软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2013,24(4):675−695 [doi: 10.3724/SP.J.1001.2013.04275]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

一种“用例+控例”驱动的软件分析与设计方法*

刘 春[3], 张 伟[1,2], 赵海燕[1,2], 金 芝[1,2,3]

[1](高可信软件技术教育部重点实验室(北京大学),北京 100871)

[2](北京大学 信息科学与技术学院 软件研究所,北京 100871)

[3](中国科学院 数学与系统科学研究院,北京 100190)

通讯作者: 金芝, E-mail: zhijin@amss.ac.cn

摘 要: 随着软件本身及其运行环境的日益复杂,软件可信性引起人们越来越多的关注.软件的分析与设计也越来越需要综合考虑软件的功能性和可信性.然而,如何在软件的分析与设计过程中综合考虑软件的功能性和可信性,目前仍然缺乏系统而有效的方法.基于控制论,提出了一种基于“用例+控例”的方法,以在软件的分析与设计过程中综合考虑软件的功能性和可信性.在需求分析阶段,“用例+控例”模型支持需求工程师在同一个模型中自然地表达软件系统的功能性需求和可信性需求.在系统设计阶段,扩展了 ICONIX 开发方法的用例驱动的系统设计技术,用以分别识别实现用例的功能对象和实现控例的可信保障对象,以最终构建出既满足功能性需求又满足可信性需求的对象模型.应用一个实例来说明所提出方法的可行性.

关键词: 软件可信性;可信性需求;需求分析;系统设计

中图法分类号: TP311 **文献标识码**: A

中文引用格式: 刘春,张伟,赵海燕,金芝.一种“用例+控例”驱动的软件分析与设计方法法.软件学报,2013,24(4):675−695. http://www.jos.org.cn/1000-9825/4275.htm

英文引用格式: Liu C, Zhang W, Zhao HY, Jin Z. “Use Case+Control Case” driven approach for software analysis and design. Ruanjian Xuebao/Journal of Software, 2013,24(4):675−695 (in Chinese). http://www.jos.org.cn/1000-9825/4275.htm

“Use Case+Control Case” Driven Approach for Software Analysis and Design

LIU Chun[3], ZHANG Wei[1,2], ZHAO Hai-Yan[1,2], JIN Zhi[1,2,3]

[1](Key Laboratory of High Confidence Software Technologies of Ministry of Education (Peking University), Beijing 100871, China)

[2](Institute of Software, School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, China)

[3](Academy of Mathematics and Systems Science, The Chinese Academy of Sciences, Beijing 100190, China)

Corresponding author: JIN Zhi, E-mail: zhijin@amss.ac.cn

Abstract: As software systems and their operational environments become more and more complex, the topic of software dependability has attracted more and more attention. The software analysis and design are both required to focus on the functionality and the dependability. However, there still lacks an effective approach to consider the functionality and the dependability simultaneously during software analysis and design. Based on cybernetics, this paper proposes a “use case+control case” driven approach for software analysis and design, which aims to address the software functionality and dependability under a unified framework. During requirements analysis, the “use case+control case” model supports the co-modeling of the functional requirements and the dependability requirements. During system design, the design techniques proposed by ICONIX approach are extended to identify the functionality objects that realize the use cases, and the dependability objects that realize the control cases. A case is also used to illustrate the feasibility of the proposed approach.

Key words: software dependability; dependability requirement; requirements analysis; system design

* 基金项目: 国家自然科学基金(90818026); 国家重点基础研究发展计划(973)(2009CB320701, 2011CB302704)
收稿时间: 2012-01-16; 定稿时间: 2012-05-29

随着软件越来越多地应用于金融、航空、交通、电力等涉及国计民生的关键领域,一方面,软件的任何异常都有可能给用户带来无法容忍的利益损失;另一方面,软件本身及其运行环境却变得越来越复杂,软件在运行过程中也面临着越来越多的干扰(比如环境的变化、来自网络的恶意攻击、用户的操作失误等等).在这种情况下,如何构建一个既能够提供所需的服务,又具有安全性、适应性、可靠性等可信属性的软件系统,使其在面临干扰时仍能提供可信赖的服务,成为人们所关注的一个问题.

然而,如何在系统的分析和设计中综合考虑软件系统的功能性和可信性,目前仍然缺乏有效的方法.当前,成熟的软件分析与设计方法(比如用例驱动的面向对象方法 ICONIX[1,2]和 RUP[3]),大多主要关注于软件系统的功能性,可信性在软件系统的分析与设计过程中并没有得到充分的考虑.即使一些开发项目在开发过程中考虑了软件系统的可信性,但是这种考虑往往也是在软件系统设计或者实现之后才进行.这种滞后性也将使得设计人员在系统的设计过程中不能充分地权衡软件系统的可信性需求给系统的设计所带来的影响,并将付出较高的代价来对系统的设计进行修改.另一方面,虽然在需求工程领域已有一些方法(比如基于目标的方法[4,5]、误用例方法[6,7])提出在软件需求分析阶段对软件系统的可信性需求进行分析,但是这些方法往往只是从可信性的不同角度出发,关注于可信性的一个或者多个方面,仍然缺乏有效的框架以同时对软件系统的功能性和可信性进行分析.

为了实现在软件的分析和设计过程中综合考虑软件系统的功能性和可信性,本文基于当前用例驱动的软件分析与设计方法 ICONIX[1,2],提出了一种“用例+控例”驱动的方法.基于控制论,“用例+控例”驱动方法的基本思想是将一个可信的软件系统建模为一个前馈-反馈控制系统.在这样的系统中,软件系统通过前馈和反馈控制器来应对外在的威胁和自身行为可能出现的偏差,从而提高自身的可信性.基于该思想,在需求分析阶段,该方法在用例模型之上引入“用例+控例”模型,以对软件系统的功能性需求和可信性需求进行建模.在“用例+控例”模型中:用例用来建模软件系统的功能性需求,关注于如何从用户的角度出发描述软件系统的功能性行为;而控例用来建模软件系统的可信性需求,关注于如何从系统可能面临的威胁及其自身可能出现的行为偏差两个角度出发描述软件系统用以保障自身可信性的控制性行为.其中,前馈控例描述软件系统在应对一个威胁时应采取的行为,而反馈控例则描述当因为未知的不确定性导致系统运行时行为与设计时行为出现偏差时系统应采取的行为.在设计阶段,该方法扩展了 ICONIX[1,2]方法的系统设计技术,用以同时识别实现用例的功能对象和实现控例的可信保障对象,并最终构建出既满足功能性需求又满足可信性需求的对象模型.本文最后应用一个实例来说明该方法的可行性.

本文第 1 节介绍用例驱动的软件分析与设计方法 ICONIX 的主要过程.第 2 节概述本文所提出的“用例+控例”驱动的软件分析与设计方法.第 3 节和第 4 节分别从需求分析和系统设计两个方面来详细介绍“用例+控例”驱动的方法.第 5 节应用一个实例来说明“用例+控例”驱动方法的可行性.第 6 节介绍与本文相关的工作.最后是对本文工作的总结.

1 用例驱动的方法

ICONIX[1,2]是一种轻量级的用例驱动的开发方法,它力图在软件系统的分析与设计过程中使用 UML 的一个较小子集(4 种 UML 元素:类图、用例图、健壮性模型图、序列图)来对软件系统进行建模.图 1 显示了 ICONIX 方法的基本过程.

如图 1 所示,该方法的基本过程包含了 4 个活动:领域建模、用例建模、健壮性分析、交互建模,而该动态过程的目标是不断精化和完善实现用例的系统对象模型(class diagram).

- 领域建模的目标是根据对问题域的描述确定一个关于问题域的词汇表,这些词汇构成了系统对象模型的雏形.领域建模也是对系统的对象类(class)的初步猜想[1,2];
- 用例建模的目标是从用户的角度出发确定系统的用例模型;

领域建模和用例建模构成了 ICONIX 方法在需求分析阶段的主要活动.

健壮性分析和交互建模则构成了 ICONIX 方法在系统设计阶段的主要活动.

- 健壮性分析所扮演的是早期设计的角色,是需求分析和详细设计之间的桥梁.该活动的目标是识别实现用例的对象,并在此过程中精化用例的描述(这也是该活动被称为健壮性分析的主要原因).在该活动中,设计人员将用例描述的需求离散化为参与者、边界对象、实体对象、与控制对象的交互所构成的健壮性模型图(robustness diagram).其中,边界对象和实体对象一般对应于用例描述中的名词,边界对象是参与者用来与系统进行交互的对象,实体对象对应于系统的数据实体;而控制对象则一般对应于用例描述中的动词,负责协调边界对象与实体对象之间的交互,代表了用例所描述的系统行为;
- 健壮性分析之后的交互建模扮演的是详细的系统设计.该活动的目标是在对每个用例绘制序列图的过程中,将系统的行为(即健壮性模型图中控制对象所代表的行为)分配到健壮性分析过程所识别的边界对象和实体对象中去(也即识别对象应具有的方法).

基于 ICONIX 方法,软件系统的开发在经过如图 1 所示的分析与设计过程之后,就进入到代码实现阶段.

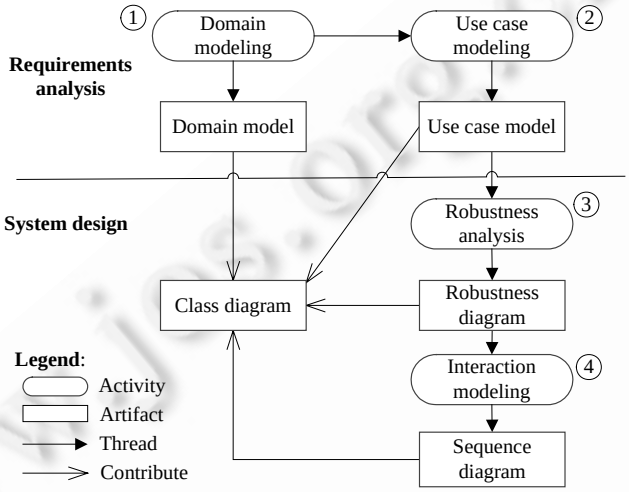


Fig.1 Analysis and design process of ICONIX

图 1 ICONIX 开发过程的分析与设计过程

2 “用例+控例”驱动的方法:概述

2.1 基本思想

一般来说,当我们要开发一个可信的软件系统时,首先需要考虑的是“可信性对当前要开发的软件系统意味着什么?”,也即“软件系统应该如何操作以使得自己可信?”.对于这一问题,本文采取的是一种逆向的方式,考虑“软件系统为什么不可信,是哪些潜在的因素影响了软件系统的可信性?”.基于这种方式,本文将一个可信的软件系统建模为一个前馈-反馈控制系统[8,9](其模型结构如图 2 所示),并将那些潜在的影响软件可信性的因素看作是软件系统在运行过程中可能面临的威胁.在这样的一个前馈-反馈控制系统中,被控对象是软件系统的核心系统,它对外展现软件系统的基本行为,提供用户期望的所有服务.前馈控制器实现一组防御性的控制措施来应对在需求分析时所识别的威胁.由于在分析阶段很难完全识别出软件系统在运行过程中可能面临的各种威胁,为了弥补前馈控制器的不足,反馈控制器则实现一组容错性的控制措施来响应系统本身在运行过程中可能出现的行为偏差.通过前馈和反馈控制器的作用,软件系统能够在一定程度上保障其行为符合预期、其提供的服务是可信赖的,从而提高自身的可信性.

