文章编号:1001 - 9081(2007)04 - 0860 - 03

基于 FCD 扩展的多构件选择过程

盛津芳,王 斌
(中南大学 信息科学与工程学院,湖南 长沙 410083)
(jfsheng@ mail. csu. edu. cn)

摘 要:大型软件系统中的构件间存在依赖关系,因此难以对单个构件独立进行评估。现有的方法大多针对单个构件,并不适合多个构件的评估和选择。对一种成熟的系统分解方法 FCD 进行扩展,提出了一种针对多构件进行评估和选择的方法。在系统分解的过程中识别出局部需求和横切需求。局部需求被分解到各个模块中作为对候选构件进行局部评估的准则,横切需求则进入全局评估。系统分解的过程中,需求吸取识别出来的现有构件的特征,从而进一步精化,同时考虑到不同粒度的构件的组合。全局选择被定义成为一个在给定约束条件下选择出一组具有最大的全局需求满足度的最优构件组合的非线性优化问题。

关键词:基于构件的软件开发;构件选择过程;软件需求分解
中图分类号: TP311 **文献标识码**:A

Multiple COTS selection process based on extension to FCD

SHENG Jin-fang, WANG Bin
(*College of Information Science and Engineering, Central South University, Changsha Hunan* 410083, *China*)

Abstract: In large software system, components depend on each other, which leads to the difficult evaluation of individual component. The present methods for single COTS selection do not fit multiple COTS selection very well. Then a multiple COTS selection process was proposed based on extension of a proven system decomposition technique named FCD. During the process of decomposition, local requirements and crosscutting requirements were identified. Local requirements were allocated into modules as local evaluation criteria for candidates COTS while crosscutting requirements were considered in global evaluation. The process supports requirements adaptation to capabilities inherent in COTS products and takes into consideration varying granularity of COTS as well. The global selection was described as a nonlinearly constrained optimization problem with the purpose of determining an optimal combination of COTS products with maximal global fitness under certain constraints.

Key words: COTS-based software development; COTS selection process; requirements decomposition

0 引言

随着软件系统复杂度的不断增加,软件开发人员需要新的方法和技术来支持软件开发过程。传统的软件开发方法是在给定的软件需求基础上进行实现,而不是选择合适的构件进行组装[1]。基于商业构件(COTS)的软件开发方法(CBSD)就是通过对现有构件的组装构建系统。这里的商业构件是指由第三方开发、不提供源代码、仅通过接口与其他软件进行集成,并能实现一定功能的软件产品。随着构件在大型系统构造中越来越广泛地应用,软件开发的早期阶段迫切需要相应的方法和模型来支持构件选择过程。

软件系统通常由多个构件组成,构件间彼此交互与协作以实现系统特定的功能与非功能需求。在 CBSD 中,将复杂系统分解成子系统是识别候选构件的必要先行步骤。分解之后,构件的评估和选择首先针对各个子系统进行,从而缩小了构件的搜索空间[2]。

传统的面向对象方法使用包、组件和子系统等结构描述对系统的分解结果,但对于如何进行分解却没有提供足够的指导原则。文献[3]提出了一种结合结构化分析与面向对象思想的系统分解方法(Function-Class Decomposition,FCD),它在识别出类的同时将系统分解成为具有层次结构的子系统。通过对 FCD 进行扩展可以为 CBSD 中的构件识别和组装提供支持。

本文提出了一种基于 FCD 扩展的多构件选择方法 MOTS(Multiple cOTs Selection)。

1 FCD 方法概述

尽管面向对象方法很适合开发小型的软件系统,软件工程师们却逐渐意识到大型而复杂的系统需要结构。FCD 方法结合了传统的基于功能需求的系统分解方法和面向对象的系统分解思想,前者将一个大的系统分解成为子系统,后者则用于对子系统中的行为进行识别和建模。

FCD 是一个迭代的过程,在对系统自上而下进行分解的同时,识别出构成系统的类并将其分组,每一个组被称为一个功能模块。聚集到一个模块中的类表现出高内聚和低耦合的性质。分解之前,整个系统被视为一个大的功能模块,一组代

收稿日期:2006 - 10 - 16;**修订日期**:2007 - 01 - 12
基金项目:湖南省自然科学基金资助项目(05JJ40132);中南大学博士后科学基金资助项目
作者简介:盛津芳(1971 -),女,湖南长沙人,讲师,博士研究生,主要研究方向:基于构件的软件工程; 王斌(1973 -),男,山西大同人,副教授,博士,主要研究方向:安全软件、面向方面的软件开发.

