Anti-Patterns in Evolutionary Design of SOA Research

ZHAO Qing, LIU Yu-yan, JIE Yong-gang, YUE Qiang
Modern Education Technology Centre of Kunming University, Yunnan Kunming, China
E-mail:anpennyhardway@gmail.com

Abstract: Enterprise SOA (Service-oriented architecture) projects in the process of implementation have to build strong system architecture, which in order to registered enterprises needs a variety of services. Anti-Patterns is a Patterns technology, but it is different from traditional model function. Anti-Patterns is concerned that analysis in enterprises software projects process of implementation in the negative influence factor. In general, the risk factor from three roles, they are developer, architect and manager. Although the different negative impact factors require different solutions, but Anti-Patterns can provide diverse refactoring solutions. Used of this technology will make project risk is controlled within the minimum range by implementing sustained refactoring solutions. So, this approach shows that build the high quality software system architecture will benefit from Anti-Patterns technology, which to suffice for enterprises service modeling process need is necessary. Therefore, Anti-Patterns in the use of technology in evolutionary design process of SOA projects is a significant research and exploration.

Keywords: Anti-Patterns；Refactoring；SOA；

反模式在 SOA 演进式设计中的研究探索

赵卿，刘渝妍，解永刚，岳强
昆明学院现代教育技术中心, 云南昆明, 中国, 650214
E-mail:anpennyhardway@gmail.com

【摘要】 企业在实施 SOA 项目过程中需要建立一个良好的系统架构，目的是在系统架构中注册企业所需要的各种不同服务。为了满足系统架构设计所提出的高质量要求，在软件项目实施过程中采用反模式技术分析的设计过程中的负影响力因素，进而实施持续的重构解决方案把开发过程中的风险因素控制在最小范围内，实现高质量的系统架构满足服务建模的要求。所以，在 SOA 的演进式设计过程中采用反模式技术是一项有意义的研究和探索。

【关键词】反模式；重构；SOA；

1. 引言

企业级的软件架构设计[1]目标通常希望软件架构具备低复杂性、高可重用性、高适应性以及业务功能和技术支撑分离等特性。因此，从 SOA（Service-oriented architecture）的概念提出至今为止世界上有很多不同类型的企业都希望运用 SOA 设计原则[2]帮助自己在业务流程重组中实现企业业务信息化集成。目的是企业能够在面向服务处理的流程中实时、准确的执行企业制定的各项服务策略，以满足业务客户的各种服务需求。可是，在实施以 SOA 设计原则为标准的企业信息化集成过程中，始终没有一种良好的开发模型能够充分满足每个企业不同服务的要求。这说明实施以 SOA 设计原则为标准的企业信息化集成过程具有很强的动态性和不确定性。因此，当前很多组织机构都在研究和探索从现有的软件系统开发模型中找出能够较好适用于 SOA 项目实施过程的开发模型。目前，这个领域内相关的研究重点和热点主要是以迭代开发策略为基础的开发模型。本文选用基于迭代式开发策略的演进式开发模型为基础，探讨演进式开发模型在 SOA 项目实施中运用可以采用的技术路线。

2. SOA 设计原则及特性

软件系统分析设计方法的发展从最初的结构化分析设计逐步过渡到面向服务的分析设计方法，每一种

978-1-935068-23-5

分析设计方法都有自身的设计原则和特性，基于 SOA 分析设计的方法是一种始终以服务建模为核心的系统分析方法，因此基于 SOA 的分析设计原则可以从：基本层、服务支持层、业务流程层，这 3 个层次[3]中找出对应的分析设计原则，如表 1 所示：

从表 1 中可以看出 SOA 分析设计过程中不同层次的分析设计原则应遵循的重点。依据这些分析设计原则，可以作为演进式开发模型的基本指导原则，为开发过程中应采用什么样的技术路线和策略确立了明确的标准。

3. 演进式开发模型

一般基于迭代策略的开发模型[4]有：增量式、演进式、增量提交式、单次迭代式，结合 RUP（Rational Unified Process）的 4 个工作过程（先启、精化、构建、产品化）分析这 4 种不同开发方式的特性，如表 2 所示：

