模拟运行和动态演化模块处理,再依次交由相应的下层模块处理。

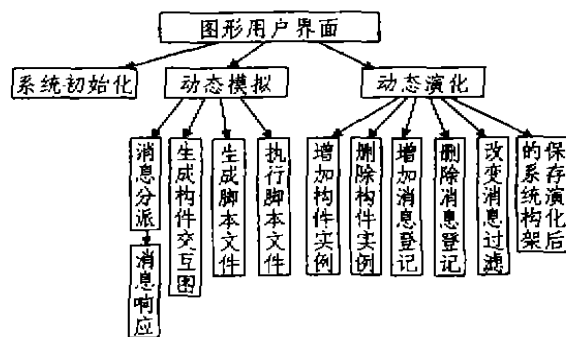


图4 青鸟软件构架动态模拟运行工具的总体结构

结束语 软件构架表达了系统的高层设计,为构件获取提供指导,为构件的集成组装提供基础和上下文。基于层次消息总线的软件构架风格要求构件有标准的接口,并提供标准的集成组装机制,从而适应了软件系统集成组装的要求。本文研究了符合这种软件构架风格的系统动态模拟和动态演化,并开发了青鸟软件构架动态模拟运行工具,使得在系统开发早期的软件构架就可以作为某种意义上的原型系统,通过动态模拟运行供用户评价和反馈,有助于得到改进软件构架的设计。

进一步的工作包括:①目前的控制流程建立在消息同步响应的基础上,以后将扩展到消息异步响应的情况;②目前只考虑点对点通讯,即每条消息只有一个构件响应的情况,以后还要考虑多点广播的情况,一条消息可能有多个响应者。

参考文献

- 1 张世琨. 基于层次消息总线的软件构架研究:[北京大学博士学位论文]. 2000. 6
- 2 覃国蓉. 软件构架动态模拟运行工具的设计与实现:[北京大学硕士学位论文]. 2000. 6
- 3 Shaw M, Garlan D. Software Architecture: Perspectives on an Emerging Discipline. Prentice Hall, 1996
- 4 Bass L, Clements P, Kazman R. Software Architecture in Practice. Addison-Wesley Publishing Company, 1997
- 5 Taylor R, et al. A Component- and Message-Based Architectural Style for GUI Software. IEEE Transactions on Software Engineering, 1996(June)