基于该思想,软件系统的功能性需求将描述其核心系统的行为,而其可信性需求将描述前馈和反馈控制器所应具有的行为.在基于用例从用户的角度来建模软件系统的功能性需求时,本文提出前馈控例(feedforward control case, i.e., FFcontrol case)和反馈控例(feedback control case, i.e., FBcontrol case)两种控例来建模软件系

统的可信性需求.从这个意义上看,“用例+控例”模型提供了一种有效的框架来同时表达软件系统的功能性需求和可信性需求.

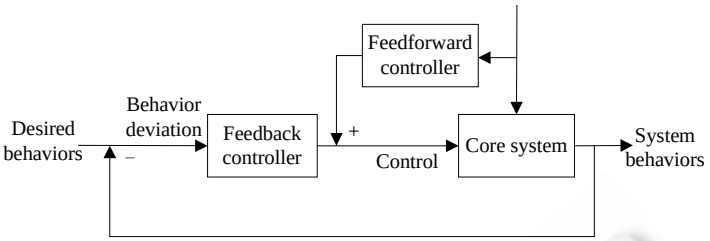


Fig.2 Feedforward-Feedback control model of the dependable software systems

图 2 可信软件系统的前馈-反馈控制模型

2.2 基本过程

“用例+控例”驱动的方法其主要思路是扩展如图 1 所示的用例驱动的过程,在需求分析过程中,应用用例和控例来分别建模软件系统的功能性需求和可信性需求;在设计过程中,综合考虑用例和控例所描述的系统需求以得到既能满足功能性需求又能满足可信性需求的设计模型.其具体过程如图 3 所示.

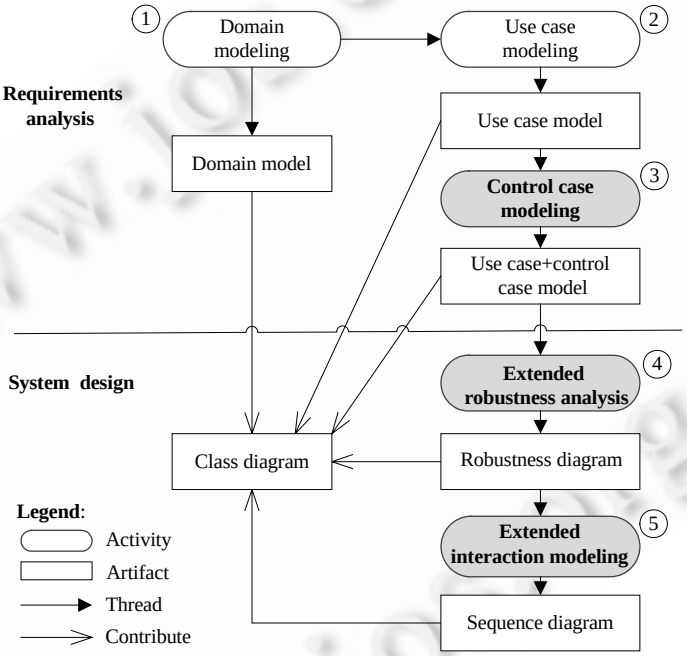


Fig.3 “Use case+Control case” driven analysis and design process

图 3 “用例+控例”驱动的分析与设计过程

该过程包含 5 个主要活动:领域建模、用例建模、控例建模、扩展的健壮性分析、扩展的交互建模.其中,

- 领域建模活动和用例建模活动与图 1 中的相应活动相同;
- 控例建模活动是新引入的活动,其目的是基于用例模型,建立系统的“用例+控例”模型;

从领域建模到控例建模的 3 个活动构成了“用例+控例”驱动的方法在需求分析阶段的主要活动,而扩展的健壮性分析和扩展的交互建模则构成了“用例+控例”驱动的方法在系统设计阶段的主要活动.

- 扩展的健壮性分析的目标是基于图 1 所示的健壮性分析活动,根据用例和控例的描述来设计健壮性模

型图,以识别实现用例的功能对象和实现控例的可信保障对象;

- 而扩展的交互建模的目标则是基于图 1 所示的交互建模活动,根据用例和控例的描述来绘制对象之间消息交互的序列图,以分配用例描述的系统功能性行为和控例描述的系统控制性行为到各个对象中,并考虑两种行为之间存在的关联.

下面,我们分别通过“用例+控例”驱动的需求分析和系统设计两部分内容,来详细介绍本文所提出的方法.

3 “用例+控例”驱动的需求分析

3.1 “用例+控例”模型

“用例+控例”模型是在用例模型中引入了控例,图 4 所示的元模型展示了“用例+控例”模型中各个概念以及概念之间的关系.

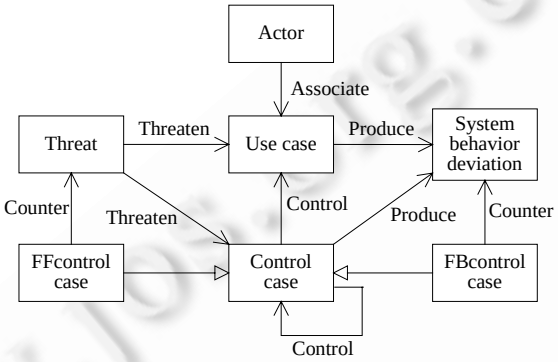


Fig.4 Meta model of the “use case+control case” model

图 4 “用例+控例”模型的元模型

控例描述用来控制用例或者其他控例的一个活动序列,该活动序列构成了软件系统为了保障自身的可信性所采取的控制措施.前馈控例描述软件系统在应对某个威胁时所采取的控制措施.威胁是系统内部或者外部所发生的、影响用例或者控例所描述的软件系统行为的干扰性事件,比如恶意的攻击、用户的操作错误、自然事件、系统内部组件的失效等.前馈控例所要应对的威胁一般是系统外部发生的威胁.

反馈控例描述软件系统为了应对自身的某个系统行为偏差所应采取的控制措施.系统行为偏差是指系统在运行时,其行为对设计时行为的偏离.这里的系统行为可以是用例所描述的功能性行为,也可以是控例所描述的控制性行为.常见的系统行为偏差有系统提供服务时间延迟、服务不可用、输出的数据值不正确或者精度不够等.

不同于用例,控例具有如下特征:

1) 控例描述的活动序列都是软件系统本身为了应对威胁或者系统行为偏差所采取的行为,因此一般不涉及用户;
2) 控例描述的活动序列并不是为了解决用户期望的问题,因此不返回用户需要的“值”.它的真正意义是应对可能的威胁或者系统行为偏差,以避免软件系统给用户带来新的问题.如果可能的威胁不存在或者系统行为偏差不会发生,那么控例就没有存在的意义.

以汽车驾驶为背景,图 5 展示了一个具体的“用例+控例”模型(其中,我们采用了不同的图形符号来显示“用例+控例”模型).如图 5 所示,由于一个控例也可以“控制”其他控例,因此“用例+控例”模型具有一定的层次性.其中,处于最内层的是用例,而之外的每一层都是由相应层次的控例所构成.

在“用例+控例”模型中,“控制”关系所蕴含的控制可以是直接的,也可以是间接的(即需要外部因素来协助完成控制).比如图 5 所示的降低最高限速(decrease the max speed limit)是一个直接的控制作用,而刹车失效报警

(brake failure alarm)则是一个间接的控制作用.此外,控例所描述的控制措施可以是静态的也可以是动态的.静态的控制措施指的是无论威胁或者行为偏差发生与否,系统都需要执行的控制措施,比如图 5 所示的锁车(lock car)、锁变速器(lock transmission)都是静态的应对威胁的控制措施.动态的控制措施则指的是当相应的威胁或者系统行为偏差发生时才需要执行的控制措施,比如图 5 所示的降低最高限速与刹车失效报警都是动态的控制措施.

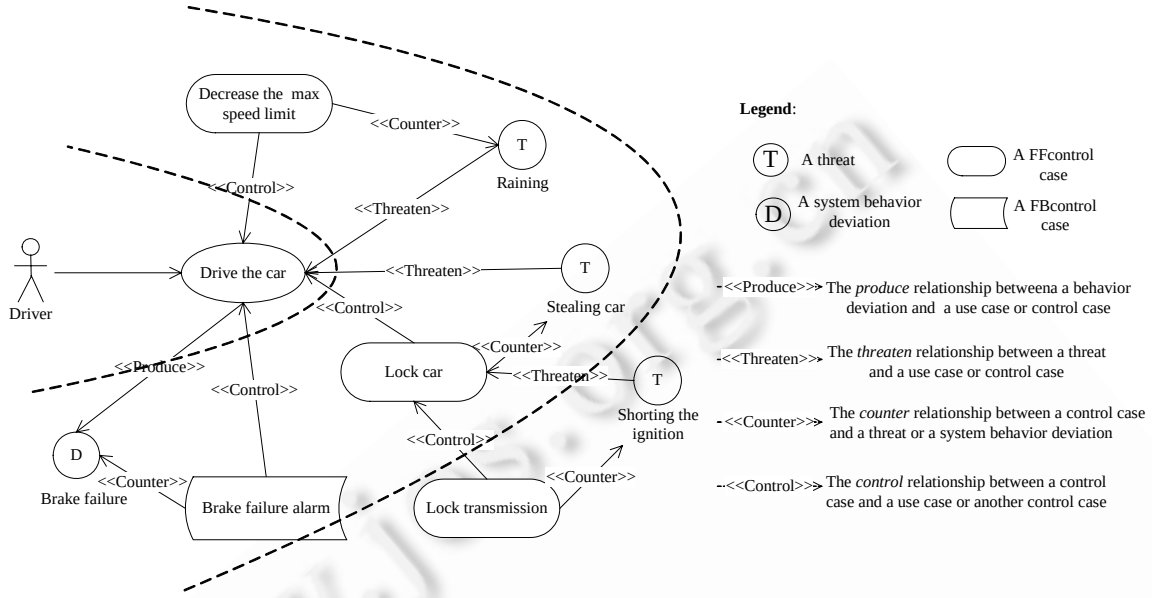


Fig.5 An example showing the “use case+control case” model

图 5 “用例+控例”模型的一个例子

在实际中,我们可以设计不同的控例文本模板来记录不同的信息.但是,一个前馈控例至少应该描述其需要应对的威胁和相应的控制措施,而一个反馈控例至少应该描述其需要响应的系统的行为偏差和相应的控制措施(本文所采取的控例模板如后文图 17 所示).