表该模块行为特性的类被识别出来。通过对场景与类之间的交互进行动态分析,类被划分到不同的分组中以取得较高的内聚度和较低的外部耦合度。每个分组构成下一层中的一个功能子模块。类聚集成为功能子模块的同时,系统的功能需求也被分解,能够由某个子模块完全实现的功能被分解到相应的子模块中。那些“横切”多个子模块的功能需求仍然保留在父模块中。分解所得的所有子模块构成分解结构中的一个新的层次。对每个功能子模块,需求进一步精化以识别出更多的类,进而分解成为粒度更小的模块。直到每个功能模块中都没有新识别出来的类,分解停止。图 1 是利用 FCD 方法将一个基于 Internet 的实时会议系统 M-Net 分解成为功能模块的结构图[3]。

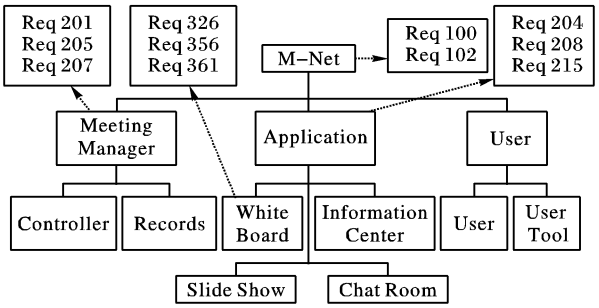


图 1　M-Net 功能模块层次结构

2　基于 FCD 扩展的多构件选择

一个良好定义的构件选择过程是 CBSD 的基础[4]。现有的由功能需求驱动的构件选择方法大多针对单个构件[5~8],并不适合多个构件的评估和选择。多构件选择需要确定的不是单个构件,而是覆盖不同功能子集的一组构件集合。要识别和评估组成系统的多个构件,首先要对系统进行分解。

FCD 方法可以为多构件选择过程提供支持,主要理由为:

1)通过需求分解降低复杂度。系统的需求从用户角度描述了对软件系统功能和非功能的要求。系统级的需求分解到松散耦合的子系统中,才能支持后续的同时进行的多个构件的识别和评估。FCD 方法在对系统进行分解的同时,也实现了需求的分解。

2)需求精化的过程中融入构件的特性。与传统的基于自主开发的软件过程不同,CBSD 中需求的获取要面向可能的构件解决方案。在系统开发的早期阶段只能给出一个粗略的需求[9,10]。随着更多的构件产品被识别出来,部分构件的特性被采纳作为需求。FCD 包含的需求精化的步骤为 CBSD 逐步获取需求提供了有效地支持。

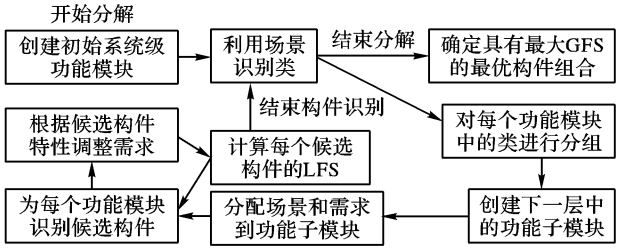
4)考虑不同粒度的构件。构件产品具有不同的粒度,大到一个数据库管理系统 DBMS,小到一个图形界面元素[11]。一组给定的需求可能由单个构件来实现,也可能由多个构件的组合来实现。FCD 将系统分解为层次结构自然地支持识别各种粒度的构件。

5)从全局角度评估构件对需求的匹配度。FCD 方法将系统需求分为局部和全局需求两大类。部分功能需求和大部分非功能需求都具有全局特性。局部需求仅与单个功能模块相关联,而涉及到多个功能子模块的交互与协作的全局需求则驻留在父模块中。局部需求和全局需求为构件选择过程中的局部评估和全局评估提供了评估准则。

