

反射式需求工程过程演化建模

张国生

(云南大学 信息学院, 云南 昆明 650091)

摘 要: 用反射机制为软件系统需求工程过程演化建模, 将需求工程过程的开发活动作为反射系统的基层, 实现软件系统的功能需求, 将软件系统的非功能需求作为反射系统的元层, 对需求工程过程的开发活动进行调节、控制, 并将非功能需求的性质反射到功能需求. 定义了有色 Petri 网及其范畴, 并用有色 Petri 网范畴描述软件系统非功能需求性质的模拟、保持和反射机制; 用有色 Petri 网建立反射式需求工程过程演化形式化模型, 模拟了软件系统需求工程过程中功能需求和非功能需求的迭代、增量、演化, 为软件系统需求工程过程演化建模提出了一种新的方法.

关键词: 反射; 需求工程过程; 演化; 有色 Petri 网; 范畴; 功能需求

中图分类号: TP311

文献标识码: A

文章编号: 1000-7180(2012)07-0154-04

Reflecting Requirements Engineering Processes Evolution Modeling

ZHANG Guo-sheng

(School of Information, Yunnan University, Kunming 650091, China)

Abstract: In this paper, reflecting mechanism is used to model for software system requirements engineering processes. Development activities of requirements engineering processes set as base level of reflecting system complete functional requirements of software system. Nonfunctional requirements of software system set as meta level of reflecting system adjust and control development activities of requirements engineering processes. Meanwhile, the properties of nonfunctional requirements are reflected to functional requirements. The CPN and its category are defined in this paper. Simulation, preservation and reflection mechanism of nonfunctional requirements properties of the software system are stated with CPN category. Reflecting requirements engineering processes evolution formal model is created by CPN. The model simulates iteration, increment and evolution of functional and nonfunctional requirements in the software system requirements engineering processes. It presents a new approach for software system requirements engineering processes modeling.

Key words: reflection; requirements engineering processes; evolution; CPN; category; functional requirements

1 引言

软件过程是一个对话的过程, 在对话中获取需要转化为软件的知识, 并在软件中实现这些知识. 过程提供了涉众与设计人员之间、涉众与不断演化的工具之间以及设计人员与不断演化的工具之间的互动. 软件开发是一个迭代的过程, 在其中演化的工具本身就作为沟通的媒介, 每新一轮对话都可以从涉

众获得更有用的知识, 整个需求过程的活动需要不断地反馈、增量、迭代、演化, 每一个需求过程活动也构成一个子生命周期^[1].

为了改善需求定义的方式, 当执行需求工程时, 将实施 3 种不同的子过程: (1) 改进知识获取和知识共享, 使得更完整地理解应用领域的限制和利益相关者的需求; (2) 在定义需求时, 更加强调迭代; (3) 更有效的沟通和协调工具, 使所有涉众进行有效的

收稿日期: 2011-11-25; 修回日期: 2011-12-23

基金项目: 国家自然科学基金项目(60463002); 云南大学重点教改项目(WX070142)

合作。

文献[2]认为,构建一个软件系统最困难的部分是精确地决定构建什么,因此,软件构建者为用户实现的最重要功能是产品需求的迭代抽取和精化。

文献[3]将软件演化定义为软件系统在其生命周期中不断维护、不断完善的系统动力学行为。如果系统的一个或多个部件随时间经历不断改进,则称之为演化,演化的目的是为了适应环境的变化或提高用户的满意度。

软件演化是一个多层次、多循环、多代理的反馈系统。来自于全局软件过程的可能约束,包括技术、业务、市场、用户以及其它活动将形成一个多层次、多循环的反馈系统。

需求演化是指软件需求从不明确到明确的变化过程。在问题域极其复杂或者需求不稳定时,可以采用演化模型,演化模型采用增量迭代和逐步展开的方式进行需求开发。需求演化是系统需求不断完善、发展的必然趋势,从需求工程过程的角度,对需求演化过程建模,突出了需求演化中的迭代和反馈^[4]。

