一种建模软件体系结构非功能属性的方法

张琳琳[1,2] 应 时[1,3] 赵 楷[1,2] 文 静[1] 倪友聪[1,4]
(武汉大学软件工程国家重点实验室 武汉 430072)[1] (新疆大学信息科学与工程学院 乌鲁木齐 830046)[2]
(武汉大学计算机学院 武汉 430072)[3] (安徽建筑工业学院数理系 合肥 230018)[4]

摘 要 非功能属性的处理一直是困扰着研究人员和实践者的难题。针对体系结构设计阶段早期,提出一种建模非功能属性的方法。该方法利用面向方面软件开发中的关注点多维分离原理,提出建模软件体系结构的"1+X"模型,并在此基础上进一步划分体系结构非功能属性的维度,归纳各个维度上的关注点,利用XML对维度和关注点进行规约。该模型为后续设计面向方面的软件体系结构奠定了基础,为体系结构设计人员提供了方法支持,而且该成果可以直接用于不同领域内的软件体系结构设计。

关键词 面向方面软件体系结构,非功能属性,关注点多维分离
中图法分类号 TP311.5 **文献标识码** A

Method of Modeling Non-functional Properties in Software Architecture

ZHANG Lin-lin[1,2] YING Shi[1,3] ZHAO Kai[1,2] WEN Jing[1] NI You-cong[1,4]
(State Key Lab of Software Engineering, Wuhan University, Wuhan 430072, China)[1]
(School of Information Science and Engineering, Xinjiang University, Urumqi 830046, China)[2]
(College of Computer Science, Wuhan University, Wuhan 430072, China)[3]
(Department of Mathematics and Physics, Anhui Institute of Architecture & Industry, Hefei 230018, China)[4]

Abstract How to address non-functional properties in software system has afflicted various stakeholders for a long time, and been one of the key points in software engineering fields. Aiming at the early stage of software architecture design, this paper proposed a new method for modeling non-functional properties, which employs the principle of multi-dimensional separation of concerns(MDSoC), and proposed a model named "1+X" for handling. Based on this model, multiple dimensions of non-functional properties were classified, as well as concerns of non-functional properties for each dimension. Finally, both non-functional properties dimensions and concerns were specified using XML. Research works in this paper can be prepared for the aspect-oriented software architecture design, for the concerns of non-functional properties handled can be directly encapsulated using aspectual components. In addition, this method provided supports for architects, and the outputs of this method can be directly used in the software architecture design related to the various domains.

Keywords Aspect-oriented software architecture, Non-functional properties, Multi-dimensional separation of concerns (MDSoC)

随着软件密集化程度越来越高,软件规模越来越大,复杂度日益增强,软件体系结构设计在软件系统开发生命周期中变得越来越重要。在设计软件体系结构时,不仅要考虑满足当前环境的需要,还要考虑系统随时间的演化适应新环境、新技术的要求,以及体系结构被重用时的要求。因此,软件体系结构设计仅覆盖系统的全部功能是远远不够的,更为重要的是软件体系结构要满足其在设计、演化和重用过程中的非功能属性。

传统的体系结构设计方法由于缺乏系统的建模非功能属性的手段,造成非功能属性散列(scatter)和混杂(tangle)在组件和连接件中的混乱局面,致使体系结构设计方案难以理解、演化和重用。本文着眼于软件体系结构设计早期,提出一种建模体系结构非功能属性的方法。该方法建立在对软件体系结构非功能属性长期观察的基础之上,利用面向方面软件开发中的多维关注点分离原理,提出描述软件体系结构的"1+X"模型。该模型归纳和总结了软件体系结构设计、演化和重用过程中涉及的非功能属性,将它们进一步划分为若干个维度,每个维度包含若干个细化的子属性,并借助XML对非功能属性进行了规约。本文的方法通过"1+X"模型,建立软件体系结构设计过程中通用的非功能属性库,并给出库中主要