3　多构件选择方法 MOTS

3.1　MOTS 概述

MOTS 方法包含四个相互关联的过程,即需求分解、构件识别、构件局部评估和全局构件评估与选择。



说明:LFS指局部需求匹配度;GFS指全局需求匹配度

图 2　基于 FCD 扩展的多构件选择过程

MOTS 的主要步骤如图 2 所示。MOTS 是一个迭代的过程,应用 FCD 方法将系统逐步分解成为具有层次结构的功能模块,每个模块对应一组特定的功能需求,在分解的过程中对每个模块识别可能的构件解决方案。首先整个系统被视为一个系统级功能模块,并与一组已知的粗粒度需求和相应的场景描述相关联。根据面向对象的系统分解原则,一组具有数据和行为特性的类被识别出来。通过场景映射[12]将类分组,构成若干功能子模块。子模块集合构成分解层次中的下一层,同时需求和场景描述被分解到相应的子模块中。

用户需求以及系统的设计需求作为构件识别和评估的准则。对每个子模块,首先检索构件库以识别可能的候选构件。候选构件可能具有一些在初始需求中不存在的特性,同样也有某些需求不被任何构件满足。评估专家需要在构件特性与需求之间进行权衡,以确定哪些构件特性被采纳作为必须被满足的需求,哪些需求被弱化成为可选需求。随着更多的构件被逐步识别出来,该“权衡”过程重复进行。评估专家对选定的候选构件计算其局部需求匹配度 LFS(Local Fitness Score)。LFS 值越高意味着构件具有越高的对局部需求的满足度,LFS 值低于预定义阈值的构件被淘汰。经过局部评估后,每个子模块对应一个候选构件队列。该队列为空,则表示没有发现合适的构件解决方案。

子模块中,不能或不必继续分解的模块称为原子模块,否则称为复合模块。FCD 中,没有新类识别出来的模块不能继续分解,该条件同样被作为 MOTS 中模块分解判停的条件。该原子模块成为分解树的一个叶节点,若对应的候选构件队列为空,则该模块只能自主开发。另一个模块分解判停条件是对该模块已找到合适的构件,不必再进一步分解。

对需要进一步分解的复合模块,首先根据精化后的需求识别出更多的类,然后对类分组,形成若干子模块。对所有子模块,重复进行上述步骤,直到所有模块都是原子模块,分解停止。分解树中所有候选构件队列不为空的叶节点对应的所有候选构件进入全局评估和选择步骤。该步骤的目标是确定一个具有最大的全局需求匹配度 GFS(Global Fitness Score)的最优构件组合。

一种特殊情况是对代表系统功能的顶级模块,在构件库中找到了合适的构件产品。这是一个单构件评估问题,MOTS 中的局部评估方法同样适用于对单个构件进行评估。

3.2　需求分解

降低复杂度是开发大型软件系统的一个主要问题[13]。通过功能分解对系统实行“分而治之”可以有效降低复杂度。本文提出的方法中假定任何大型复杂的软件系统都由一组满足特定用户需求的子系统构成。

MOTS 是由需求驱动的构件选择方法。部分需求与功能相关,另外一些则是非功能性的,例如人机交互界面、系统性

能和设计约束等。部分功能需求和大部分非功能需求不能被划分到单个模块中,本文将这种全局需求称为横切需求。对构件进行全局评估时才考虑构件组合对横切需求的匹配度。

将一个模块分解成为若干子模块,是通过对模块中已识别出来的类进行聚集而成的。聚集的原则是度量子模块的内聚度和子模块间的耦合度。文献[12]中提出了一种场景映射法,通过场景路径来反映场景与模块的交互。当一个模块内部的所有场景都表现出较高的功能相关性时,该模块具有较高的内聚度,反之则具有较低的内聚度。场景路径也用于度量模块间的耦合度,当一条场景路径频繁穿越两个模块之间时,这两个模块具有较高耦合度,反之亦然。

模块分解的同时,需求也被分解。当一个场景所表示的行为被封装于单个子模块内时,该场景与相关的需求描述被划分到该子模块中。如果场景所表示的行为涉及到多个子模块,则仍然被保留在父模块中。