3.2 “用例+控例”驱动的需求分析过程

“用例+控例”驱动的需求分析首先从领域建模开始,然后分别进行用例建模和控例建模(如图 3 所示).领域建模和用例建模的具体过程在文献[1,2]中已给出具体的描述,这里主要介绍控例建模的过程.

控例建模过程包含 4 个活动(如图 6 所示):威胁与系统行为偏差的识别、风险评估、控制措施的确定、控例的定义.

- 威胁与系统行为偏差的识别以用例或者控例为背景,识别软件系统在运行时可能面临的威胁以及本身可能出现的行为偏差;
- 由于控制措施的增加必然引起成本的增加,因此在确定控制措施以应对所识别的威胁或者行为偏差之前,风险评估活动估计所识别的每个威胁或者系统行为偏差可能带来的风险;
- 控制措施的确定活动根据每个威胁和系统行为偏差的风险以及可选的控制措施的代价,权衡是否需要采取控制措施以及采取什么样的控制措施来应对所识别的威胁或者系统行为偏差;
- 当应对威胁或者系统行为偏差的控制措施确定之后,控例的定义根据控例模板来填写相应的信息以生成控例.

由于控例也可能面临新的威胁或者在执行过程中出现行为偏差,需求工程师在分析过程中需要迭代地执行上述控例建模过程,以识别生成的控例本身可能面临的威胁及可能出现的行为偏差,并确定相应的新的控例.

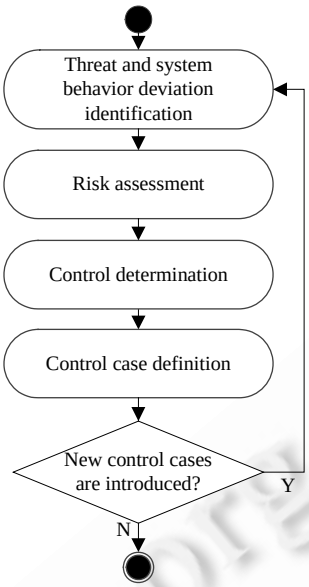


Fig.6 Control cases modeling process

图 6 控例建模过程

4 “用例+控例”驱动的系统设计

“用例+控例”驱动的系统设计主要包含两个活动:扩展的健壮性分析和扩展的交互建模(如图 3 所示).其中,前者主要关注于如何识别系统的对象,而后者则主要关注于如何识别对象应该具有的方法.下面分别介绍这两个活动.

4.1 扩展的健壮性分析

在 ICONIX[1,2]方法的健壮性分析活动中,设计人员逐句阅读用例的描述,并画出用例所涉及的参与者、边界对象、实体对象、控制对象以及它们之间的交互关系.对于每一个用例,设计人员执行上述过程,最终得到对应于每个用例的健壮性模型图.在扩展的健壮性分析过程中,设计人员不仅需要识别实现用例的边界对象、实体对象和控制对象,还需要识别实现控例的边界对象、实体对象和控制对象.本文将实现用例的对象和实现控例的对象分别称为功能对象和可信保障对象(如图 7 所示,本文采用不同的图形符号来区分功能对象和可信保障对象).

- 功能对象:该类对象实现用例所描述的功能性需求,其行为体现了系统需要支持的业务活动;
- 可信保障对象:该类对象实现控例所描述的可信性需求,其行为体现了系统所采取的旨在提高系统可信性的控制性行为.

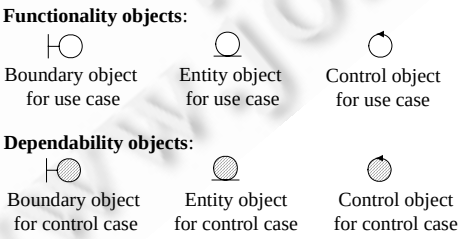


Fig.7 Symbols for the extended robustness diagram

图 7 扩展的健壮性模型图的图形符号

由于控例与用例,或者控例与控例之间存在着控制关系,相比于原有的健壮性分析过程,扩展的健壮性分析

除了需要考虑如何实现用例和控例以外,还需要考虑如何实现它们之间存在的控制关系.图 8 所示的算法给出了扩展的健壮性分析的过程.

Process: "**Use case+Control case" driven robustness analysis**
Input:
US: The set of use cases
CS: The set of control cases
Output:
Extended robustness diagrams
Begin:
1 for each use case *uc*∈*US*
2 identify the boundary object, entity object, and control object from *uc*'s description;
3 construct the robustness diagram for *uc*;
4 find the set of control cases CS_{uc} that directly or indirectly control the use case *uc* from *CS*;
5 for each control case *cc*∈CS_{uc}
6 identify boundary object, entity object, and control object from *cc*'s description;
7 if the controls described by *cc* are dynamic
8 construct an independent robustness diagram for *cc*;
9 else
10 if *cc* directly controls *uc*
11 add the objects identified from *cc* to the robustness diagram of *uc*;
12 else
13 add the objects identified from *cc* to the diagram of the control case that *cc* controls;
14 endif
15 endif
16 endfor
17 endfor
End.

Fig.8 Extended robustness analysis process

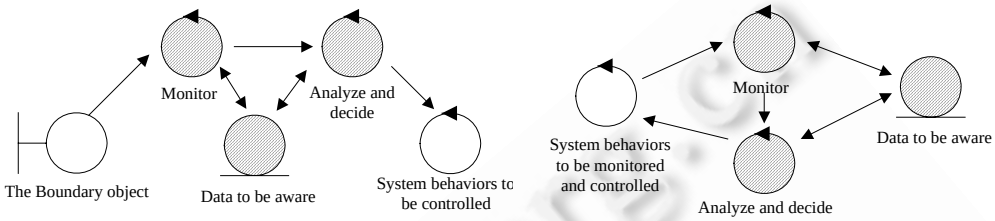
图 8 扩展的健壮性分析过程

首先,选定一个用例(第 1 行),识别实现该用例的参与者、边界、实体以及控制对象,并绘制该用例的健壮性模型图(第 2 行、第 3 行).然后,对当前选定的用例,识别直接或者间接作用于该用例的控例集合(第 4 行).对每一个控例,逐句阅读并理解该控例所描述的控制措施,识别该控例所涉及的边界、实体与控制对象,并确定哪些对象是该控例所控制的用例或者控例所涉及的功能对象,哪些是实现该控例的可信保障对象(第 5 行、第 6 行).这时,如果该控例描述的控制措施是动态的,则单独为该控例绘制一个健壮性模型图(第 7 行、第 8 行);否则:1) 如果该控例是直接作用于用例的,则将新识别的对象添加到用例的健壮性模型图中以扩展该健壮性模型图,实现控例和用例所描述的行为之间的融合(第 10 行、第 11 行);2) 如果该控例是间接作用于用例的(即该控例是控制于其他控例的),则将新识别的对象添加到其所控制的控例的健壮性模型图中(该健壮性模型图可能是已经扩展了的用例的健壮性模型图,也可能就是控例的健壮性模型图),实现不同控例所描述的控制措施之间的融合(第 12 行、第 13 行).

由于静态控制措施的执行过程往往是其控制的用例或者控例的执行场景的一部分,因此对于描述静态控制措施的控例,我们将实现该控例的对象和实现其所控制的用例或者控例的对象融合在一起,使得所确定的健壮性模型图中各个对象之间的交互不仅反映了用例的行为,也反映了控例的行为.而由于动态的控制措施需要动态的监控所控制的用例或者控例的行为、并根据一定的策略来决定是否执行,因此,动态的控制措施的执行过程是一个伴随着其所控制的用例或者控例的执行而存在的并行过程.在这种情况下,我们可以为描述动态控制措施的控例单独建立一个健壮性模型,以展示在运行时各个对象是如何交互以执行控例所描述的动态控制措施的.

描述动态控制措施的控例的健壮性模型图可以如图 9 所示.图 9(a)展示了描述动态控制措施的前馈控例的

健壮性模型图.它往往涉及一个或者多个边界对象,一个负责监视的控制对象,该控制对象从边界对象处获取一定的信息(这些信息一般会表征外在威胁),并将这些信息发给分析和决策控制对象,分析和决策控制对象最后施加一定的控制作用于其他的控制对象.其中,边界对象和所要控制的控制对象往往是实现被控制的用例或者控例的对象.而图 9(b)则展示了描述动态控制措施的反馈控例的健壮性模型图.不同于前馈控例的健壮性模型,它首先涉及一个控制对象,该控制对象是被控制的用例或者控例的健壮性模型中的对象.然后,一个监视控制对象负责感知被监控的控制对象的一些能够表征系统行为的信息,并将这些信息传递给分析和决策控制对象,分析和决策控制对象最后施加一定的控制作用于一个或者多个控制对象.这里,被控制的控制对象往往也是被监控的控制对象.



(a) 前馈控例的一般健壮性模型图 (b) 反馈控例的一般健壮性模型图

Fig.9 General robustness diagram of the control cases that describe the dynamic controls

图 9 描述动态控制措施的控例的一般健壮性模型图

在对描述动态控制措施的控例进行健壮性分析时,会存在某些对象存在于两个并行场景(也即两个不同的健壮性模型图)的情况,如图 9 所示.在这种情况下,设计人员需要考虑对象在两个场景中行为的同步性问题(对该问题的处理可以在接下来的交互建模过程中进行).