到稿日期:2009-01-06 返修日期:2009-02-21 本文受国家高技术研究发展计划863项目(2006AA01Z168),国家自然科学基金资助项目(60773006),高等学校博士学科点专项科研基金资助项目(20060486045)资助。
张琳琳(1974-),女,博士生,讲师,主要研究方向为体系结构、面向方面软件开发,E-mail:zllnadasha@yahoo.com.cn;应 时(1965-),男,博士,教授,博士生导师,主要研究方向为面向对象软件工程方法、基于组件的软件工程方法、软件体系结构和模式、软件的可重用性与互操作性等。

构成成分的规约形式。非功能属性库的建立，将会为软件体系结构的设计人员提供极大的参考价值。

本文首先介绍了在软件体系结构中建模非功能属性的必要性，进而提出描述软件体系结构的“1＋X”模型；第 2 节重点介绍了该模型，给出了本文的非功能属性库及相应的规约形式；第 3 节结合网上拍卖系统的部分内容，说明了运用本文方法的过程；第 4 节介绍了相关工作；最后总结了全文并指出下一步的研究工作。

1 软件体系结构的非功能属性

软件体系结构的非功能属性覆盖了软件系统中未被功能属性描述的特征，涉及的内容十分丰富。首先，软件体系结构的非功能属性涉及诸多涉众(stakeholders)，如软件体系结构设计人员、维护人员、质量控制与保障人员、软件体系结构的重用者等，每种人员对体系结构的要求和侧重有所不同。其次，体系结构的非功能属性覆盖问题空间的多个方面，如软件体系结构的设计要求、技术性约束条件、质量属性等，它们从不同角度制约着体系结构设计方案的形成。最后，软件结构的非功能属性总是与系统所期望的体系结构达到的目标、施加在体系结构设计方案上的约束、体系结构所满足的条件密切相关。

此外，软件体系结构的非功能属性之间的关系比较复杂，表现在：(1)非功能属性之间相互重叠，如效率和性能都涉及响应时间和吞吐量两个指标；(2)非功能属性之间相互制约，如安全性和性能之间，提高系统的安全性需要考虑诸如加密算法及身份验证在内的技术，运用这些技术势必会延缓系统处理事务的响应时间。

另外，传统体系结构设计方法大都采取二维的方式来处理体系结构的功能和非功能属性，即体系结构设计人员总是先设计出满足功能属性的解决方案，然后在该方案基础上考虑非功能属性的设计问题。由于缺乏系统地处理非功能属性的手段，致使所设计出的体系结构解决方案中存在着大量的横切现象，这些现象导致了组件、连接件之间的紧密耦合，模糊了组件与连接件之间的边界，致使最终的体系结构难以理解、演化和重用。因此，为了获得高质量免横切的软件体系结构，很有必要在体系结构设计早期对非功能属性进行有效建模。具体而言，就是要从多个不同维度对它们进行分析，独立地标识乃至详细地描述，以便在软件开发生命周期中的其它阶段(如维护阶段等)对它们进行跟踪。下文为了描述方便，除做特别说明的地方之外，提到的非功能属性均在软件体系结构设计阶段。

2 软件体系结构的“1＋X”模型

为了描述非功能属性，我们借助面向方面软件开发中的多维关注点分离原理，把非功能属性分为多个维度，联合功能属性，提出了建模软件体系结构的关注点多维分离模型——“1＋X”模型。除了直观地表示了软件体系结构层的功能属性和非功能属性，还提供了通用于软体体系结构设计的非功能属性库以及非功能属性的规约描述方法。

2.1 “1＋X”模型

建模体系结构的“1＋X”模型，如图 1 所示。“1＋X”模型把体系结构需求空间(图 1 中大圆)划分成 $1+X$ 个维度，其中“1”表示一个功能维度，对应于所有的功能属性，称为主维或主要维度。“X”表示 X 个非功能维度，每一个非功能维度对应一个非功能属性，称之为一个次维或次要维度，如图 1 中的 $sd_1, sd_2, \cdots, sd_x$。一个次维可被进一步精化为包含一个或多个次维关注点的集合，如图 1 中 sd_1 精化为 sc_{11} 和 sc_{12} 两个次维关注点的集合。为体现次维关注点的横切特征，特地用菱形表示它们。精化后的次维关注点(即前文所述的子属性)之间存在一定的关系，彼此之间相互依赖，在图 1 中表示为虚线连接。值得说明的是，该模型中的“1”不仅表示仅有的一个主维，而且还表示该维上只有一个关注点。这一处理有利于在主维的基础上考察多个次维之间的关系，如冲突、贡献等(见 2.3.1 节)。