2 需求分类

需求可以分为不同的类别,不同的类别有不同的特性和不同的处理要求。在各种需求的分类中,最常见的是将需求分成以下5种类型^[5]:

(1) 功能需求:在不考虑物理约束的情况下,用户希望系统所能够执行的活动,这些活动可以帮助用户完成任务。功能需求主要表现为系统和环境之间的行为交互。功能需求通常体现为3个层次:业务需求、用户需求和系统需求。

(2) 性能需求:系统整体或系统组成部分应该拥有的性能特征,如:CPU利用率、内存使用率等。

(3) 质量属性:系统完成工作的质量,即系统需要在一个“好的程度”上实现功能需求。软件的质量包含6个特征:功能性、可靠性、可用性、效率、可维护性和可移植性,同时为每个特征定义了更加详细的子特征。

(4) 对外接口:系统和环境中其他系统之间需要建立的接口,包括硬件接口、软件接口、数据库接口等。

(5) 约束:进行系统构造时需要遵守的限制,如:编程语言、硬件设施等。

其中,除功能需求之外的其他4种类型的需求统称为非功能需求。功能需求是用户对软件系统的显式要求,用户在软件系统创建之前就可以清晰地

向开发者表达这种要求,而非功能需求属于隐式要求,用户在软件系统创建之前无法清晰地告诉开发者他们希望该系统具备什么样的非功能性特征,但在软件系统投入使用之后,他们却可以快速地判断出软件系统的哪一部分非功能需求不能满足他们的要求。

3 有色 Petri 网及其范畴的形式化定义

3.1 有色 Petri 网形式化定义

定义1 有色 Petri 网定义为一个多元组 $N = (P, T, Pre, Post, pri, C, G, cd, P_{out}, P_{in}, T_{out-in}, M)$, 其中

(1) P 和 T 分别为有限库所集和有限变迁集, 且 $P \cap T = \Phi$;

(2) Pre 和 $Post$ 为弧函数, 分别定义了后面射进或是向前射出的颜色函数, 满足

$$Pre, Post: P \times T \rightarrow C(p)_{MS}$$

其中, $C(p)_{MS}$ 是库所 p 的数据类型集上的多重集, 分别表示前置条件和后置条件;

(3) $pri: T \rightarrow IN$, 是将一个变迁和一个优先级关联起来的向量。默认情况下, 假设 $\forall t \in T, pri[t] = 0$;

(4) $C = \{C_1, \dots, C_n\}$ 是基本颜色类的集合。基本颜色类是有限的、不相交的, 并且每个类可以被划分为静态子类:

$$C_i = C_i^1 \cup \dots \cup C_i^n;$$

(5) G 是 Guard 函数, 定义为 $G: T \rightarrow G(t)$, 满足 $\forall t \in T: Type(G(t)) = B$

用于指定变迁发生须满足的前提条件;

(6) cd 为一个映射, 它将网的一个库所和变迁与一个颜色域相关联, 后者通过基本颜色类的笛卡尔积运算进行复合来定义。Guard 函数可以对变迁的颜色域进行限制;

(7) $P_{out} \subset P$ 是一个库所集, 称为输出接口库所集, 对于 $\forall p_{out} \in P_{out}: M_0(p_{out}) = \langle \rangle$;

(8) $P_{in} \subset P$ 是一个库所集, 称为输入接口库所集, 对于 $\forall p_{in} \in P_{in}: M_0(p_{in}) = \langle \rangle$, 且集合 P_{out} 和 P_{in} 两两不相连;

(9) $T_{out-in}: P_{out} \rightarrow P_{in}$ 是一个双射, 满足

$$\textcircled{1} \quad \forall (p_{out}, T_{out-in}(p_{out})) \in P_{out} \times P_{in}, \forall t_n \in (T_{out-in}(p_{out})), \exists t_1, \dots, t_{n-1} \in T, t_1 \in p_{out} \wedge \forall i \in \{1, \dots, n-1\}: t_i \cap t_{i+1} \neq \Phi$$

$$\textcircled{2} \quad \forall (p_{out}, T_{out-in}(p_{out})) \in P_{out} \times P_{in}, \forall t_1 \in p_{out}, \exists t_1, \dots, t_n \in T, t_n \in (T_{out-in}(p_{out}))' \wedge \forall i \in \{1, \dots,$$