因为 SOA 项目的特点一般是架构风险较大，在系统建立初期需要明确架构的结构，当架构设计完成的时候对应的系统服务也注册成功。所以演进式开发模型更适用于 SOA 项目的实施。演进式开发存在对在演进式开发过程中需要采用那些技术路线实现风险降低，以保证最终软件系统的可靠性。开发风险是软件系统开发过程中不可避免的一种重要因素，在这里可以把风险的各种影响力看作是一种原力[5]的组成。所以，在演进式开发过程中需要引入反模式技术分解原力成分达到有效的降低风险。

4. 反模式演进开发过程

模式设计先驱 Jim Coplien 在对反模式的研究活动中指出“对反模式的研究是一项重要的研究活动。仅仅在一个成功的系统中展现出‘好’的模式是不够的；还必须说明这些模式不会出现在不成功的系统中。与之相似，揭示出特定模式（反模式）出现在不成功的系统中，而不会出现在成功的系统中同样有益。”，所以反模式是关注软件系统开发中负面影响力的一个新的研究方向。尽管模式设计已经被广泛的应用在各种软件系统设计中，包括 SOA 项目。虽然过去在 SOA 项目分析设计中采用反模式技术的例子并不多见，但是在 SOA 项目中已经采用了一些类似反模式技术的方法，把 SOA 软件系统开发过程中存在的风险因素按层次、粒度、级别、领域、人员角色等标准对相应的风险因素进行有效的划分，达到对开发风险有目的实施控制。随着反模式技术的不断发展和完善，在 SOA 软件系统开发过程中从软件系统的层次上可以把反模式的类型分为[5][6]3 类：软件开发性反模式、软件架构性反模式和软件项目管理性反模式，其特点如下：

1) 软件开发性反模式：指出开发过程中可能遇到的技术性问题和对应处理的解决方案。属于基本层的反模式，主要面向的是程序设计一类的人员。
2) 软件架构性反模式：确立规划系统结构过程中可能遇到的问题和对应的解决方案。属于服务支持层（系统架构）的反模式。主要面向的是系统架构分析师一类的人员。
3) 软件项目管理性反模式：指开发团队在软件系统开发过程和软件项目管理过程中可能遇到的问题和对应的解决方案。属于业务流程层的反模式，主要面向的是软件系统项目的管理者一类的人员。

SOA 项目分析设计过程中采用什么样的架构设计才能满足对应的服务模型是一项重要的任务，这也是 SOA 项目分析设计的重点之一。经历多次 SOA 项目实践后，人们已经在面向服务领域内的分析设计过程中找到一些存在的反模式[7]，如下：

1) 多头服务：一个服务承担了其他多个服务的职能，服务可用性差，耦合度高。
2) 过小服务：一个服务需要多个服务的支持才能完成处理，服务可用性差，耦合度高。
3) 烟囱服务：在不同多个独立的服务中，存在相同功能需求被差异化实现的实例，使得软件系统面向用户服务处理过程中功能出现不一致性。从而降低了服务的可靠性和正确性。
4) 客户完成服务：一个服务集合在客户组件中，失

Table 1. Level of SOA analysis and design
表 1 .SOA 分析设计层次

层次	基本分析设计原则	服务建模
基本层	重用性，粒度，模块化，可组合性，构件化以及交互操作性	服务实现
服务支持层	符合标准(通用的或行业的)	服务定义
业务流程层	服务的识别和分类，提供和发布，监控和跟踪。	服务识别

Table2. Features of iterative strategy development model
表 2 迭代策略开发模型特性

开发模型	适用架构风险度	架构迭代次数	项目规模	主要工作阶段
增量式	适用较小风险	多次	小	构建
演进式	适用较大风险	多次	大	精化
增量提交式	适用一般风险	多次	中	产品化
单次迭代	适用较小风险	一次	小	产品化

去原有的通用性脱离服务层，使得对于其它不同客户应提供的服务出现缺失。

以上存在的反模式指出面向服务架构设计过程中可能出现的负面设计，要解决这些问题可以采用对应的重构技术。通常采用的重构技术解决方案是：接口划分、接口合并、技术服务层、跨层重构，如表 3 所示：