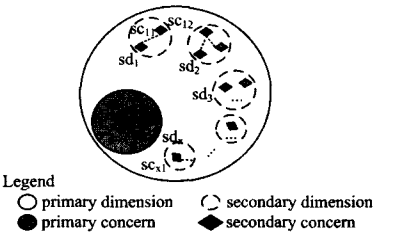

图 1 “1＋X”模型

2.2 非功能属性库

“1＋X”模型的另一个用途是为体系结构设计人员提供非功能属性库，该库尽可能全面地覆盖软件体系结构设计、演化和重用过程中应考虑到的非功能属性。在设计一个具体系统的软件体系结构时，设计人员可以根据我们的非功能属性库选取适合于待建系统的非功能属性。

非功能属性库建立在 Rational 公司 FURPS＋框架[6]的基础上，并对其进行了扩展，目前包括 12 个次要维度(即“1＋X”模型中的 X 为 12)，涉及 60 多个非功能子属性。与 Chung 等人提出非功能需求列表[1]的工作相似，本文在表 1 中列出 12 个次要维度及其所包含的关注点，也就是 12 个非功能属性以及它们的子属性。其中，$id1-5$ 是最常见也是最重要的 5 个非功能属性，$id6-7$ 与特定领域有关，$id8-9$ 与延长软件系统以及体系结构生命周期有关，最后 3 个则与系统实现密切相关。

表 1 非功能属性及其子属性

id	属性名	所包含子属性
1	Reliability	accuracy, availability, recoverability, fault-tolerance, robustness
2	Performance	recovery time, response time, shutdown time, start-up time, throughput
3	Security	auditability, access control, confidentiality, integrity, authentication, non-repudiation
4	Efficiency	response time, throughput, storage space
5	Cost	development costs, operational costs
6	Database	transaction management, persistence, concurrency
7	Distribution	compatibility, heterogeneity, concurrency
8	Supportability	adaptability, compatibility, installability, localizability, scalability, portability, testability, modifiability, reusability
9	Usability	accessibility, aesthetics, consistency
10	Implement Requirements	the 3rd party components, implementation languages, platform support, resource limits, standards-compliance
11	Interface Requirements	external system, interface formats
12	Physical Requirement	shape, size and weight of the system

2.3 非功能属性的规约

考虑到 XML 具备定义标记的灵活性、可护展性，并且能够实现与 UML 数据交换等特点，本文采用基于 XML 的两种规约模板——维度规约模板和关注点规约模板，分别对非功能属性及其子属性进行规约。

2.3.1 非功能属性规约

规约“1+X”模型中的“X”使用维度规约模板，它规约了一个非功能属性的基本信息和不同非功能属性之间的关系。基本信息包括以下内容：

1)id：唯一地标识该非功能属性。

2)名称：非功能属性的名字。

3)描述：对该非功能属性的简单描述。

4)应用示例：说明该非功能属性的典型应用，通常根据领域专家的经验得来。

5)关注点列表：描述体现该非功能属性的所有子属性集合。

6)影响：描述不同维度之间的关系。本文将非功能属性之间的关系归纳为一种贡献关系，该关系可以取两种值：“+”，“-”。“+”表示增强关系，如安全性和可用性。安全性的增强，一定会提高系统的可用性。“-”表明表示冲突关系，如安全性和响应时间、成本等。这里将相互之间具有负向贡献关系的维度称为相互冲突的维度。“1+X”模型中不同的次维会对体系结构产生不同的影响，因此，体系结构设计师将在冲突维度所对应的体系结构之间进行权衡。

