文章编号：2096-1618(2018)05-0498-05

基于SOA的服务构件粒度划分方法的研究应用

杨 晓[1]， 舒红平[2]， 徐 虹[1]， 刘 魁[2]
(1.成都信息工程大学计算机学院，四川 成都 610225；2.成都信息工程大学软件工程学院，四川 成都 610225)

摘要：分析当前SOA服务粒度划分研究现状，提出一种基于SOA的服务构件粒度划分方法：两级拆分抽象划分方法。该方法从功能模块分离出业务构件，一方面对业务构件向下拆分得到基础构件，另一方面对业务构件向上抽象得到行业构件。该划分方法满足服务构件粒度划分基本原则，得到的3类服务构件能够有效适用于基于SOA的软件开发。最后，结合实际的软件开发案例对此划分方法进行分析说明。

关 键 词：SOA；服务构件粒度划分；两级拆分抽象；基础构件；业务构件；行业构件
中图分类号：TP399 **文献标志码**：A
doi：10.16836/j.cnki.jcuit.2018.05.003

0 引言

SOA(Service Oriented Architecture)早在1996年由Gartner提出，其核心理念是业务驱动，采用松耦合、灵活的体系架构满足软件开发需求。在SOA架构中，服务是核心的抽象手段，业务被划分为一系列不同粒度的服务，服务之间相对独立、可重用。

SOA及其相关软件设计是对传统软件设计技术的继承和发展[1]。从体系结构的角度来看，在SOA中有3种角色：服务请求者；服务提供者；服务代理(服务注册中心)[2]。三者之间的架构如图1所示。

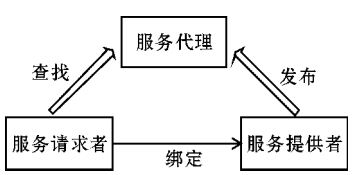

图1 SOA体系结构图

使用SOA架构进行软件开发需要对服务进行设计，设计过程中一个关键点即是服务建模。服务建模的难点主要在于对服务的粒度划分，服务粒度是指一个服务所包含的功能大小，服务粒度不能太大或者太小，而应该大小合适[3]。基于SOA的系统是由不同粒度的服务共同组成，一种服务对应一种服务构件，两者之间具有一一对应的关系。

当前比较流行的服务粒度划分有自上而下、自下而上2种划分方法，如下述所示。

自上而下。从业务流程向下，来定义业务服务和组件服务。业务服务面向用户完成具体的业务功能；组件服务主要运算处理业务状况，负责各个业务角色之间的交互，仍然处于业务层面[4]。此方法虽然符合SOA业务驱动服务的思想，但是业务服务和组件服务都定位在业务，没有一个层次清晰的粒度划分策略，可能会造成粒度划分整体偏向业务层面，无法准确定位服务的层次。

自下而上。从技术层面向上，把对数据的采集、存储、计算、传输等划分为基础服务，再从底层细粒度出发向符合业务的粗粒度组合，通过服务依赖关系图的划分来确定基础服务的粒度大小[4]。这种方法虽然有一定的灵活性，但可能会造成服务粒度划分过细，不完全符合SOA业务驱动服务的思想。

上述2种服务粒度划分方法仍然存在一定的不足和缺陷。然而，不恰当的服务粒度划分会增加系统服务设计的复杂性和系统维护的难度，甚至不正确的服务构件划分规则和策略还会导致致命后果，从而降低软件可靠性，影响整个软件系统[5]。

1 服务构件粒度划分

1.1 构件粒度划分关键原则

为了保证服务构件粒度划分方法是合理并高可用，需要遵循服务构件的关键原则如下。

(1)可复用性原则：这是服务设计的最基本原则，也是服务化的根本。

(2)抽象提炼原则：将功能目标类似的服务进行抽象，将其隐藏的部分模式和方法进行提取。

(3)松散耦合原则：保证功能的单一性，服务构件输入输出参数的粒度与服务构件功能匹配。

收稿日期：2018-04-15
基金项目：四川省科技计划资助项目(2017JY0196)