在扩展的健壮性分析活动的最后,设计人员根据健壮性分析的结果将新识别的对象,或者对象新的属性添加到对象模型中去.另一方面,健壮性分析的过程实际上也是对软件系统的需求重新进行确认和理解的过程.如同在 ICONIX 方法中,设计人员利用健壮性分析来精化用例的描述,在扩展的健壮性分析过程中,设计人员不仅可以精化用例所描述的功能性需求,还可以精化控例所描述的可信性需求,以确定控例描述的需求是否是恰当的、完整的,并且是可以通过一组对象来实现的.

4.2 扩展的交互建模

在得到实现用例和控例的对象之后,扩展的交互建模通过确定各个对象之间的交互序列,将用例描述的系统功能性行为和控例描述的保障系统可信性的控制性行为分配到各个对象中去.

在 ICONIX 方法的交互建模活动中,设计人员对每一个用例,首先将用例的内容复制到序列图的左侧,然后将健壮性模型图中的实体对象、边界对象以及参与者添加到序列图的顶部.在这之后,序列图分析的最重要的一步就是根据用例的文本描述,确定序列图顶部各个对象之间的交互关系.而这一步本质上是将健壮性模型中的控制对象所描述的行为解构为一组方法,并将它们分配到各个对象中去(实际上,有时一些控制对象也将作为系统的对象,添加到序列图的顶部).而扩展的交互建模活动不仅需要考虑实现用例的功能对象之间的交互,还需要考虑实现控例的可信保障对象之间的交互,以及这两类对象之间存在的交互.由于在扩展的健壮性分析过程中已经考虑到控例所描述的系统行为与用例所描述的系统行为之间的关联,因此扩展的交互建模活动可基于扩展的健壮性分析的结果,为每一个健壮性模型图建立序列图,其具体过程如图 10 所示.

针对扩展的健壮性分析所得到的每个健壮性模型图(第 1 行),首先,设计人员将健壮性模型所涉及的用例和控例的内容复制到序列图的左侧(第 2 行);然后,将健壮性模型中的参与者、实体对象和边界对象添加到序列图的顶部(第 3 行~第 5 行);最后,根据用例和控例的描述确定序列图顶部各个对象之间的交互,以将健壮性模型图中的控制对象的行为分配到序列图顶部的各个对象中去(第 6 行、第 7 行).在此过程中,设计人员可以将控例所描述的控制措施标注在其所影响的某个用例所描述的行为之后(如后文图 19 所示,利用“/”分隔用例和控例的

描述.此时,如果有多个处于同一层级的控例,则可以在控制措施之后标注控例的名字以区分不同控例所描述的不同控制措施.对于作用于控例的控例,我们可以利用多个"/"来表明它们之间的层次性,比如,可以利用"//"来表明第 2 层的控例与第 1 层的控例之间的关系),以便于设计人员检查序列图中对象之间的交互是否满足了用例或者控例所描述的需求.

Process: "**Use case+Control case" driven interaction modeling**
Input:
US: The set of use cases
CS: The set of control cases
RS: The set of extended robustness diagram
Output:
Extended sequence diagrams
Begin:
1 for each extended robustness diagram *ers*∈ *RS*
2 copy the involved use case and control cases of *ers* to the left margin of the sequence diagram;
3 add the involved actors in *ers* to the sequence diagram;
4 add the involved entity objects in *ers* to the sequence diagram;
5 add the involved boundary objects in *ers* to the sequence diagram;
6 for each control object *co* in *ers*
7 allocate the behaviors of *co* among the collaborating objects;
8 endfor
9 endfor
End.

Fig.10 Extended interaction modeling process
图 10 扩展的交互建模过程

序列图一般包含了对象(由包含了类名以及对象名称的矩形表示)、对象的生命线(由起始于对象的虚线表示)、消息交互(由生命线之间的箭头表示)等图形元素.为了实现需求到设计之间的可跟踪性,在扩展的交互建模过程中,我们扩展了序列图,以采用不同的图形符号(如图 11 所示)来分别表示实现用例的对象以及对象之间的消息交互、实现控例的对象以及对象之间的消息交互.在扩展的交互建模的最后,设计人员基于交互建模的结果,将所识别出的对象和对象方法添加到对象模型中去,以最终完善系统的对象模型.

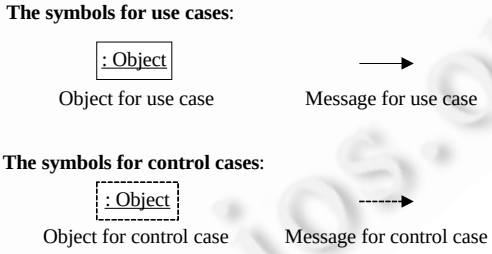


Fig.11 Symbols for the extended sequence diagram
图 11 扩展的序列图的图形符号

5 案例分析

本节以某金融机构的在线股票交易系统为例,来说明"用例+控例"驱动的分析与设计方法的应用.

5.1 在线股票交易系统简介

该案例分析的在线股票交易系统实现了一个电子股票经纪人的角色功能,其上下文如图 12 所示.在该场景

中,投资经理(portfolio manager)确定投资策略之后,指示股票交易员(trader)下单.订单提交到系统之后,交易系统检查订单的合法性,并将订单发送到相应的股票交易所(exchange)进行交易.在此过程中,系统从股票行情提供商(ticker feeder,比如路透社)那里获取实时的股票行情信息(tickers),并将这些信息提供给股票交易员和投资经理.当某个订单在股票交易所执行之后,股票交易所产生一个交易记录发送给交易系统,交易系统接收到该交易记录之后更新订单的执行状态.

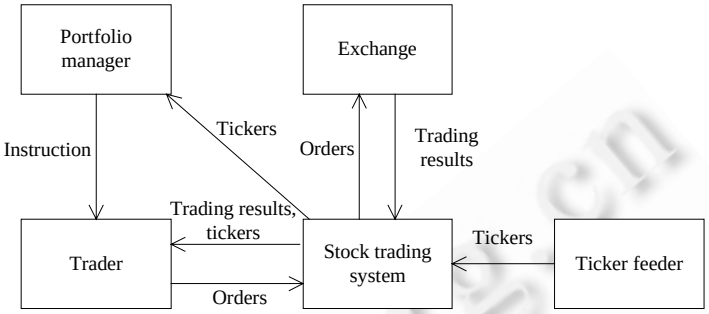


Fig.12 Context diagram of the online stock trading system

图 12 在线股票交易系统的上下文关系图

5.2 在线股票交易系统的需求分析

基于对股票交易过程的描述,初步确定的在线股票交易系统的领域模型如图 13 所示.

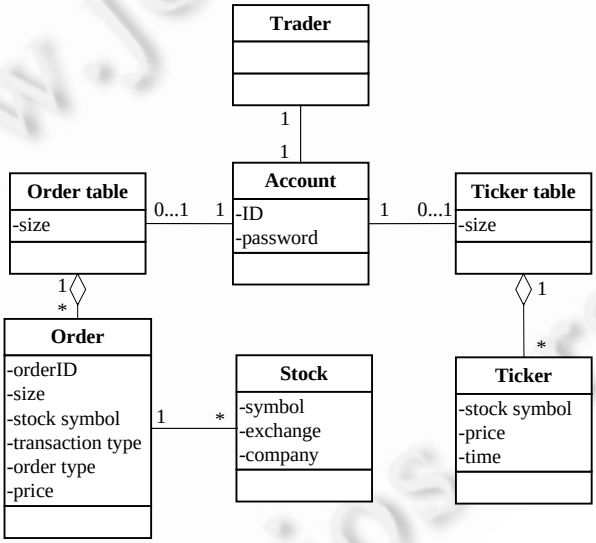


Fig.13 The domain model of the online stock trading system after preliminary analysis

图 13 初步确定的在线股票交易系统领域模型

基于如图 12 所示的上下文图,在线股票交易系统的参与者包括:投资经理、股票交易员、股票交易所、股票行情提供商.分别从这些参与者的角度出发,建立在线股票交易系统的用例,最终得到的在线股票交易系统的用例模型如图 14 所示.其中,用例“登录(log in)”和用例“提交订单(submit an order)”的文本描述如图 15 所示.

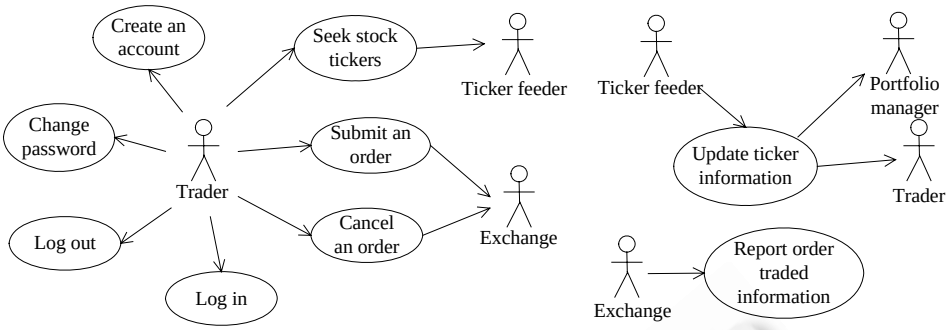


Fig.14 Use cases diagram of the online stock trading system

图 14 在线股票交易系统的用例图

Use case: **Log in**
Actor: **Trader**
Preconditions:
- The stock trading system is available.

Main flow:
1. The trader clicks the login button on the Home page;
2. The system displays the Login page;
3. The trader enters the account name and the password, and clicks the submit button;
4. The system validates the account information against the persistent account data and returns the trader to the Home page.

Postconditions:
- The trader has logged in the system.

Alternative flows:
4a. The account information is not right:
4a1. The system displays a message to inform the failure and prompts the trader to either re-enter the account information or click the create account button

Use Case : **Submit an order**
Actor: **Trader**
Preconditions:
- The exchanger which the order will route is connected and can accept instructions from system;
- The trader has logged in.

Main flow:
1. The trader clicks the submit order button on the Home page;
2. The system displays the order submission page;
3. The trader sets the basic information of the order: the stock symbol, the size, the type of the order in remote flag field, the price, and the type of the transaction(buy or sell);
4. The trader clicks the submit button to send the order to system;
5. The system checks the order if legal;
6. The system routes the order to the exchange where the stock lists for trading;
7. The system sends a submission success message to the trader.