一个关于非功能属性规约的示例如图 2 所示。非功能属性 Security 包含 6 个子属性，对 Reliability 维度中的 accuracy 和 availability 具有增强作用；而与 Performance，Efficiency 和 Cost 3 个非功能属性产生冲突。因为系统的安全性越高，占用的时间和空间会越大，必然降低系统的性能和效率。同样，增强安全性需要采取多种措施，致使软件开发的成本变高。

2.3.2 非功能子属性规约

传统的非功能属性及其之间的关系通常被表示成层次结构[1]，表明非功能属性可以被进一步细化。因而，我们在非功能属性规约的基础上，为非功能子属性的规约引入了两个新元素：

1)子关注点：表示非功能属性被进一步细化为粒度更小的属性的集合。如可修改性 modifiability，它包含 maintainability，scalability 和 structure reengineering 3 个粒度更小的子属性。

2)映射：表示非功能属性的可操作化结果。通过对非功能属性的长期观察，研究人员发现大多数非功能属性是可操作化的，即它们可以被映射为生命周期后续阶段中的一个功能或特征、设计决策或是影响设计或实现阶段的其它方案[7]，并且可以通过某些具体的操作或手段来实现。如机密性可通过加密技术来实现，因而对应于一个具体的横切操作(即 aspect)；而响应时间则映射为体系结构的一个技术约束。不同映射结果的类型用“type”表示，详见图 3 和表 2。非功能子属性规约示例如图 3 所示。

```
〈? xml version="1.0" encoding="UTF-8"?〉
〈Dimension id ="3" name="Security"〉
        〈Description〉
        It means assure the whole computer be in safe,
        including hardware,software,and information.
        〈/Description〉
        〈Examples〉 various encryption and decryption
                algorithms,digital signature
        〈/Examples〉
        〈Concerns〉
            〈Concern id="3.1" name="auditability"/〉
            〈Concern id="3.2" name="access control"/〉
            〈Concern id="3.3" name="confidentiality"/〉
            〈Concern id="3.4" name="integrity"/〉
            〈Concern id="3.5" name="authentication"/〉
            〈Concern id ="3.6" name ="non-repudiation"/〉
        〈/Concerns〉
        〈Impacts〉
            〈Dimension id ="1"〉
                〈Concern id ="1.1" type ="+"/〉
                〈Concern id ="1.2" type ="+"/〉
        〈/Dimension〉
            〈Dimension id="2" type="-"/〉
            〈Dimension id="4" type="-"/〉
            〈Dimension id="5" type="-"/〉
        〈/Impacts〉
〈/Dimension〉
```

图 2 “Security”维度规约

```
〈? xml version="1.0" encoding="UTF-8"?〉
〈Concerd id="3.3" name="Confidentiality"〉
        〈Description〉
            ensuring that infomation is accessible only
            to those authorized to have access
        〈/Description〉
        〈Examples〉the confidentiality of emails,user
            information,and account information,etc.
        〈/Examples〉
        〈Subconcerns〉
        〈/Subconcerns〉
        〈Mappings〉
            〈Operation id ="3.3.1" name ="Encryption"
                type="aspect"/〉
            〈Operation id="3.3.2" name="Decryption"
                type="aspect"/〉
        〈/Mappings〉
〈/Concern〉
```

图 3 “Confidentiality”关注点规约

综上所述，“1+X”模型展现了体系结构中对功能属性和非功能属性进行处理的两种不同手段，特别是以多维的方式分离并规约了非功能属性。通过为体系结构设计人员提供非功能属性库及其规约，本文的方法一方面大大简化了设计人员对非功能属性的设计工作，另一方面则可以根据非功能子属性规约所提供的信息(特别是 mappings 部分)，指导体系结构方面组件的建模。因而，“1+X”模型与提出的面向方面的

软件体系结构描述语言 AC2-ADL[5]衔接起来，成为有机的整体。最后需要说明的是，“1+X”模型中的非功能属性库仅是一个初始版本，随着研究工作的深入，将对其不断完善。

