基于问题框架的行为驱动开发研究

高 宁 李 智
(广西师范大学计算机科学与信息工程学院 桂林 541004)

摘 要 问题框架(Problem Frames,PF)在需求工程研究领域中已经获得了广泛重视和研究。目前,问题框架的相关研究已经取得了较多成果,但如何从需求模型(问题图)平滑过渡到软件设计以及实现仍是一个有待解决的问题。文中对如何将问题图转换到用户场景文本以及通过用户场景来辅助行为驱动开发方法进行软件设计与开发进行了研究,提出了一种问题框架与行为驱动开发相结合的软件开发方法(PFBDD)。该方法能够帮助系统分析员将用户需求平滑过渡到软件设计及测试,从而避免软件项目中表达不一致带来的问题。此外,通过实例介绍了如何将此方法应用到一个车管业务排队系统中,并介绍了 Gherkin 语言和 Specflow 工具。该方法对于问题框架进一步走向实践具有重要的推动作用。

关键词 问题框架,问题图,行为驱动开发,测试驱动开发
中图法分类号 TP311 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2017.11.028

Research on Behavior-driven Development Based on Problem Frames

GAO Ning LI Zhi
(College of Computer Science and Information Technology,Guangxi Normal University,Guilin 541004,China)

Abstract Problem frames have been widely recognized and studied in the research domain requirements engineering. Although many research outcomes about problem frames have been obtained, how to transform requirements models (i.e., problem diagrams) into software design artifacts and implementations smoothly is still a difficult problem to be solved. In this paper, we demonstrated how to transform problem diagrams into user scenarios, which are then used to assist behavior driven design and development of software, and we proposed a software development method which combines problem frames and behavior driven design (PFBDD). The method can help system analysts to move smoothly from requirements analysis to software design and testing, thus avoiding inconsistency problems in software development. In addition, a case study was presented to demonstrate how to apply our method into a queuing problem of a vehicle management system. The Gherkin language and Specflow tool were introduced and applied in the case study. This method plays an important role in driving problem frames into further practical applications.

Keywords Problem frames, Problem diagrams, Behavior driven development, Test driven development

1 引言

随着人们对系统功能的要求越来越多样化,新型的智能化应用系统不断涌现,使得软件规模与复杂程度不断提高。然而,系统复杂程度越高,开发的难度就会越大,项目的失败率也就越高,从而导致软件项目返工甚至失败的例子屡见不鲜。很多失败的软件项目都证实了需求不明确是导致项目不能成功的关键因素。在应用问题框架进行需求建模时,缺乏一种从需求模型平滑过渡到软件设计以及实现的方法[1]。

DE Carvalho R A 等人[2]提出相对于文本描述的需求来说需求人员更喜欢使用图表的形式进行需求建模,并给出了一种通过 UML 状态图生成用户场景的设想,通过自动生成用户场景来辅助行为驱动开发。另外,DE Carvalho R A 等人[3]还提出了结合业务流程建模与 Petri Net,通过 Petri Net 生成用户场景,使得需求能够平滑过渡到软件设计与实现的方法。

本文利用需求建模时得出的问题图,导出与需求模型一一对应的用户场景文本,并且通过这些用户场景文本生成测试框架代码,其对软件的设计以及质量保证有一定的帮助,并且可以保持软件代码对需求的可追溯性。

到稿日期:2016-10-14 返修日期:2016-12-21 本文受国家自然科学基金(61262004),广西自然科学基金(2012GXNSFCA053010),广西科学研究与技术开发计划项目(桂科合1347004-22),2013年度广西高等教育教学改革工程项目(2013JGB121),广西多源信息挖掘与安全重点实验室开放基金(14-A-03-01),"八桂学者"工程专项经费资助。

高 宁(1991-),男,硕士生,主要研究方向为软件需求工程,E-mail:gaoninggn@foxmail.com;李 智(1969-),男,博士,教授,主要研究方向为软件需求工程、经验软件工程和软件测试,E-mail:zhili@gxnu.edu.cn(通信作者)。