Postconditions:
- The system has received an order from the trader;
- The system waits for the trading result of the order.

Alternative flows:
5a. The order is not legal.
5a1. The system asks the trader to reset the information of the order.
7a. The order’s submission fails.
7a1. The system returns the failure information to the trader.

Fig.15 Textual description of the use cases *log in* and *submit an order*

图 15 用例“登陆”和“提交订单”的文本描述

图 16 展示了我们基于用例“登录”和用例“提交订单”进行控例建模之后所得到的“用例+控例”模型(由于篇幅原因,这里没有列出完整的股票交易系统的“用例+控例”模型).其中,用例“登录”和用例“提交订单”的相关控例的文本描述如图 17 所示.

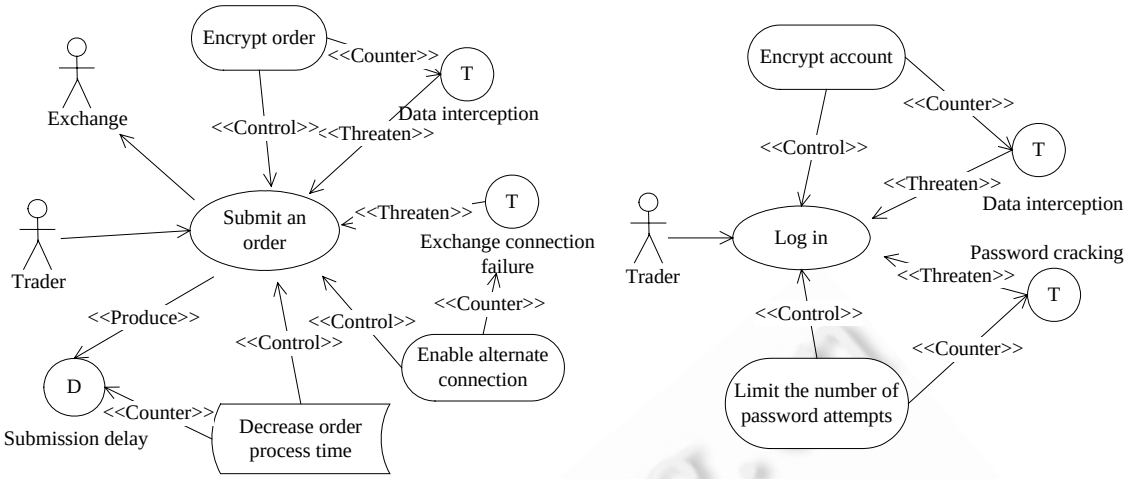


Fig.16 Part of the “use case+control case” diagram of the online stock trading system

图 16 在线股票交易系统的部分“用例+控例”图

FFcontrol case: **Encrypt account**
Stakeholder: Trader
Controlled use case: Log in
Threat model:
Threat name: Data interception
Threat description: After the trader enters the account information, the account information may be intercepted by some malicious persons through some sniffers. The malicious person may use the account information for some purpose undesired by the trader.
Controls:
After the trader enters the account information for login, the system needs to encrypt the account information before other actions.

FFcontrol case: **Limit the number of password attempts**
Stakeholder: Trader
Controlled use case: Log in
Threat model:
Threat name: Password cracking
Threat Description: Once some malicious persons know the account name of the trader, he will crack the account password by testing the password again and again with the help of some software tools.
Controls:
while the trader enters the account information, the system first needs to check the number that the trader has attempted, and then validate the account. If the password is right, then return the trader to the Home page. If the password is not right, the system needs to increase the number of the password attempts. If the number of attempts is bigger than three, the system displays the denying message on the Login page.

FFcontrol case: **Encrypt order**
Stakeholder: Trader
Controlled use case: Submit an order
Threat model:
Threat name: Data interception
Threat description: Someone may use some agents to intercept the order information that the trader has submitted. In that way, the malicious person may fake the information to destroy the system or cause losses to the trader.
Controls:
The system needs to encrypt the order after the trader has submitted it.

FFcontrol case: **Enable alternate connection**
Stakeholder: Trader
Controlled use case: Submit an order
Threat model:
Threat name: Exchange connection failure
Threat description: Because of the physical reasons, the connection between the system and each exchange may be not available. This will cause that the order can’t be routed to the exchange timely, and bring some losses.
Controls:
The system sends the “SYSTEM CHECK” message to the exchange in every 5mintures. If the connections are ok, the system will receive the same message from the exchange. If one connection is down, the system needs to alarm, and enable the alternate connection.

FBcontrol case: **Decrease order process time**
Stakeholder: Trader
Controlled use case: Submit an order
Behavior deviation model:
Deviation name: Response delay
Deviation Description: Because of some reasons, some orders may be blocked at some steps during the order processing in system. This may cause the submission delay of the orders. However, the system managers event don t know about this. Since routing the orders to exchange timely is very important for the stock trading, the submission delay may cause great losses to the investor.
Controls:
After the trader submits the order, the system needs to start to monitor the process time of the order. After the system has routed the order, the system ends the monitor. The system computes the average time of the order processing. If the average process time is bigger than expected, the system allocates more serves to deal with the orders. Otherwise, if there are no allocatable servers, the system should alarm to report the submission delay.

Fig.17 Textual description of the control cases of the use cases *log in* and *submit an order*

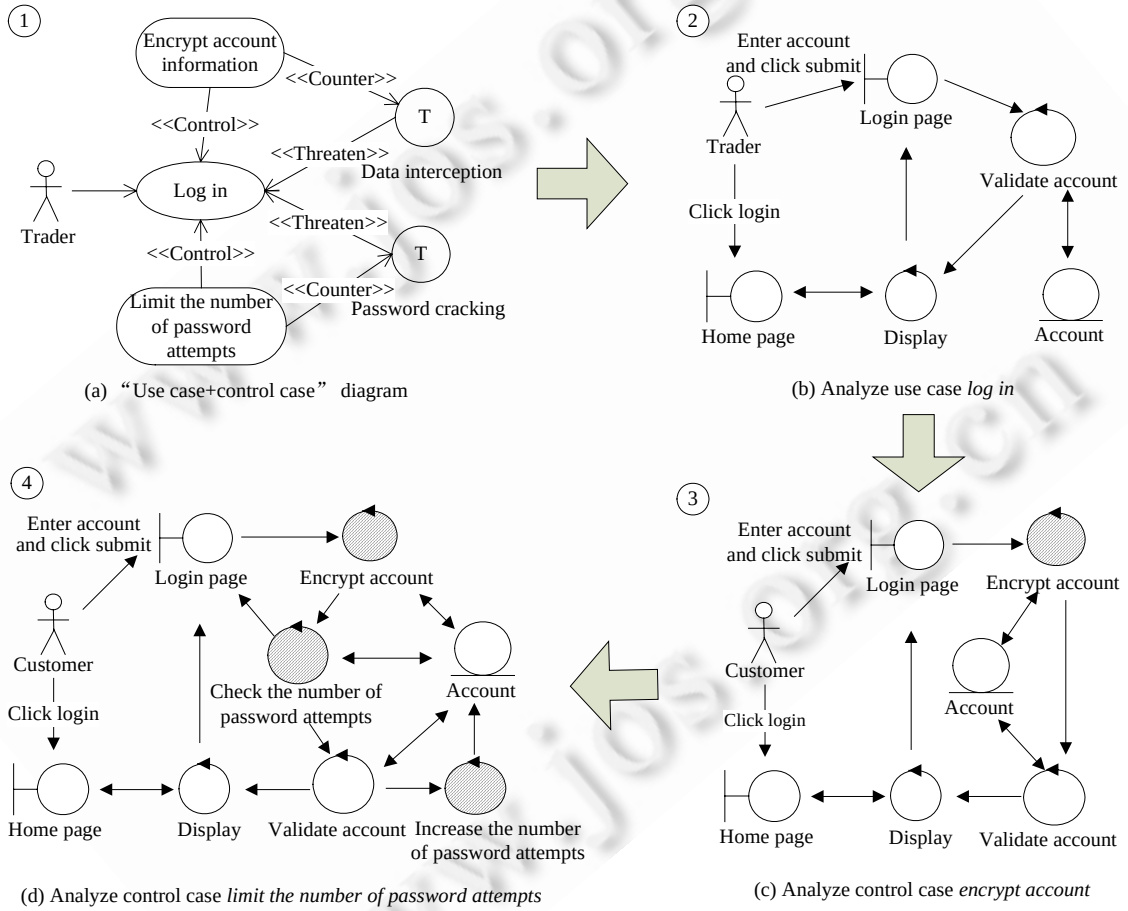
图 17 用例“登录”和“提交订单”的控例的文本描述

5.3 在线股票交易系统的设计

针对如图 16 所示的用例和控例,本节将以用例“登录”和“提交订单”及其相关的控例为例,进一步详细说明“用例+控例”驱动的在线股票交易系统的设计过程.

图 18 展示了对用例“登录”增加两个控例之后(“对账户加密(encrypt account)”、“限制密码尝试次数(limit the number of password attempts)”),其健壮性模型的更新过程:

- 首先,对用例进行健壮性分析,其健壮性模型如图 18(b)所示;
- 然后考虑控例“对账户加密”,由于该控例描述的控制措施是静态的,将该控例所引入的对象加入到用例的健壮性模型中.更新之后的健壮性模型如图 18(c)所示,其中,一个支持该控例的控制对象“对账户进行加密”被加入到用例的健壮性模型图中;
- 最后考虑控例“限制密码尝试的次数”.由图 17 所示的控例文本描述可知,该控例所描述的控制措施仍然是静态的,对该控例进行分析之后所更新的健壮性模型图如图 18(d)所示.从图 18(d)可以看出,此时又有两个新的控制对象被加入到用例的健壮性模型图中,它们是“检查密码尝试的次数(check the number of password attempts)”和“增加密码尝试的次数(increase the number of password attempts)”.



(a) “Use case+control case” diagram

(b) Analyze use case *log in*

(c) Analyze control case *encrypt account*

(d) Analyze control case *limit the number of password attempts*

Fig.18 “Use case+Control case” driven robustness analysis based on the use case *log in* and its control cases

图 18 针对用例“登录”及其控例的健壮性分析过程

基于图 18(d)所示的健壮性模型图,对用例“登录”及其控例“对账户加密”和“限制密码尝试的次数”进行交互建模的结果如图 19 所示.