3.3 构件识别与局部评估

MOTS 对系统进行分解的同时从构件库中识别可能的候选构件。搜索所需的构件产品以及如何描述构件的特征是获取构件的一个主要问题。来自于构件产品制造者的构件描述信息往往不够充分,同时缺乏统一的分类方式,为识别构件带来了更大的困难。目前,构件的刻面描述是一种正逐步得到重视与应用的描述方法。例如,REBOOT、NATO 和青鸟构件库中的构件都是基于刻面分类的。文献[11]中提出了一种构件的刻面分类方案,例如构件的类型是其中的一个主刻面,该刻面对应的术语空间包括“操作系统”、“数据库管理系统”、“中间件”等术语。

查询构件时,查询条件除了涉及到功能需求外,还可能涉及系统设计需求。某些功能需求为必要的,即要求构件必须满足的。除了这类需求外,某些设计需求在构件识别阶段也可以作为筛选构件的必要条件。例如,构件的实现模型就是设计需求的一种。目前,主流的构件实现模型有 EJB、COM、CCM、Web Service 和 JBCL 等。构件的实现模型是对构件规约和组装技术的描述,可作为对构件的非功能性特征,如互操作性、可移植性、可重用性和可维护性进行评价的基础。此外查询条件中还可能包含对构件产品和提供商的要求,例如要求构件的质量已经过第三方认证。

对构件局部评估时,常采用的决策分析方法有多属性效用理论、加权和法和层次分析法(AHP)等。文献[14]讨论了这些方法的用途和局限性。作者在文献[15]中提出了一种基于相似度的局部构件评估方法,评估时既要考虑需求中定义的功能,也要考虑构件中已实现的但系统并不需要的功能。局部需求匹配度 LFS 反映了构件功能与需求的“相似”程度。LFS 值低于预定义阈值的构件被淘汰,其他的则被保留进入全局评估。

3.4 全局评估与选择

系统分解与局部评估结束后,得到一个功能模块的树型结构,部分叶节点对应一组根据 LFS 值排序的候选构件。候选构件队列为空的叶节点表明构件库中不存在与该模块足够匹配的构件产品,只能采用内部开发。全局评估与选择过程如图 3 所示,作为局部评估结果的 LFS,以及在构件识别过程中得到的构件产品的相关信息,例如构件的购买价格、构件生产者的信誉等,作为全局评估过程的输入参数。从每个非空的候选队列中,最多选择一个构件组装生成目标系统。

如 3.2 小节所述,“横切”需求涉及到多个构件的交互与协作,构件组合对“横切”需求的满足度只能在全局评估阶段进行。目标系统的全局需求匹配度 GFS 由两部分相加而得,一部分是所有组成构件 LFS 之和,另一部分则是部分构件组合对“横切”需求匹配度之和。

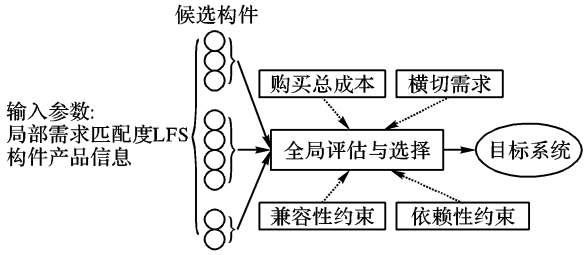


图 3 全局评估与选择过程

全局选择的目标是确定在给定约束条件下使得 GFS 最大化的最优构件组合。MOTS 中定义了三种全局选择约束条件,即构件购买总成本约束、兼容性约束和依赖性约束。构件购买总成本指的是用于购买构件产品的成本上限。兼容性约束指某些构件产品由于采用的协议、开发平台和操作系统的不同导致无法同时被使用。依赖性约束描述的是构件产品间的依赖关系,例如在选择了某个构件的同时,必须要选择其他的构件。

作者在文献[16]中将全局构件选择半形式化地表示为一个非线性的优化问题,该问题可以利用数学软件 LINGO 对其建模并求解。LINGO 是一种通用建模语言,采用一组通用的符号对线性和非线性优化问题建模。LINGO 同时也是求解器,可用于对该优化问题进行求解。