2 问题框架方法

问题框架(Problem Framework,PF)[4-5]是由软件工程领域的著名学者 Michael A. Jackson 提出的一种软件开发方法。其认为软件系统对现实世界的作用是软件问题的来源,强调应该对软件系统将要作用的现实世界进行刻画,并且把需求的含义指称落实到现实世界相关领域的描述上。

问题框架是一种分解问题的方法。需求提供方给定一个软件开发需求,其将会被视为一个问题。问题框架在对问题进行分析时,将一个问题分解为若干个功能单一的子问题,使得关注点分离。通过问题的分解,可以逐个解决这些子问题,最后将它们组合起来,形成一个完整的系统解决方案。系统分析员在解决这些功能单一的子问题时,只需将关注点放在这个简单的子问题上,从而使得系统的可伸缩性得到提升。每一个子问题直接对应需求中的一个简单功能点,当需求发生变更时,只需修改简单的子问题,并且修改对应的解决方案或实现即可。

问题框架主要使用问题图来描述需求,问题图使用不同的符号和标注将某个软件开发问题划定为不同的结构,如描述计算行为的机器领域,描述与软件系统密切相关的外部环境和相关设备的问题领域,描述需求提出方对所要开发的项目所要达到某种要求的期望(称为需求),以及需求对某些领域的约束等。图 1 给出了一个简单的问题图示例。

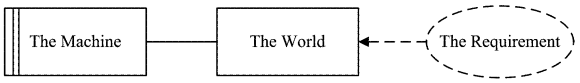

图 1 一个简单的问题图

从图 1 中可以看到,带有双竖线的矩形图案为机器领域,一个问题图中一般只有一个机器领域;不带竖线的矩形框是外部世界或相关设备的问题领域,用于描述现实世界;虚线的椭圆框代表需求,描述了需求提供方想要开发的软件;图中带箭头的虚线连接到领域上,表示需求的约束,规定了一些指定的关系或者约束。

3 行为驱动开发

行为驱动开发(Behavior Driven Development,BDD)[6]最初是由 Dan North 提出的一种以“设计”为核心价值的敏捷开发技术。它鼓励一个项目中的需求提供方、系统分析员、软件设计师、程序员、测试工程师一起协作,使用一种项目组成员都能理解的“通用语言”来描述系统的行为,从而避免由于表达不一致所带来的问题[7]。通用语言其实就是一些广为人知的词汇表,项目组的成员使用词汇表中的词汇来描述系统的功能以及行为。在行为驱动开发中,通常使用类似测试用例的用户场景文本来描述系统的行为,如图 2 所示。

Scenario:Successful widthrawal from an account in credit	
Given I have $100 in my account	#context
When I request $20	#event
When $20 should be dispensed	#outcome(s)

图 2 描述取钱的用户场景

图 2 描述了用户从信用卡中成功取钱的行为。给定的上下文说明了某用户信用卡中有 100 美元;接着,用户向系统请求取款 20 美元,取款的请求就是用户向系统发出的一个事件;之后,系统将 20 美元分配给用户,取款的分配操作就是系统对这个请求的响应,即输出。图 2 中描述系统行为时所采用的是 Gherkin 语言[8],Gherkin 语言的主要目标是设计一种人类可读的、文档化的、可执行的语言,它采用一种轻量级的结构来描述需求提出方想要开发的软件的行为。这种轻量级的语言结构使得项目组的成员能够轻易地理解系统的行为。

项目组成员可以将使用 Gherkin 语言描述的系统行为文本导入到开源工具 Specflow[8]中,并通过工具自动生成可执行的自动化测试框架代码,从而实现自动化测试。

4 问题框架与行为驱动开发相结合的软件开发方法(PFBDD)

为了从需求模型平滑过渡到软件设计以及实现,本文提出一种问题框架和行为驱动相结合的开发方法。

4.1 主要思想

需求在系统开发的各个阶段均扮演着至关重要的角色,然而需求分析阶段得出的各种需求建模图表只能够对软件设计和开发人员起到一定的指导作用。问题框架在获取软件需求的过程中,能够充分考虑现实世界相关的上下文及其对满足需求所起到的桥梁作用,并以此作为确定和解决需求问题的切入点。使用问题框架对需求进行建模时,主要的产物是描述需求的问题图。通过问题图可以让用户直观地体会到需求与各个领域之间在物理上或者逻辑上的联系。行为驱动开发主张测试先行,使用规定格式的可执行的用户场景来描绘系统的行为。