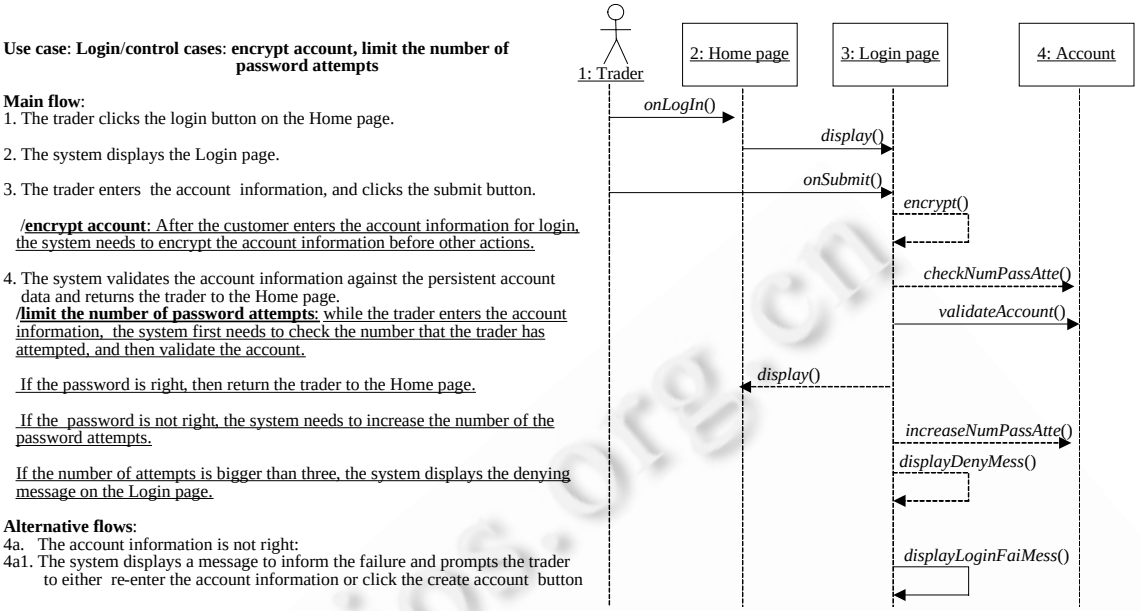


Fig.19 Sequence diagram for the use case *log in* and its control cases

图 19 用例“登录”及其控例的序列图

由图 17 所示的文本描述可知,用例“提交订单”的控例“对订单加密(encrypt order)”描述的是静态的控制措施,而控例“降低订单的处理时间(decrease order process time)”和控例“启用备用链接(enable alternate connection)”则描述的都是动态的控制措施,且三者均是直接控制于用例“提交订单”.对控例“对订单加密”进行健壮性分析,识别出支持该控例的控制对象“对订单加密”,把该控制对象添加到用例“提交订单”的健壮性模型图中所得到的健壮性模型如图 20 所示.对控例“降低订单的处理时间”和“启用备用链接”进行健壮性分析之后所得到的健壮性模型图如图 21 和图 22 所示.

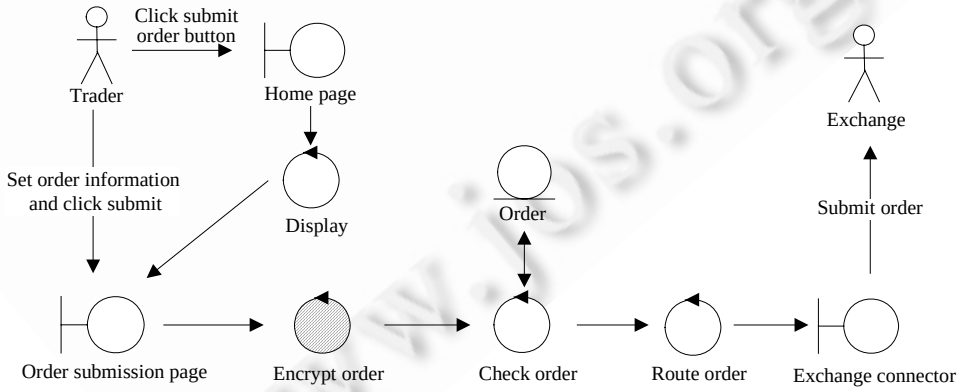


Fig.20 Robustness diagram of the use case *submit an order* and control case *encrypt order*

图 20 用例“提交订单”及控例“对订单加密”的健壮性模型图

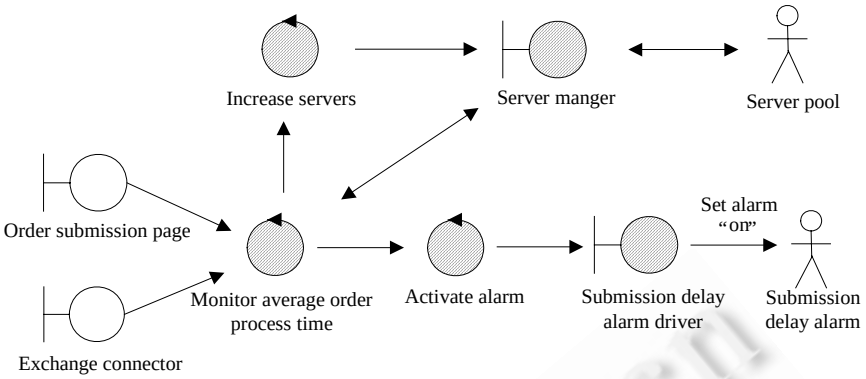


Fig.21 Robustness diagram of the control case *decrease order process time*

图 21 控例“降低订单处理时间”的健壮性模型图

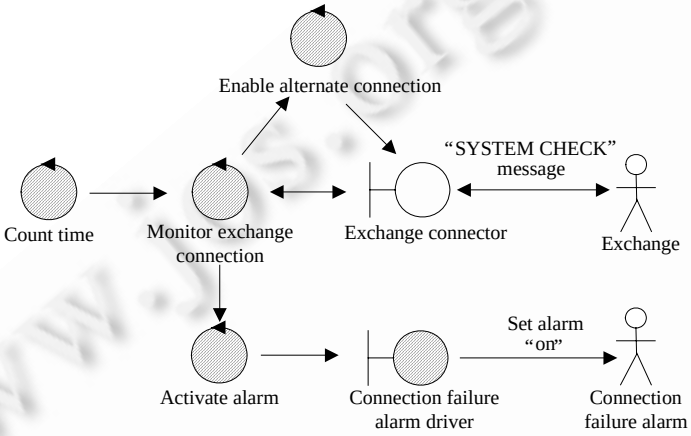


Fig.22 Robustness diagram of the control case *enable alternate connection*

图 22 控例“启用备用链接”的健壮性模型图

图 20~图 22 展示了 3 个相关的场景.由图 20 和图 21 可知,图 20 和图 21 所示的场景不仅共享某些对象,还存在着场景之间消息的传递.而图 22 则只是共享对象“交易所连接器”这一边界对象.

图 23 展示了基于图 20 的健壮性模型图所绘制的序列图.由于图 20 和图 21 的两个场景之间存在着消息传递,考虑到两个场景之间消息的同步问题,图 23 将图 21 健壮性模型图中的一个对象“监控订单的平均处理时间”添加到基于图 20 所示的健壮性模型图所绘制的序列图中.

图 24 展示了基于图 21 的健壮性模型图所绘制的控例“降低订单的处理时间”的序列图.

图 25 则展示了基于图 22 的健壮性模型图所绘制的控例“启用备用链接”的序列图.

如图 24 和图 25 所示,在进行交互建模时,我们直接将图 21 和图 22 的控制对象“监控订单的平均处理时间”和控制对象“监控交易所链接”作为了一个系统对象.

基于对用例“登录”和“提交订单”及其控例的健壮性分析和交互建模的结果,对领域模型(如图 13 所示)进行更新所得到的对象模型如图 26 所示(图 26 的类图采用了 ICONIX[1,2]开发方法所采用的风格).

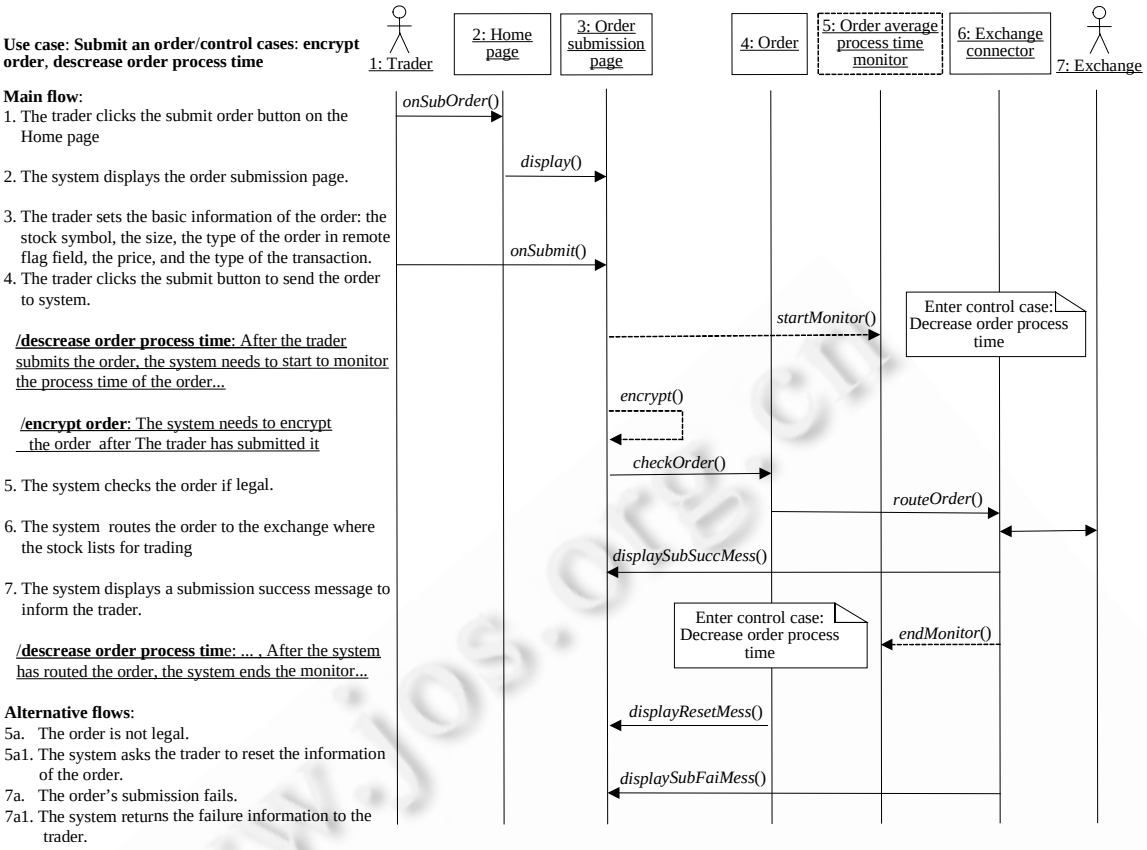


Fig.23 Sequence diagram for the use case *submit an order* and the control case *encrypt order*