4 结语

大型软件系统中,构件间相互依赖,因而难以单个进行评估。现有的针对单个构件的选择方法很容易导致从全局角度来说非最优解的出现。由于系统分解是识别组成系统的多个构件的必要先行步骤,本文提出的多构件选择方法 MOTS 建立在一个成熟的、经过验证的系统分解技术 FCD 基础之上。系统分解的过程中,识别出局部需求和“横切”需求。局部需求被划分到功能模块中作为对构件进行局部评估的准则。系统开发的初期阶段,用户需求是粗略的、灵活的。在系统分解的过程中,需求逐步精化。随着更多的候选构件被识别出来,某些构件特性被采纳作为用户需求。由于 FCD 系统分解结果呈现层次结构,分解的过程自然地支持多种粒度构件的识别。在全局选择环节,首先评估候选构件组合对“横切”需求的匹配度,然后确定一个在给定的购买成本约束、兼容性约束和依赖性约束下具有最大全局需求满足度的最优构件组合。全局选择问题被定义为一个非线性的受限的优化问题,可利用数学软件 LINGO 对其建模并求解。

参考文献:

[1] WALLNAU KC, HISSAM SA, SEACORD RC. Building Systems from Commercial Components[M]. Addison-Wesley, 2001.

[2] BAUM L, BECKER M, GEYER L, *et al*. Mapping Requirements to Reusable Components using Design Spaces[A]. Proceedings of IEEE Int'l Conference on Requirements Engineering (ICRE-2000)[C]. Schaumburg/Chicago, USA, 2000.

[3] CHANG CK, HUANG JC, HUA S, *et al*. Function-Class Decomposition: A Hybrid Software Engineering Method[J]. IEEE Computer, 2001, 34(12): 87-93.

(下转第 867 页)

```
<Process-Service >
  …
  <SynchronizedTransitions >
  <Sources >
    <Source Id = "Send Components " >
    <Description > Components have already been send out
    </Description >
    </Source >
  <Sources >
    <Targets >
      <Target Id = "Receive Finance" Index = "0" >
        <Description > Finance receiving can be done only when
          the money has been sent
        </Description >
      </Target >
    </Targets >
  </SynchronizedTransitions >
  <DataTranspotations >
    <Observables >
      <Observable DataId = "ReceiptId" ActivityId = "Send Receipt" >
        <Description > The receipt Id can be transportated after
          sending receipt
        </Description >
      </Observable >
    </Observables >
  </DataTranspotations >
</Process-Service >
```

5　结语

文献[2,5]给出的语言对 Web 服务的支持限制在外部应用的层次上,不支持同异步调用、流程间的协作。BPEL4WS 只适合描述自动化的流程,基本上不支持人工干预,所调用的业务逻辑对应的实现也只有 Web 服务,所有应用都以 Web 服务的方式实现必然导致其在效率方面的不足,同时其复杂的设计也导致目前少有成熟支持 BPEL4WS 的工作流引擎[12]。同时,包括 XPDL2.0 在内的这些语言采用直接调用 Web 服务接口的方式提供对服务互操作的支持,而封装成 Web 服务的流程接口并不存在统一的标准,这给流程的适配和协作带来了困难。

相比而言,本文基于XPDL提出的PS-XPDL扩展语言模型提供了对所有流程协作场景的支持,同时拥有一些其他支持 Web 服务的流程定义语言不具备的特性。首先,它支持人工参与活动,不要求所有的外部应用都必须以 Web 服务方式实现;其次,由于其将服务具体方法和参数定义交予执行支持层协议,语言本身不涉及方法调用的技术细节,使流程定义的复杂度降低,同时基于标准协议也利于流程服务集成和实现。最后,其语言模型在数据传输元素上的精确定义使得在 XML 编码条件下能高效传输数据。

参考文献:

[1] 龚晓庆. 基于 Web 服务的分布式工作流管理系统研究[D]. 西安: 西北大学, 计算机软件与理论研究所, 2004.