PFBDD 的主要思想是鼓励项目组成员参与需求建模,通过算法将建模期间得出的问题图转换成指定格式的可执行的用户场景文本,并通过工具自动生成测试框架代码。通过测试来推动软件的实现,从而保证软件开发的质量。开发步骤如下:

步骤 1 使用问题框架方法进行需求建模,并得到描述问题的问题图;

步骤 2 使用自动转换算法将问题图转换成指定格式的可执行的用户场景文本;

步骤 3 使用 Specflow 工具自动生成测试框架代码;

步骤 4 通过测试来推动软件功能的实现;

步骤 5 执行测试;

步骤 6 如果步骤 5 中的测试通过,则跳转到步骤 7,如果测试失败,则跳转到步骤 4;

步骤 7 一个问题流程结束,回到步骤 1,开始对新的问题进行建模。

在 PFBDD 的过程中,项目组成员通过头脑风暴对需求进行不断的完善,不断地迭代问题图,最终将得到与需求提供方的期望较为接近的模型。软件设计人员不必先考虑具体的类与方法,而是使用工具将模型转换成可执行的测试框架代码即可。开发人员为了让当前的测试通过,须实现相应的代码;实现代码之后再执行测试框架代码,以确认自己的实现是

否符合之前建立的模型所规定的行为。由于测试通过与否直接决定了软件的实现是否正确,因此通过测试的某一部分在以后的测试中出现问题的几率就会大大减小。这就不会像传统的软件开发那样,将各种各样的错误集中到最后的测试阶段来集中显示,因此能够改善代码的整体质量。由于测试框架代码是通过描述需求的用户场景而生成的,因此测试框架代码对需求具有可追溯性,从而保证了软件的实现对需求的可追溯性。

4.2　自动转换算法

本节主要介绍问题图到用户场景文本的自动转换算法。为了实现问题图到用户场景文本的自动转换,基于文献[9]提出的 3 套规则,实现了自动转换算法,算法描述如下:首先将直接连接需求的领域作为起始节点,若有多个领域连接需求,则需要人为指定起始节点;检索起始领域的属性,找到形如 A->B,B->C 的因果关系指示,即可知道起始节点的下一步;遍历所有的领域和机器,检索领域的因果关系属性;如果存在合理的因果关系,则遍历成功,然后记录领域中的文本以及下一步连线上的文本,从而可以串联成该问题图的流程描述信息文本,即用户场景文本;若某个领域不存在合理的因果关系,则结束遍历,流程结束。

算法的流程框图如图 3 所示。

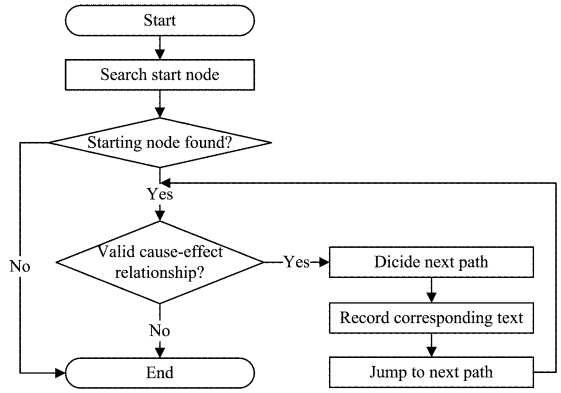

图 3　转换算法的流程图

5　案例研究

下面将通过车管中心排队系统的案例进行分析,展示如何使用 PFBDD 对软件进行需求分析以及指导软件开发与设计。

5.1　问题背景描述

随着社会的快速发展,汽车成为人们生活中不可或缺的一部分。私家车主不断增多,传统的排队方法已经成为车管所服务质量的瓶颈,阻碍了车辆管理的可持续发展。车管中心排队系统能够减少服务人员的工作量,提高服务质量。