(4)可组合性原则:将具体业务流程划分为各个独立的逻辑单元,能组合这些逻辑单元形成新的服务构件。

(5)无状态性原则:服务构件的内部对于服务请求者是透明的,提高服务请求者和提供者的伸缩性[6]。

(6)可自治性原则:服务构件的逻辑单元由自身边界内部控制,不受外界条件影响[7]。

(7)业务、IT 对齐原则:服务构件必须有针对性的业务含义,通过契约设计方法规范服务参与各方职责[1]。

(8)灵活独立原则:适当粒度的服务构件能使服务在各角度进行服务装配;通过松散耦合技术减少服务消费者和提供者间技术依赖性。避免了那些需要在操作之间维护过渡状态的服务构件,每个服务构件都要保持高度的独立性[8]。

1.2　服务构件粒度划分方法

目前,关于服务构件的粒度划分并没有一个权威的国际标准进行约束和确定。为了解决上述服务划分粒度的不足,在遵循服务构件粒度划分原则基础上,提出一种“两级拆分抽象”服务构件粒度划分方法。

1.2.1　业务分析切入

首先从功能模块出发,以业务为驱动,对功能模块按照实际业务需求进行工作流程分解,分析完成该功能模块所对应的完整业务流程(通常一个完整的业务流程包含多个流程节点),整个业务流程对应实现的具体业务即属于业务服务,它能够代表完成一个完整的业务功能,在实际操作时可以直接通过业务构件的服务接口直接使用该服务提供的功能。

1.2.2　业务向下拆分

对业务构件向下进行拆分(即对功能模块向下进行拆分),每个业务构件对应的业务工作流程中的每个流程节点是基础构件的来源和基础。业务工作流程中每个流程节点隶属于业务层面,需要对每个业务流程节点从业务层面向下拆分,分析实现此业务所需要的基础流程节点。例如,一个新增班级信息的业务,按照业务流程分析,在新增操作节点完成后,会有一个返回“新增成功”提示信息的流程节点,此节点是在业务层面。接着对此节点向下进行拆分(在拆分过程中要从具体到广泛),可以得到一个通用的“返回生成数据”的基础流程节点,此基础流程节点对应的即是基础构件,要想返回“新增成功”只需对此基础构件进行配置就可达到效果,在使用过程中直接通过基础构件的服务接口直接使用该服务。

1.2.3　业务向上抽象

对业务构件向上进行抽象(即对功能模块向上进行抽象)。在某一具体行业内,一个类似的业务对应的业务流程具有相似性,在向上抽象过程中,采用共性提取法,将类似业务中的共性提取出来,作为行业构件的基本工作流程,并且提取的工作流程必须满足此类业务所代表的共性且能够独立完成一个具体业务。整个流程对应完成的功能即为行业构件提供的功能。例如:对于制造行业,新增采购信息这个业务的流程基本上都满足必要的 3 个流程节点(即验证输入数据→新增数据→返回新增结果)。然而,有些新增采购信息业务会在返回新增结果节点后,还有一个查询采购信息节点,但是,此时在对业务构件抽象过程中,只需提取类似业务中的共性,对于一些业务的特性,可以忽略。通过这样的方式可以得到具体的行业构件,在使用过程中可以直接通过行业构件的服务接口直接使用该服务提供的行业业务功能,也可以在行业构件上增删流程节点以满足业务需求。

根据上述的 3 个基本步骤可以分析出服务构件粒度划分算法如下。

```
Algorithm 1 Component_Granularity_Partition_Algorithm
1:input:functional module(简称:fun_mod)
2:if fun_mod has workflow:
3:do:analyze workflow to basicflow (split down)
4:    change basicflow into BasicComponent
        (BasicComponent←basicflow)
5:do:analyze workflow to commonflow(abstractedupwards)
6:    change commonflow into IndustryComponent
        (IndustryComponent←commonflow)
7:else:
8:do:check and modify functional module
9:    goto: 2
10:do:analyze workflow into BusinessComponent
        (BusinessComponent←workflow)
11:output:BasicComponent, BusinessComponent,
        IndustryComponent
```

通过上述服务构件粒度划分方法将得到 3 种不同粒度的构件,分别是基础构件、业务构件、行业构件。在实际软件开发过程中,这 3 类构件可以根据业务需求进行服务装配,完成具体的业务功能。无论是基础构件、业务构件或是行业构件,在其使用过程中都只能够调用服务的完整功能,如果需要完成的某一功能是调用某一服务的部分功能,这样的服务构件调用是完全禁止的。在服务装配过程中,当业务流程(BPEL)变动幅度较小时,只需简单调整(插拔)个别的基础服务