3 案例研究

为了进一步说明本文所提出的方法，结合网上拍卖系统[17]中的电子银行部分（下文简称电子银行系统）开展了案例研究。电子银行系统的需求描述如下：用户可以建立和注销自己的账户，并能够修改账户密码；可以进行查询账户余额及明细；为了参与拍卖，用户还须将银行账户的钱转入系统账户，拍卖结束后还可将余额转回银行。在上述过程中，要求系统能够记录用户的各项操作，提供安全措施，以保证所有的交易安全可靠、方便快捷。

由上可知，电子银行系统的功能需求有 3 个：(1)账户管理：包括申请和注销账户以及修改账户密码；(2)账户查询：包括查询账户余额以及浏览交易明细；(3)转账管理：实现银行与网上拍卖系统电子银行之间的资金流动。由于“1+X”模型将功能需求视为一个关注点，因此，这里不对功能需求做深入讨论。

利用“1+X”模型，可以直接从非功能属性维度中选取与本例相关的非功能属性以及子属性。这一工作分两步完成：首先选取符合系统要求的非功能属性；然后在被选中的非功能属性维度上进一步选取符合特定系统的非功能子属性。根据电子银行系统的需求，从“1+X”模型中选取 security，reliability 和 performance 3 个维度。接下来对每个维度进行分析，选取符合电子银行系统的非功能子属性。考虑到本案例涉及银行交易，对安全性的要求远高于一般系统，因而选取了 security 的所有子属性(6 个)。对用户来说，可能更加关心系统的可用性，故在 reliability 维度中重点选取了 availability。类似地，在 performance 维度中选取了 response time。

在选取了非功能属性以及其子属性之后，接下来的工作就是重用“1+X”模型中上述维度以及关注点的规约。为了节省篇幅，本文不再一一罗列电子银行系统中各维度以及关注点的规约，仅将它们的映射结果展示在表 2 中。通过表 2，可以看出大多数非功能属性是可操作化的，并且在后续的体系结构设计中多以 aspect 组件的形式被建模。

表 2 电子银行系统非功能属性映射结果

id	name	〈mappings〉
1.1	availability	〈Constraint id="1.2.1" name="Availability" type="aspect"/〉
2.2	Response time	〈Constraint id="3.1.1" name="RspsTm" type="decision"/〉
3.1	auditability	〈Operation id="3.1.1" name="Log" type="aspect"/〉
3.2	Access control	〈Operation id="3.2.1" name="AcsCtr" type="aspect"/〉
3.3	confidentiality	见图 3 中
3.4	integrity	〈Operation id="3.4.1" name="MD5" type="aspect"/〉
3.5	Authentication	〈Operation id="3.5.1" name=" DigSign" type="aspect"/〉
3.6	*non-repudiation*	〈Operation id="3.6.1" name="DigSign" type="aspect"/〉

由于本文给出的仅是关于非功能属性及其子属性的通用规约，因而在具体问题时可能会遇到描述粒度过粗，或者与具体系统要求不符的情况，此时需要结合具体的软件系统对“1+X”模型中的规约做进一步的精化或修正，这不是本文的重点，不再赘述。

4 相关工作

在软件体系结构阶段处理非功能属性是非常合适的[18]，尤其是面向方面技术的出现，为非功能属性的处理提供了新的思路。目前在软件体系结构设计阶段，利用 aspect 封装非功能属性，取得了一定的成果。其中较有代表性的如下：

文献[8]提出一个支持分布式、实时系统，用于 NFRs (Non-Functional Requirements)设计和分析的面向方面的框架——FDAF(Formal Design Analysis Framework)。在这一框架中，非功能需求被定义为方面库(aspect repository)中的可重用方面。与本文的工作相比，本文则致力于构建体系结构层通用的横切关注点库，而不仅仅是面向某个特定领域。

文献[2,9]提出了一种建模体系结构中非功能需求的分组机制，与传统的体系结构中处理非功能属性的方法类似，采用二维的方式来处理非功能属性，即体系结构设计被划分为两个阶段：首先进行传统的体系结构设计，继而用 XML 绑定器把非功能需求织入到设计方案中。尽管作者提出了用方面组件表示可操作化的非功能需求的语义，但仍然没有跳出功能-非功能的二维划分模式，也没有针对某个非功能属性给出具体的解决方案。本文的工作在体系结构阶段设计早期利用关注点多维分离原理来处理非功能属性，在后续阶段用方面组件来封装，继而可通过 AC2-ADL 进行描述。