图 23 用例“提交订单”和控例“对订单加密”的序列图

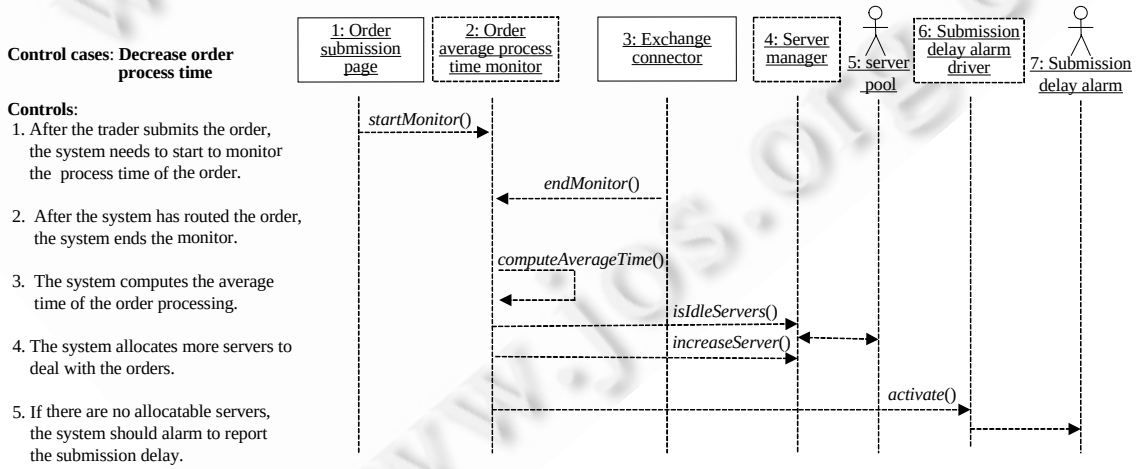


Fig.24 Sequence diagram for the control case *decrease order process time*

图 24 控例“降低订单的处理时间”的序列图

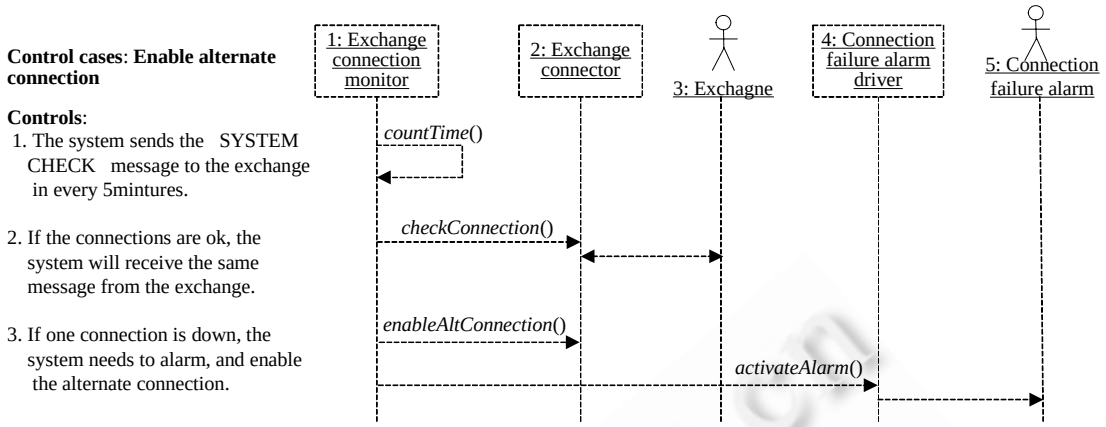


Fig.25 Sequence diagram for the control case *enable alternative connection*

图 25 控例“启用备用链接”的序列图

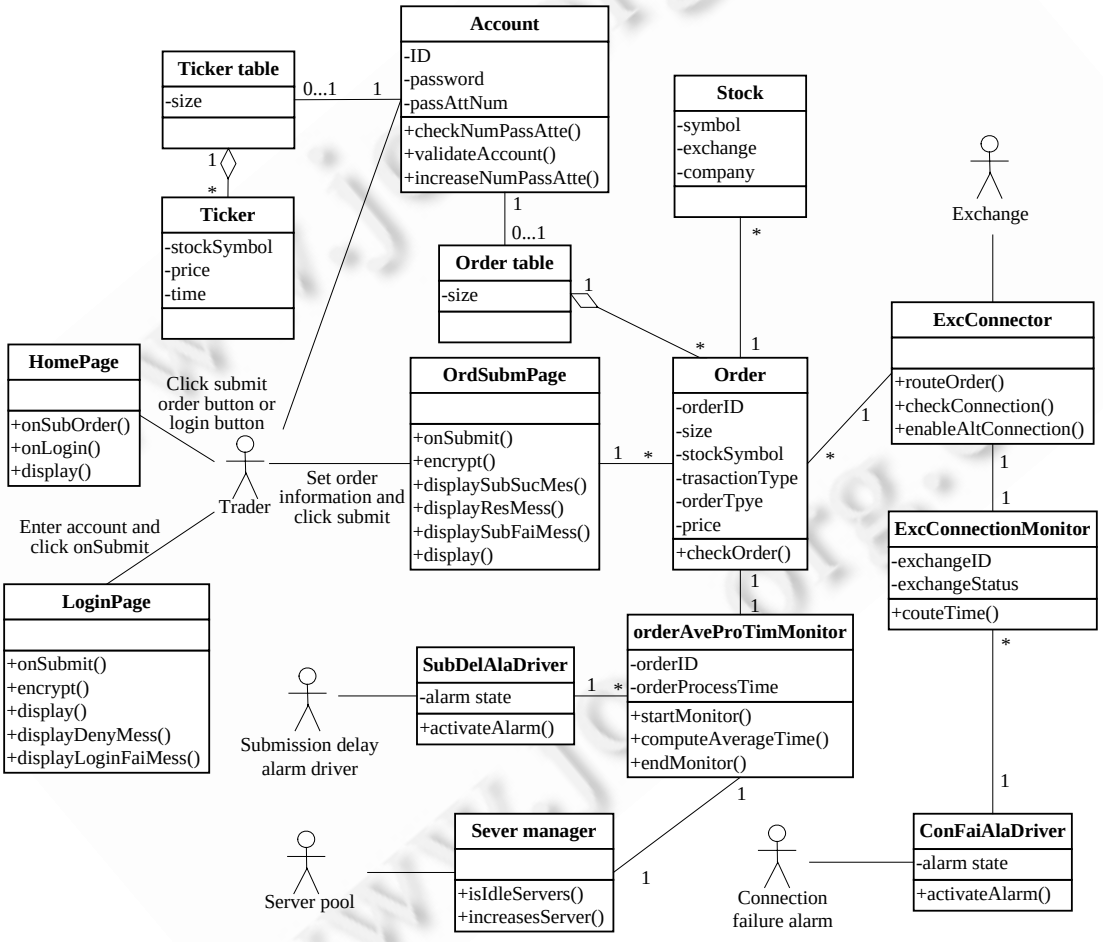


Fig.26 Class diagram updated after the analysis of the use cases *log in*, *submit an order* and their control cases

图 26 对用例“登录”、“提交订单”及其控例进行分析之后所得到的系统类图

6 相关工作

可信性(dependability)作为一个广为接受的概念由 Laprie 在 20 世纪 80 年代提出[10].Laprie 选择可信性作为一个集合性的概念来包含可靠性、可用性、安全性以及可维护性等属性特征,使得这些常见的属性特征成为人们从不同角度对系统可信性的不同认识.在此之后,许多研究工作[11,12]发展了可信性的定义,但是目前,对于可信性仍然缺乏一个统一的认识,比如国际上有些研究人员则利用另外两个单词:trustworthiness(trust)[13−15]和confidence[16,17]来表达可信性.

如何在软件系统的分析与设计中考虑软件系统的可信性,目前已经引起了很多研究人员的关注.基于用例方法,Sindre 等人提出了误用例方法[6]来捕获和建模软件系统的信息安全性(security)需求.在误用例方法中,误用例从误用者角度出发描述误用者是如何来攻击和破坏系统的,而用例(或者叫信息安全性用例,security use cases)则用来描述应对误用例的控制措施(也即信息安全性需求).在文献[7]中,Alexander 更进一步认为误用例方法不仅可以用来建模信息安全性需求,还可以建模其他方面的非功能性的需求.其中,误用例可以用来描述威胁以及其他因素对软件系统的影响,而用例则可以用来描述应对这些因素的策略.虽然误用例方法具有较好的建模软件系统的信息安全性需求的能力,但是由于仅仅从误用者角度出发,该方法并不能建模某些其他方面的可信性需求,比如由系统本身的缺陷所引起的可靠性需求.并且,由于用例是用来从用户的角度出发去描述用户与系统之间的交互场景,而应对威胁等因素的控制措施却往往是用户不可见的,也并不能描述成用户与系统之间的一个交互过程,因此,使用用例来描述应对威胁等因素的控制措施违背了用例本身的含义.更进一步地,在使用用例来描述软件系统功能性需求的同时,使用用例来描述应对威胁等因素的控制措施,也将使得软件系统的功能性需求关注点和可信性需求关注点混淆在一起,不利于关注点的分离.

同样,基于用例方法,在文献[18]中,Laibinis 等人在考虑容错性问题时,提出为每一个用例提供一个用来描述容错机制的用例以应对用例可能的失效.虽然该方法可以基于用例模型来建模软件系统的容错性需求,但是它仍然不能分离软件系统的功能性需求和可信性需求的不同关注点.