构件即可实现新的业务功能[9]。这样不仅满足 SOA 业务驱动服务的思想,而且能够保证高度的松散耦合性、灵活性、重用性和独立性。

1.3 服务构件间的关联

根据上述的服务构件粒度划分方法得到服务构件具有各自的特点,并且不同粒度的服务构件之间在数量和逻辑层次结构关系上都具有一定的联系。

基础构件是系统提供的最小粒度服务构件(原子服务构件),具有高可重用性,与基础服务相对应,记为 α。它只能够完成简单、独立的某一个功能操作(如数据库新增服务、数据验证服务),与具体的业务没有明显的关联。

业务构件是根据实际业务,将基础构件按照业务流程进行服务组合而形成的服务构件(如新增班级信息服务),与业务服务相对应,记为 β。这些业务构件能够在类似的系统中被多次使用,并且业务构件在名称上能够明确体现其与具体业务相关联。

行业构件是以行业为背景,用于满足行业应用的粗粒度构件,与行业服务相对应,记为 γ。行业构件的命名与行业名称相关(如:制造业新增库存数据业务),因此通过行业构件的名称也能够明显地辨别出该服务构件是否属于行业构件的范畴。

1.3.1 构件数量关系

从细粒度到粗粒度的服务构件划分过程中,三者之间的数量关系符合菱形定律,且逻辑关系逐层粒度更粗,其服务构件组成图如图 2 所示。

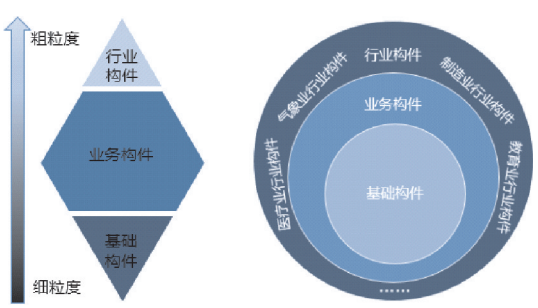

图 2 服务构件组成图

基础构件作为最细粒度的服务构件,是地基形式的存在,涉猎于各项基本操作,其数量介于行业构件和业务构件之间。业务构件与具体的业务绑定,任何一个系统都具有多种业务,可能多个业务构件的完成都需要调用同一个基础构件,因此,业务构件的数量在 3 个层次构件中占比最大,数量最多。行业构件是在业务构件的抽象上形成的,具有一定的通用性,数量占比是 3 个层次中最少的。

3 个层次的服务构件数量总和应该满足的数量关系:

$$\sum\beta > \sum\alpha > \sum\gamma \tag{1}$$

1.3.2 构件结构关系

不同粒度的构件之间,其逻辑层次结构也具有一定的包含关系,具体的构件结构关系图如图 3 所示。由结构关系图可以得出,业务构件是由基础构件按照业务流程(BPEL)装配而成,行业构件可以在业务构件的工作流程上增加新的基础构件。

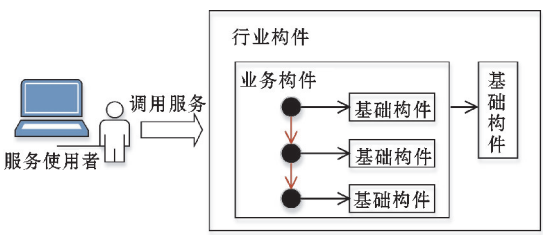

图 3 服务构件结构关系图

2 应用实例

2.1 实验实例

通过一个具有代表性业务功能并且业务流程清晰的实验案例来分析说明服务构件粒度是如何进行划分的。此实验是以招投标管理系统中,一个具体的功能模块(新增招标申报表)作为实验实例进行说明。

在传统的软件开发过程中,对新增招标申报表功能模块的设计开发是直接按照“烟囱”式设计开发模式进行的,系统中每个模板是一个独立的个体,最终产物只是一个新增招投标申报表功能模块,并没有任何沉淀。在有类似业务出现时,设计开发人员需要重新设计开发一个类似的功能模块,不能够有效利用之前的成果,软件复用率低。