文献[10,11]提出一种扩展 xADL2.0，支持将非功能需求作为可方面 NFRs (aspectable Non-Functional Requirements)注入到软件体系结构中。可方面 NFR 是那些在体系结构元素中多次重复出现的成分。然而作者认为某些质量属性不能作为可方面 NFR 来处理，因为某些质量属性不具备多次重复的特点，这与本文的工作有所不同，本文将质量属性视为非功能属性的一部分，并用规范的方式——关注点规约模板对它们进行处理。

此外，在如何划分安全性维度问题上，Bashir 等人[12]把 Security 划分成 7 个维度，分别是 authentication, access control, audit trail, confidentiality, integrity, availability, and non-repudiation。本文把 availability 作为一个独立的 NFR 关注点，并把它归入 Reliability 维度。

Chung 等人[1]分析了软件工程中的非功能属性，提出用非功能属性框架来处理它们。他们工作的重点是给出软件工程中非功能属性列表，并提供处理它们的手段。而本文的非功能属性库与其非功能需求列表有相似之处，但更强调体系结构设计阶段的非功能属性。

在关注点多维分离方面的具有代表性的研究工作有：文献[4]提出沿着多维的分解维度、分离相互重叠的关注点方法。文献[13]提出的 Cosmos 是一个关注点-空间建模模式，通过关注点、关注点之间关系和谓词来建模关注点-空间。关注点被划分成表示概念的逻辑关注点和表示软件系统元素的物理关注点两类。然而这两种方法只是提出了一个基本的抽象模型和初步的有关想法，没有针对每个阶段的开发任务，研究具体的多维分离关注点的方法。文献[14]提出一种基于关注点多维分离的需求工程方法，用超立方体(hypercube)表示

需求层的关注点空间，然而并没有给出具体的多维分离模型，并且文献[14]的工作仅限于需求阶段。文献[15]针对模型驱动软件项目过程采用多种开发方式、缺少系统化方法指导的问题，提出了一种模型驱动过程框架设计方法。该方法采用一般化、行为化和抽象化作为元关注维，对开发方式进行比较，并组合这三维的期望演化曲线，给出过程实现模型框架。然而该文所描述的三维只是一个抽象概念框架，并不对应具体软件系统的开发过程。

结束语 非功能属性的处理长期以来一直困扰着软件开发人员，成为软件工程领域内一个热点和难点问题。本文立足于软件体系结构设计的早期阶段，提出一种多维分离非功能属性的建模方法。该方法通过"1＋X"模型，将软件体系结构的非功能属性划分为 X 个维度，并进一步细化为包含多个非功能子属性的集合，同时提供了软件体系结构的非功能属性库，采用 XML 对非功能属性和子属性进行了规约。"1＋X"模型直观地描述了在软件体系结构设计阶段对非功能属性和功能属性所采用的不同处理方式。在实际系统的软件体系结构设计中，可以借助本文所提供的非功能属性库选取适合的非功能属性，直接重用非功能属性及其子属性的规约。本文提出的方法，为处理软件体系结构的非功能属性提供了一种新的思路。

在今后的一段时间内将进一步完善非功能属性库，明确"1＋X"模型中 1 与 X 之间的关系及其规约方法，研究非功能子属性向体系结构构成元素映射的映射规则。

参 考 文 献

[1] Chung L, Nixon B A, Yu E, et al. Non-Functional Requirements in Software Engineering [M]. Norwell: Kluwer Academic Publishers, 2000

[2] Xu L, Ziv H, Richardson D, et al. Towards Modeling Non-Functional Requirements in Software Architecture[EB/OL]. http://trese.cs.utwente.nl/early-aspects-AOSD2005/Papers/12_XuZivRichardsonLiu_UCIrvineCSFullerton.pdf