图 4 为排队系统中的排队问题图,描述了车主携带本人身份证和相关必备材料到车管中心办理业务的流程。首先,车主需要将身份证和相关材料递交给前台服务人员 A(Front Desk Staff A)进行审核;审核通过后,前台服务人员 A 将车主的身份证置于二代身份证读卡器(ID Card Reader)上,读卡器将身份信息传递到排队终端(Queuing Terminal)中;排队终端向排队凭条打印机(Queuing Slip Printer)传递车主的排队信息;打印机向前台服务人员 B(Front Desk Staff B)打印出排队凭条;前台服务人员 B 把排队凭条交还给车主,车主等待叫号。

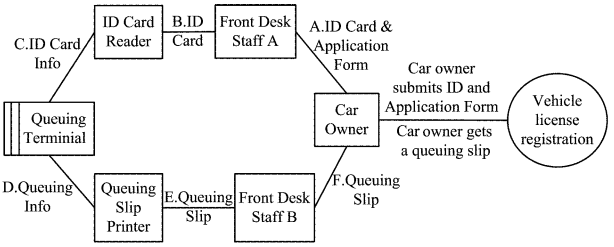

图 4　车管中心排队系统问题图

5.2　将问题图转换成用户场景文本

得到描述系统行为的问题图后,根据 4.2 节中描述的自动转换算法得到与该问题图一一对应的用户场景文本。用户场景文本使用 Gherkin 语言描述如下:

Scenario: Vehicle license registration

Given Car owner gives ID Card & Application Form to Front Desk Staff A

Given Front Desk Staff A gives ID Card to ID Card Reader

When ID Card Reader gives ID Card Info to Queuing Terminal

Then Queuing Terminal gives Queuing info to Queuing Slip Printer

Then Queuing Slip Printer gives Queuing Slip to Front Desk Staff AB

Then Front Desk Staff B gives Queuing Slip to Car Owner

其中,**Scenario** 关键字后的文本对应用户场景的名称;**Given** 与 **And** 这两个关键字对应用户场景所给出的上下文;**When** 关键字对应触发排队终端(Queuing Terminal)的一个事件,这里描述的事件是二代身份证读卡器将车主身份信息传递到排队终端,该事件触发排队终端做出响应;系统做出的响应即为输出,对应的是用户场景文本中的 **Then** 关键字,例如文本中出现的第一个 **Then** 关键字表示系统的输出,即排队终端将车主的排队信息传递到排队凭条打印机中。

5.3　使用工具生成自动化测试框架代码

在得到由问题图转换的用户场景文本后,可以使用一些 BDD 工具,例如 SpecFlow(C#),Jbehave(JAVA)和 Cuke4PHP(PHP)等,更多的工具可以参考文献[8]。

这里使用 SpecFlow 来生成可执行的 C# 自动化测试框架代码。SpecFlow 工具截图如图 5 所示。

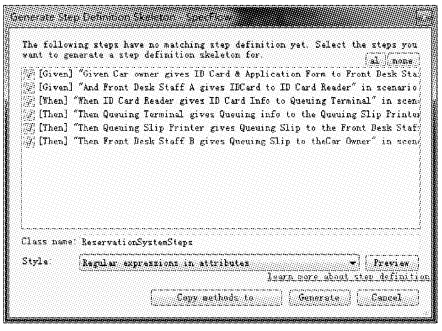

图 5　SpecFlow 工具截图

生成的部分代码如图 6 所示。

```
[Given(@"Front Desk Staff A gives ID Card to ID Card Reader")]
public void GivenFrontDeskStaffAGivesIDCardToIDCardReader()
{
    ScenarioContext.Current.Pending();
}
[When(@"ID Crad Reader gives ID Crad Info to Queuing Terminal")]
public void WhenIDCradReader_GivesIDCradInfoToQueuingTerminal()
{
    ScenarioContext.Current.Pending();
}
[Then(@"Queuing Terminal gives Queuing info to Queuing Slip Printer")]
public void ThenQueuingTerminalGivesQueuingInfoToQueuingSlipPrinter()
{
    ScenarioContext.Current.Pending();
}
```