[2] LI HX, LU ZS. Decentralized Workflow Modeling and Execution in Service-Oriented Computing Environment[A]. Proceedings of IEEE International Workshop on Service-Oriented System Engineering (SOSE'05)[C]. 2005. 29 - 36.

[3] Workflow Management Coalition. Workflow Process Definition Interface-XML Process Definition Language1.0, WFMC-TC-1025[S].

[4] WEERAWARANA S, CURBERA FP. 使用 BPEL4WS 的业务流程[EB/OL]. http://www-128.ibm.com/developerworks/cn/webservices/ws-bpelcol/index.html, 2002 - 06 - 04/2006 - 9 - 22.

[5] WU ZH, DENG SG, LI Y. Introducing EAI and Service Components into Process Management[A]. Proceedings of 2004 IEEE International Conference on Services Computing (SCC'04)[C]. 2004. 271 - 276.

[6] Workflow Management Coalition. Workflow Process Definition Interface-XML Process Definition Language2.0, WFMC-TC-1025[S].

[7] Organization for the Advancement of Structured Information Standards. Asynchronous Service Access Protocol[[EB/OL]. http://www.oasis.org/, 2006 - 08.

[8] Workflow Management Coalition. Wf-XML 2.0[Z]. 2004.

[9] Workflow Management Coalition. Workflow Standard-Interoperability, WFMC-TC-1012[S].

[10] 李红臣, 史美林. 工作流模型及其形式化描述[J]. 计算机学报, 2003, 26(11): 1456 - 1463.

[11] WfMC. Workflow Management Coalition: The Workflow Reference Model, TC0021003[Z].

[12] 马华. 分布式应用集成中的面向服务工作流研究[D]. 长沙: 中南大学, 计算机应用技术研究所, 2006.

(上接第 862 页)

[4] BARRY B, YE Y, JESAL B, *et al.* Composable spiral processes for COTS-based application development[A]. 4th International Conference COTS-Based Software Systems (ICCBSS), LNCS3412[C]. 2005.6 - 7.

[5] KONTIO J. OTSO: A Systematic Process for Reusable Software Component Selection, CS-TR-3478[R]. University of Maryland Technical Reports, 1995.

[6] ALVES C, CASTRO J. CRE: A Systematic Method for COTS Components Selection[A]. XV Brazilian Symposium on Software Engineering (SBES)[C]. Rio de Janeiro, Brazil, October 2001.

[7] VIGDER M, MCCLEAN T, BORDELEAN F. Evaluating COTS Based Architectures[A]. Proceedings of the ICCBSS 2003[C]. Ottawa, Canada, 2003.

[8] CARNEY D. COTS Evaluation in the Real World[R]. SEI interactive, Carnegie Mellon University, 1998.

[9] MAIDEN N, NCUBE C. Acquiring COTS Software Selection Requirements[J]. IEEE Software, 1998, 15(2): 46 - 56.

[10] LAUESEN S. COTS tenders and integration requirements[A]. Proceedings of 12th IEEE International Requirements Engineering Conference[C]. 2004. 166 - 75.

[11] BUHR RJA. Definition and Classification of COTS: A Proposal[A]. Proceedings of ICCBSS[C]. Orlando, Florida USA, 2002. 165 - 175.

[12] BUHR RJA. Use-case Maps as Architectural Entities for Complex Systems[J]. IEEE Transactions Software Engineering, 1998, 24(12): 1131 - 1151.

[13] HSIA P, YAUNG AT. Another Approach to System Decomposition[A]. IEEE Proceedings of COMPSAC'88[C]. 1988. 75 - 82.

[14] NCUBE C, DEAN JC. The Limitations of Current Decision-Making Techniques in the Procurement of COTS Software Products[A]. Proceedings of ICCBSS[C]. Orlando, Florida, USA, 2002. 176 - 187.

[15] 盛津芳, 陈松乔, 王斌. 软件功能需求驱动的商业构件评估[J]. 计算机工程, 2005, 31(21): 99 - 101.

[16] SHENG JF, CHEN SQ, WANG B. COTS Evaluation and Selection Based on Requirements Decomposition[J]. Chinese Journal of Electronics, 2005, 14(1): 62 - 67.