通过以上分析并结合 RUP 分析设计过程，提取演进式开发执行过程片段可以得到：如图 1 所示：

通过图中演进式开发片段的流程分析，可以看到在 SOA 项目分析设计实施过程中随着开发过程的推进，代价最高的是在产品化阶段的项目管理反模式引发的重构过程，代价最小的是架构性反模式引发的重构过程。因此，演进式开发在精化阶段的多次迭代可以充分有效的降低 SOA 项目在推进过程中的风险。经过精化阶段的多次迭代建立的系统架构也满足 SOA 分析设计的相关原则。反模式在整个开发过程中始终是一种驱动力，也是一种持续的集成过程。在持续的集成过程中避免错误方案的延伸，并采取较小的代价最大限度的提升系统架构的质量。

Table 3. Anti-patterns and Refactoring solutions
表 3.反模式与重构方案

反模式	重构方案	层次	粒度	反模式类型
多头服务	接口划分	基本层	细	软件开发性反模式
过小服务	接口合并	基本层	细	软件开发性反模式
烟囱服务	技术服务层	服务支持层	中	软件架构性反模式
客户完成服务	跨层重构	业务流程层	粗	软件项目管理性反模式

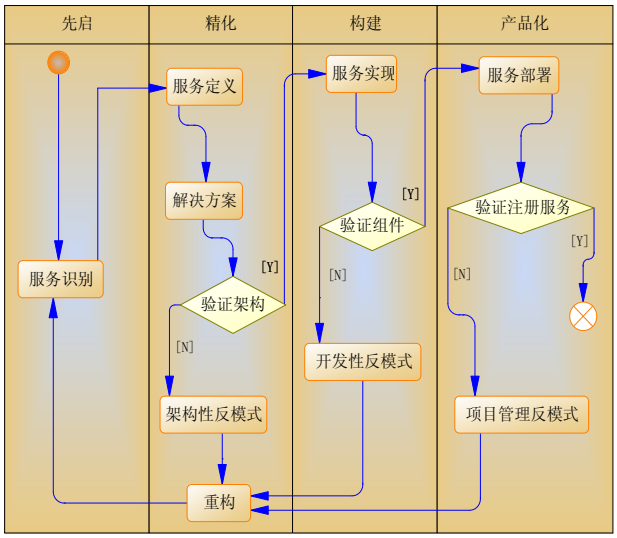

Figure 1. Fragment of Evolutionary development process
图 1.演进式开发过程片段

5. 总结与展望

反模式技术引入到 SOA 的演进式开发过程中后能使 SOA 项目在分析设计过程中从 3 个层次对软件系统开发过程中的风险进行控制。因此，SOA 演进式开发过程在反模式技术的支持下避免了错误的延伸，同时驱动重构方案的持续进行，增强企业在实施基于 SOA 原则的信息集成系统过程的可控制性，使得软件系统架构的质量得到了有效保证。

基于反模式技术的 SOA 演化开发过程可以推广为一个模型，如下图所示：

从图中看到改进的演进式开发模型通过对未知和已知错误的有力控制，把风险因素作用的范围控制在较低水平有利于软件系统架构的成功实现。引入了反模式技术的演进式开发过程实质上从影响软件系统开发的两个对立因素出发，在单纯的模式复用路线上，建立负影响力的因素主导的反模式技术路线，这导致