图 6 SpecFlow 工具生成的部分代码

结束语 本文提出了一种基于问题框架的行为驱动开发方法(PFBDD 方法),为需求模型到软件设计与实现的过渡提供了一种实现方法。但是随着项目规模的扩大,可能存在需求冲突,例如某些共享现象或者物理实体间的冲突。如何检测、规避和解决这些冲突,以及存在冲突时如何指定子问题的优先级,我们将在下一步进行研究和探讨。

参 考 文 献

[1] LAVAZZA L,DEL BIANCO V. Combining problem frames and UML in the description of software requirements[C]// International Conference on Fundamental Approaches To Software Engineering. Springer-Verlag,2006:199-213.

[2] DE CARVALHO R A,MANHÃES R S. Filling the Gap between Business Process Modeling and Behavior Driven Development[J]. arXiv preprint arXiv:1005.4975,2010.

[3] DE CARVALHO R A,E SILVA F L C,Manhaes R S. Business language driven development:Joining business process models to automated tests[J]. Advances in Enterprise Information Systems II,2012(7):1-5.

[4] BLAINE J D. Problem Frames-Analyzing and structuring software development problems[J]. Software Quality Professional,2002(2):39-40.

[5] LI Z,PANG L,LIU G Y,et al. A Model-Driven Software Requirements Analysis Method and Its Technical Support[J]. Journal of Guangxi Normal University(Natural Science Edition),2013(2):19-26. (in Chinese)
李智,庞柳,刘国源,等.一种模型驱动的软件需求分析方法及技术支持[J].广西师范大学学报(自然科学版),2013(2):19-26.

[6] SOLIS C,WANG X F W. A Study of the Characteristics of Behaviour Driven Development [C]// 2011 37th EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA). 2011:383-387.

[7] CAO Y,CUI M. The Research on Behavior-driven Development of Automated Testing [J]. Journal of Qingyuan Polytechnic,2013,6(6):1-4. (in Chinese)
曹洋,崔萌.基于行为驱动开发的自动化测试方法研究[J].清远职业技术学院学报,2013,6(6):1-4.

[8] WYNNE M,HELLESOY A. The cucumber book:behaviour-driven development for testers and developers[M]. Pragmatic Bookshelf,2012.

[9] LIU G Y,WAN G H,PANG L,et al. Research and Development of Computer-aided Requirements Engineering Tool Based on Problem Frames [J]. Computer Science,2014,41(11):137-140. (in Chinese)
刘国源,万光海,庞柳,等.基于问题框架的计算机辅助需求工程工具的研发[J].计算机科学,2014,41(11):137-140.

[10] LI Z,JIN Z. From User Requirements to Software Specifications: An Approach Based on Problem Transformation [J]. Journal of Software,2013,24(5):961-976. (in Chinese)
李智,金芝.从用户需求到软件规约:一种问题变换的方法[J].软件学报,2013,24(5):961-976.

[11] LAZĂR I,MOTOGNA S,PARV B. Behaviour driven development of foundational UML components[J]. Electronic Notes in Theoretical Computer Science,2010,264(1):91-105.

[12] TIAN D,WEN J,LIU Y,et al. A Test-Driven Web application model based on layered approach[C]// 2010 IEEE International Conference on Information Theory and Information Security (ICITIS). IEEE,2010:160-163.

[13] SCHOENEMAN L,LIU J B. Integrating Behavior Driven Development and Programming by Contract[C]// Computational Science and Its Applications(ICCSA 2013). Springer Berlin Heidelberg,2013:590-606.

[14] SOEKEN M,WILLE R,DRECHSLER R. Assisted behavior driven development using natural language processing[M]// Objects, Models, Components, Patterns. Springer Berlin Heidelberg,2012:269-287.

[15] JANZEN D S,SAIEDIAN H. Does test-driven development really improve software design quality? [J]. Software, 2008, 25(2): 77-84.

[16] LANDandhÄUβER M,GENAID A. Connecting user stories and code for test development [C]// Proceedings of the Third International Workshop on Recommendation Systems for Software Engineering. IEEE Press,2012:33-37.