基于上述问题,采用基于 SOA 的服务构件粒度划分方法,通过服务构件进行软件开发能够有效提高软件复用率。

以新增招标申报表功能模块为例。首先,将此功能进行业务流程分解,能够得到完成此功能模块需要 3 个业务流程节点,分别是:验证新增数据→新增申报表信息→返回“新增”成功信息,其对应的业务流程图如图 4 所示。因为申报表属于业务单据,所以,新增招标申报表业务对应着一个业务构件,即新增业务单据到数据库业务构件。然后,对这个业务流程中的每个流程节点进行向下拆分,验证新增数据拆分为数据验证;新增申报表信息拆分为数据新增;返回“新增”成功信息拆分为返回生成数据,拆分后其对应的基础流

程图如图 4 所示。

每一个基础流程节点都对应着基础构件,以“数据验证”为例,数据验证可以分为用户信息验证、表单验证、数据内容验证等验证方式,这些都能够作为基础构件存在。在此实验中,根据实际的业务可得,此处需要数据内容验证作为此流程节点的基础构件。同理可得,实验中,每一个基础流程节点都对应着一个基础构件,如图 4 所示。

接下来,对业务构件进行向上抽象。招投标管理系统属于服务行业业务,分析服务行业完成新增招标申报表所必须的业务流程(行业共性流程)。要完成此业务流程必须要有 2 个流程节点,即:验证新增数据→新增申报表信息,其对应的行业共性流程图如图 4 所示。此行业流程封装的服务即为行业构件,行业构件在业务构件和基础构件上进行服务装配即可得到。

因此,通过上述服务粒度划分方法的拆分抽象,能够得到整个实验案例的服务构件粒度划分过程如图 4 所示。

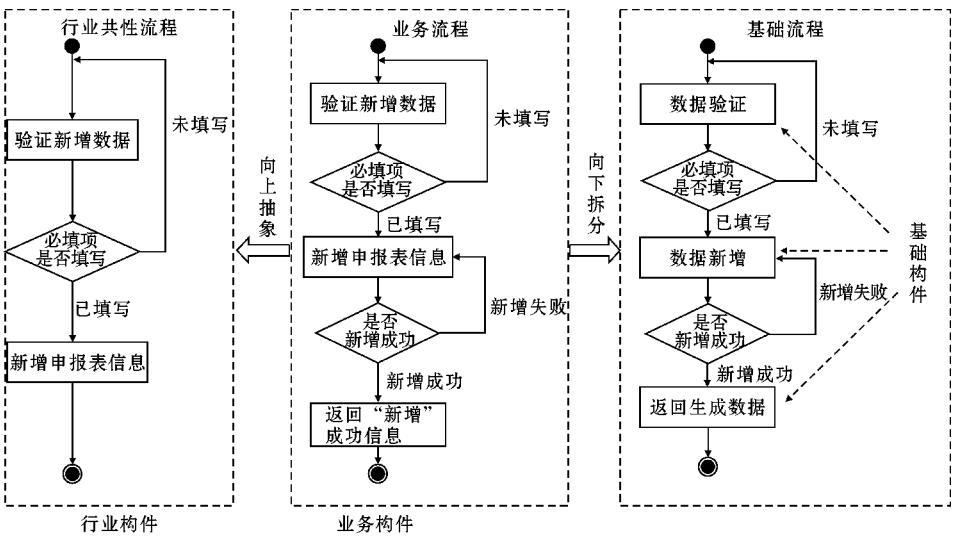

图 4　新增招标申报表粒度划分过程图

综上,经过对新增招标申报表功能模块的分析,能够得到 3 种粒度的服务构件,分别是行业构件:服务业新增招标申报表构件;业务构件:新增业务单据到数据库;基础构件:数据验证构件、数据新增构件、返回生成数据构件。得到的所有服务构件都存储在服务构件中心中,供软件开发重复利用。

2.2　结果分析

2.2.1　结果说明

在满足服务构件粒度划分关键原则前提下,从新增招标申报表功能模块中,可以划分出:1 个行业构件、1 个业务构件、3 个基础构件。经过沉淀和归纳,在实际的业务场景中,这 3 类构件都可以使用在相同或者类似的业务需求中,避免了传统软件开发过程中软件复用率低等问题,大大提高了软件的复用率,也提高了软件开发整体效率。