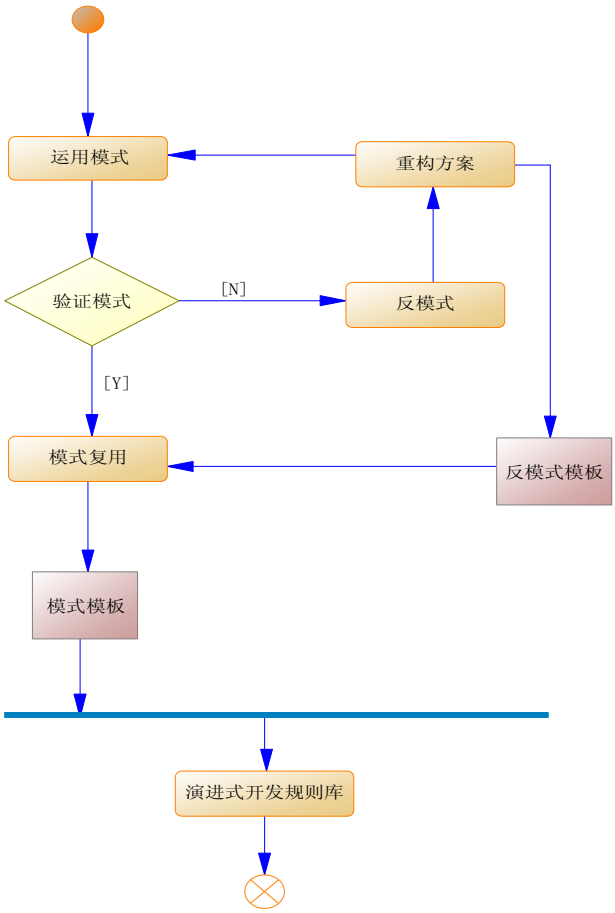

Figure 2. Improved Evolutionary development process model
图 2.改进的演进式开发过程模型

978-1-935068-23-5 © 2010 SciRes.

重构解决方案的提出，目的是在开发过程中尽量的避免未知错误的扩散和延伸。反模式现在还在不断的发展完善，项目实践中还会产生很多不同的反模式，通过反模式的模板化定义，把这些反模式不断的积累起来，反模式才会成为一种新的模式，代表一类应该避免的错误以及对应的解决策略，只有这样做才能使反模式技术具备良好的可推广性。才能在演进式开发规则库中通过不断的积累和实践，构建出各种适用于不同软件系统分析设计的规则与模板，实现软件分析设计复用的最终目标。

References (参考文献)

[1] Dirk Krafzig Karl Banke, Dirk Slama. Service-Oriented Architecture Best Practices [M].TISGHUA UNIVERSITY PRESS,2006.P4-7.
Dirk Krafzig Karl Banke, Dirk Slama. Service-Oriented Architecture Best Practices [M]. 清华大学出版社, 2006.P4-7.

[2] ZHANG Guang-sheng, JIANG Chang-jun, TANG Xian-fei, et al. Service-oriented enterprise application integration system description and verification [J] Journal of Software.2007 (12):
张广胜，蒋昌俊，汤宪飞等，面向服务的企业应用集成系统描述与验证 [J]. 软件学报, 2007, (12):

[3] Blazer Y. Improve your SOA project plans [M]. http://www.ibm.com/developerworks/webservices/library/ws-improvesoa/ 2004.

[4] FU Chun-yi.Iterative software development techniques[M]. IBM Rational Technical White Paper,2004,(9):
傅纯一.迭代化软件开发技术 [M]. IBM Rational 技术白皮书, 2004, (9):

[5] William J. Brown, Raphael C. Malveau, Hays W. McCormick III, Thomas J. Mowbray, etal. AntiPatterns Refactoring Sofeware, Architectures, and Projects in Crisis [M]. Ptpress, 2008.P3-65
William J. Brown, Raphael C. Malveau, Hays W. McCormick III, Thomas J. Mowbray, etal. AntiPatterns Refactoring Sofeware,Architectures,and Projects in Crisis [M]. 人民邮电出版社, 2008.P 3-65

[6] John, Evdemon.Service-Oriented Architecture Design Patterns and Anti-patterns [J]. Msdn, 2005, (6)P46-51.
John,Evdemon. 面向服务架构设计的模式与反模式 [J]. Msdn 开发精选, 2005, (6)P46-51.

[7] Bill Dudney, Stephen Asbury, Joseph K.Krozak, et al. J2EE Anti Patterns [M].CHINA MACHINE PRESS, 2006.P70-93.
Bill Dudney, Stephen Asbury, Joseph K.Krozak, et al. J2EE AntiPatterns [M]. 机械工业出版社, 2006.P70-93.