不同于上述基于用例的方法,本文在基于用例来表达软件系统的功能性需求的同时,提出控例来表达软件系统的可信性需求.“用例+控例”模型不仅可以支持需求工程师在需求分析过程中同时表达软件系统的功能性需求和可信性需求,还可以帮助需求工程师分离需求分析的不同关注点.其次,基于“用例+控例”模型,分析人员不仅能够分析外在各种威胁对软件系统的影响,还可以分析系统本身的行为偏差,使得在需求分析阶段不仅可以建模系统应对各种潜在威胁的需求,还可以建模系统应对各种潜在的系统行为偏差的需求.

基于目标的方法,Chung 等人[4]提出了软目标这一概念来捕获人们对安全性、可靠性、可用性等可信性属性也即非功能性属性方面的需求.从这些高层的抽象的软目标出发,需求工程师精化分解这些目标,直到推理出满足这些目标的领域假设和软件系统行为.这种方法提供了一种结构化的方法推理出软件系统为了满足用户的可信性需求所应具有的行为.但是,该方法并不能为用户和开发人员提供一种视图,让他们了解可信性需求与功能性需求之间的关系以及软件系统是如何与外部环境进行交互来满足这些可信性需求的.并且,目前虽然有一些研究者讨论如何将系统的非功能需求映射到系统的设计中去[19],但是对于如何将基于目标所表达的需求转化为系统的设计,仍然还缺乏成熟的方法.在本文所提出的“用例+控例”的方法中,虽然“用例+控例”模型支持功能性需求与可信性需求之间的关注点分离,但它能够建立软件系统的功能性需求和可信性需求之间的关系.“用例+控例”模型提供了一种视图,使得用户、需求工程师以及开发人员不仅可以了解软件系统的功能性需求和可信性需求,还可以了解这些需求之间所存在的关联.

将控制论引入到软件系统的分析与设计中,目前也引起了越来越多研究者的关注.在文献[20]中,Souza 等人基于反馈控制论提出了一类新的软件系统需求:感知需求(awareness requirement).感知需求关注于软件系统的某些需求满足与否.基于目标模型,Souza 等人认为,感知需求是附着在目标之上的元需求.在文献[21,22]中,研究人员则更进一步提出了基于控制论来设计适应性软件系统的构想.不同于这些研究工作,本文在当前成熟的用例驱动的分析与设计方法之上,应用控制论提出了“用例+控例”驱动的方法.这不仅可以与当前成熟的方法结合起来,还可以充分利用现有的成熟技术,比如在设计阶段的健壮性分析技术和对象交互建模技术.

在文献[23]中,Gran 等人从不同的可信性属性出发,提出了一种基于模型驱动的风险估计的方法来分析软件系统的可信性.为了统一不同的可信性属性以方便用户表达可信性需求,Donzelli 等人则在文献[24]中提出了UMD(unified model of dependability)模型.该模型包含了 5 个概念:问题、问题域(是系统级的还是某个局部的)、导致问题发生的事件、度量(度量表达用户对问题的容忍度)以及系统应对事件的响应.由于这些研究工作都仅关注于软件系统的可信性需求本身,因此它们并不能提供一种整体的框架以在软件系统的分析过程中同时表达软件系统的功能性需求和可信性需求.

7 结束语

为了实现在软件系统的分析与设计中综合考虑系统的功能性和可信性,本文基于控制论以及当前成熟的用例驱动的分析与设计方法,提出了一种“用例+控例”驱动的方法.该方法将一个可信的软件系统建模为一个前馈-反馈控制系统.在需求分析阶段,该方法利用“用例+控例”模型来对软件系统的功能性需求和可信性需求进行分析与建模.其中,用例被用来建模软件系统的功能性需求,控例则被用来建模软件系统的可信性需求.控例描述了软件系统应对各种威胁以及其本身在运行过程中可能出现的行为偏差所采取的各种控制措施.在设计阶段,该方法扩展了用例驱动的分析与设计方法 ICONIX[1,2]的系统设计技术以分别识别实现用例的系统功能对象和实现控例的系统可信保障对象,以构建出既满足系统的功能性需求又满足系统的可信性需求的对象模型.本文应用了一个在线股票交易系统来说明该方法的可行性.

未来的工作包括进行更多的案例研究以提高方法的可用性.特别是,如何采取一种有效的方式来组织控例的描述,不仅使得需求工程师更好地描述可信性需求,也使得控例所描述的可信性需求能够更好地转化为设计,是我们在未来需要进一步研究的问题.同时,支撑工具的开发也是未来需要进行研究的问题.

References:

[1] Rosenberg D, Scott K. Use Case Driven Object Modeling With UML: A Practical Approach. Addison-Wesley, 1999.

[2] Rosenberg D, Scott K. Applying Use Case Driven Object Modeling With UML: An Annotated E-Commerce Example. Addison-Wesley, 2001.

[3] Jacobson I, Booch G, Rumbaugh J. The Unified Software Development Process. Addison-Wesley, 1999.

[4] Chung L, Prado LJC. On non-functional requirements in software engineering. Lecture Notes in Computer Science, 2009,5600: 363−379. [doi: 10.1007/978-3-642-02463-4_19]

[5] Cysneiros LM, Do Prado Leite JCS. Nonfunctional requirements: From elicitation to conceptual models. IEEE Trans. on Software Engineering, 2004,30(5):328−350. [doi: 10.1109/TSE.2004.10]

[6] Sindre G, Opdahl AL. Eliciting security requirements with misuse cases. Requirements Engineering, 2005,10(1):34−44. [doi: 10.1007/s00766-004-0194-4]

[7] Alexander I. Misuse cases: Use cases with hostile intent. IEEE Software, 2003,20(1):58−66. [doi: 10.1109/MS.2003.1159030]

[8] Goodwin GC, Graebe SF, Salgado ME. Control System Design. Prentice Hall, 2000.

[9] Cai K, Cangussu JW, Decarlo RA, Mathur AP. An overview of software cybernetics. In: Proc. of the 11th Annual Int’l Workshop on Software Technology and Engineering Practices. Amsterdam: IEEE, 2003. 77−86. [doi: 10.1109/STEP.2003.4]

[10] Laprie JC. Dependability computing and fault tolerance: Concepts and terminology. In: Proc. of the 15th Annual Int’l Symp. on Fault-Tolerant Computing. Munich, 1985. 2−11. http://www-users.cselabs.umn.edu/classes/Fall-2010/seng5861/Laprie-Definitions.pdf

[11] Jackson D. A direct path to dependable software. Communications of the ACM, 2009,52(4):78−88. [doi: 10.1145/1498765.1498787]

[12] Avizienis A, Laprie JC, Randell B, Landwehr C. Basic concepts and taxonomy of dependable and secure computing. IEEE Trans. on Dependable and Security Computing, 2004,1(1):11−33. [doi: 10.1109/TDSC.2004.2]

[13] Schneider FB. Trust in Cyberspace. Washington: National Academy Press, 1999.

[14] Craig M, Pierre de V, Peter H, Matt C. Trustworthy computing. Microsoft White Paper. Microsoft Corporation, 2002. http://www.microsoft.com/china/mscorp/twc/twc_whitepaper.mspx

[15] Lin C, Peng XH. Research on trustworthy networks. Chinese Journal of Computers, 2005,28(5):751−758 (in Chinese with English abstract).

[16] High Confidence Software and Systems Coordinating Group. High confidence software and systems research needs. 2001. http://www.ccicgov/pubs/hcss-research.pdf

[17] Chen HW, Wang J, Dong W. High confidence software engineering technologies. Acta Electronica Sinica, 2003,31(12):1933−1938 (in Chinese with English abstract).

[18] Laibinis L, Troubitsyna E. Fault tolerance in use-case modeling. In: Proc. of the 4th Int’l Workshop on Requirements for High Assurance Systems. Paris, 2005. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.84.4950&rep=rep1&type=pdf

[19] Xu L, Ziv H, Alspaugh TA. An architectural pattern for non-functional dependability requirements. The Journal of Systems and Software, 2006,79(10):1370−1378. [doi: 10.1016/j.jss.2006.02.061]

[20] Souza V, Lapouchnian A, Robinson WN, Mylopoulos J. Awareness requirements for adaptive systems. In: Cheng BHC, ed. Proc. of the 2011 ICSE Symp. on Software Engineering for Adaptive and Self-Managing Systems. Waikiki, 2011. 60−69. [doi: 10.1145/1988008.1988018]

[21] Souza V, Mylopoulos J. From awareness requirements to adaptive systems: A control-theoretic approach. In: Proc. of the 2nd Int’l Workshop on Requirements@Run.Time. Trento: IEEE, 2011. [doi: 10.1109/ReRunTime.2011.6046242]

[22] Filieri A, Ghezzi C, Leva A, Maggio M. Self-Adaptive software meets control theory: A preliminary approach supporting reliability requirements. In: Alexander P, Pasareanu CS, Hosking JG, eds. Proc. of the 26th IEEE/ACM Int’l Conf. on Automated Software Engineering. Lawrence: IEEE, 2011. 283−292. [doi: 10.1109/ASE.2011.6100064]

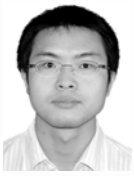
[23] Gran BA, Fredriksen R, Thunem APJ. Addressing dependability by applying an approach for model-based risk assessment. Reliability Engineering and System Safety, 2007,92(11):1492−1502. [doi: 10.1016/j.ress.2006.10.002]

[24] Donzelli P, Basili V. A practical framework for eliciting and modeling system dependability requirements: Experience from the NASA high dependability computing project. The Journal of System and Software, 2006,79(1):107−119. [doi: 10.1016/j.jss.2005.03.011]

附中文参考文献:

[15] 林闯,彭雪海.可信网络研究.计算机学报,2005,28(5):751−758.

[17] 陈火旺,王戟,董威.高可信软件工程技术.电子学报,2003,31(12):1933−1938.



刘春(1982－),男,河南信阳人,博士生,主要研究领域为需求工程.
E-mail: liuchun@amss.ac.cn

赵海燕(1966－),女,博士,副教授,CCF 高级会员,主要研究领域为软件工程,领域工程.
E-mail: zhhy@sei.pku.edu.cn

张伟(1978－),男,博士,副教授,主要研究领域为需求工程,软件复用.
E-mail: zhangw@sei.pku.edu.cn

金芝(1962－),女,博士,教授,博士生导师,CCF 高级会员,主要研究领域为需求工程,知识工程,基于知识的软件工程.
E-mail: zhijin@amss.ac.cn