通过以功能模块为基准进行剖析,可以首先定位业务构件,然后剖析出更细粒度的基础构件;接下来也可以对业务构件进行抽象总结,提升为更粗粒度的行业构件。在整个实验验证过程中,能够清楚地看到基于 SOA 的服务构件粒度划分的方法和结果,满足实际的软件开发需求,使软件开发变得更加灵活。

2.2.2　结果总结

总之,以 SOA 为基础的服务构件可以通过不同层次进行粒度划分,从细粒度到粗粒度层层抽象,从粗粒度到细粒度面面具体,各个层次服务构件均有显著特点。在同等条件下,以 SOA 服务思想进行软件开发相比于传统的基于模块的开发方法具有更高可用性和灵活性[10]。随着时间的积累,通过此服务构件粒度划分方法,可以沉淀大量的各种层次的服务构件,能够为构件中心累积大量构件,明显提高了软件开发的构件复用率。

3　结束语

在大数据、人工智能迅速发展的前景下,软件开发会面临业务多元化、业务需求复杂化,以 SOA 面向服务的方法来进行软件开发将成为越来越多企业选择和使用的方法[11-12]。服务不是越多越好,要以一定的方法或准则来设计服务的划分原则,以文中服务构件粒度划分方法,可以较准确地把握服务的精细度,在遵循 SOA 架构的标准和规范上,真正地发挥 SOA 架构的优势。

参考文献:

[1] 毛新生. SOA 原理方法实践[M]. 北京:电子工业出版社,2007:35-36.

[2] 王珂珂,张立朝,潘贞,等. 业务驱动的地理信息服务粒度划分[J]. 北京测绘,2010(1):4-7.

[3] 范颖. 面向实体划分服务粒度[J]. 河南广播电视大学学报,2009,22(2):108-109.

[4] 蒋延耀,康维,乐文静. 一种混合模式两级抽象服务粒度划分法[J]. 电脑知识与技术,2009,5(34):9782-9784.

[5] 杜攀,徐进. SOA 体系下细粒度组件服务整合的探讨[J]. 计算机应用,2006,26(3):700-702.

[6] 梁利. 基于 SOA 粒度关联的服务组合模型研究[J]. 电脑与信息技术,2013,21(4):57-62.

[7] 唐立文,岳峥. 基于 SOA 的航天发射场服务构件设计与应用[J]. 装备指挥技术学院学报,2011,22(5):104-108.

[8] Eric Newcomer, Greg Lomow. Understanding SOA with Web Services(中文版)[M]. 徐涵(译). 北京:电子工业出版社,2006.

[9] 余浩,朱成,丁鹏. SOA 实践—构建基于 Java Web 服务和 BPEL 的企业级应用[M]. 北京:电子工业出版社,2009.

[10] 李攀. 基于 SOA 架构和构件技术的软件开发[J]. 电子技术与软件工程,2016:61.

[11] 刘洋,程春. 基于 SOA 服务构件封装技术应用研究[J]. 信息系统工程,2015:67.

[12] 吕海燕,车晓伟. 数据仓库中数据粒度的划分[J]. 计算机工程与设计,2009,30(9):2323-2328.

Research and Application of Service Component Granularity Partition Method based on SOA

YANG Xiao[1], SHU Hong-ping[2], XU Hong[1], LIU Kui[2]

(1. College of Computer Science, Chengdu University of Information Technology, Chengdu 610225, China; 2. College of Software Engineering, Chengdu University of Information Technology, Chengdu 610225, China)

Abstract: This paper analyzes the current research status of SOA service granularity partitioning and proposes a SOA-based service component granularity partitioning method: two-level split abstraction method. This method separates the business components from the functional modules. On the one hand, the business components are split down to get the basic components. On the other hand, the business components are abstracted upwardto get the industry components. The partitioning method satisfies the basic principle of granularity partitioning of service components, and the obtained three types of service components can be effectively applied in software development based on SOA. Finally, this division method is analyzed based on the actual software development case.

Keywords: SOA; service component granularity partition; two-level split abstraction; basic component; business component; industry component