[3] Kiczales G, Lamping J, Mendhekar A, et al. Aspect-Oriented Programming[C]// Proceedings of 11th European Conference on Object-Oriented Programming (ECOOP1997). New York: Springer-Verlag, 1997: 220-242

[4] Tarr P, Ossher H, Harrison W, et al. N Degrees of Separation: Multi-Dimensional Separation of Concerns[C]// Proceedings of the 21st Int'l Conf. on Software Engineering (ICSE1999). Washington: IEEE, 1999: 107-119

[5] Wen Jing, Shi Ying, Zhang Lin-lin. Architectural design of the Online Auction System with Aspect-Oriented Software Architecture Design approach[C]// Proceedings of 2008 IEEE International Conference on e-Business Engineering (ICEBE 2008). Los Alamitos: IEEE, 2007: 5-12

[6] Eeles P. Capturing architectural requirements[R]. 2004

[7] Rashid A, Moreira A, Araujo J. Modularization and Composition of Aspectual Requirements[C]// Proceedings of the 2nd on Aspect-Oriented Software Development. New York: ACM, 2003: 11-20

[8] Dai L, Cooper K, Wong W E. Modeling and Analysis of Performance Aspects for Software Architecture: a UML-based approach [J]. Software Engineering and Knowledge Engineering, 2006, 16(3): 347-378

[9] Xu L, Ziv H, Alspaugh T A, et al. An Architectural Pattern for Non-functional Dependability Requirements [J]. Systems & Software, 2006, 79(10): 1370-1378

[10] Bagheri H. Injecting Security as Aspectable NFR into Software Architecture[C]// Proceedings of the 14th Asia-Pacific Software Engineering Conference (APSEC'07). 2007: 310-317

[11] Bagheri H, Mirian-Hosseinabadi S H, Esfahani H C. An Aspect Enhanced Method of the NFR Modeling in Software Architecture[C]// Proceedings of 10th International Conference on Information Technology (ICIT 2007). 2007: 240-242

[12] Bashir I, Serafini E, Wall K. Securing Network Software Applications: Introduction[J]. Communications of the ACM, 2001, 44(2): 28-30

[13] Sutton S, Rouvellou I. Modeling of Software Concerns in Cosmos[C]// Proceedings of the 1st International Conference on Aspect-oriented Software Development (AOSD'02). New York: ACM Press, 2002: 127-133

[14] Moreira A, Araujo J, Rashid A. Multi-Dimensional Separation of Concerns in Requirements Engineering [C]// Proceedings of 13th Requirements Engineering Conference (RE'05). Washington: IEEE, 2005: 285-296

[15] 段玉聪，顾毓清. 多维关注分离的模型驱动过程框架设计方法[J]. 软件学报，2006, 17(8): 1707-1716

[16] Pinto M, G'amez N, Fuentes L. towards the Architectural Definition of the Health Watcher System with AO-ADL[C]// Proceeding of the Early Aspects Workshop at ICSE 2007. May 2007: 94-114

[17] Auction System[OL]. http://lgl.epfl.ch/research/use_cases/RE-A2-case-studies/auction/problem-description.html

[18] 杨放春，龙湘明. 软件非功能属性研究[J]. 北京邮电大学学报，2004, 27(3): 1-12

(上接第 81 页)

[2] Adler R, Feldman R, Taqqu M. A Practical Guide to Heavy Tails[M]. Birkhauser, Boston: Chapman & Hall, 1998

[3] Samorodnitsky G, Taqqu M. Stable Non-Gaussian Random Processes[M]. London: Chapman & Hall, 1994

[4] 闻勇. 具有重尾特性的自相似网络通信量建模及预测[D]. 武汉：华中科技大学，2006

[5] Rice J. Mathematical Statistics and Data Analysis, Second Edition[M]. Wadsworth Inc, 1995

[6] Li H X, Chen C L, Huang H P. Fuzzy Neural Intelligent Systems: Mathematical Foundation and the Applications in Engineering[M]. FL: CRC Press, 2001

[7] URL: http://ita.ee.lbl.gov/html/traces.html

[8] URL: http://crawdad.cs.dartmouth.edu/data.php