$n-1\}; t_i \cap t_{i+1} \neq \Phi$

③ $\forall (p_{out}, T_{out-in}(p_{out})) \in P_{out} \times P_{in}: p_{out} \cap T_{out-in}(p_{out}) = \Phi$

3.2 有色 Petri 网范畴形式化定义

自 1945 年 Eilenberg 和 MacLane 为描述“同构是自然的”之确切含义而引入“范畴”以来,范畴理论有了很大的发展. 范畴由对象和对象之间的射组成. 范畴论通过范畴、函子、自然变换、函子范畴、范射、极限等,提供了一个方便的概念语言,使原来一些模糊的概念得以精确地刻画,并且把许多数学结构,如:群、环、拓扑空间、向量空间等统一起来,具有更高的抽象层次,范畴论提供了一个较好的架构,并容纳了一些新的思考方法,它不只是研究单个数学结构的性质,而是研究结构整体的性质. 范畴论侧重于对象之间的关系,而不是对象自身的描述,并且支持图形化表示^[6]:对象为结点,态射为有向边.

定义 2 一个范畴由以下 3 部分组成:

(1) 多个对象组成的集合;

(2) 多个态射组成的集合;

(3) 态射合成法则:对于态射 $f: A \rightarrow B, g: B \rightarrow C$, 如果 $\text{cod}(f) = \text{dom}(g)$, 那么 $g \circ f: A \rightarrow C$ 就有定义.

定义 3 有色 Petri 网 N 的范畴 MN^+ 的对象集由形如 $\Sigma = (N, M)$ 的有色 Petri 网系统组成, 每个有色 Petri 网系统由无多重圈的有色网 N 和标识 $M: P \rightarrow \mathbb{N}^{|P|}$ 组成; 态射 (f, F) 是有色 Petri 网 $\Sigma = (N, M_0)$ 到 $\Sigma' = (N', M'_0)$ 的形态映射, $f: T \rightarrow T^+, F: P' \rightarrow P$, 使得 $(M_0, M'_0) \in F^+$, 且对 $\forall t \in T, \forall p' \in P'$, 有 $W(F(p'), t) \geq W'(p', f(t)) \wedge W(t, F(p')) \leq W'(f(t), p')$.

定义 4 (模拟)态射 (f, F) 是两个有色 Petri 网系统 $\Sigma = (N, M)$ 和 $\Sigma' = (N', M')$ 之间的一个模拟, 当且仅当 $(M, M') \in F^+$, 且

$$\forall (M_1, M'_1) \in F^+: (M_1 \xrightarrow{t} M_2 \Rightarrow (M'_1 \xrightarrow{f(t)} M'_2 \wedge (M_2, M'_2) \in F^+))$$

定义 5 (保持)称带标识的有色 Petri 网 (N, M) 在解释 θ 下满足的性质 Φ 可以由态射 (f, F) 保持, 如果对于有色映像 Petri 网 (N', M') 下式成立

$$(N, M) \models_{\theta} \Phi \Rightarrow (N', M') \models_{\theta} \Phi$$

性质 Φ 可由态射 (f, F) 保持, 当且仅当

$$\forall \Upsilon: \Upsilon \in \parallel \Phi \parallel_{\theta} \Rightarrow f\Upsilon \in \parallel \Phi \parallel_{\theta}$$

其中, $\Upsilon = \Upsilon_0, \Upsilon_1, \dots, \Upsilon_n$ 为有限计算序列.

定义 6 (反射)称带标识的有色映像 Petri 网 (N', M') 在解释 $f\theta$ 下满足的性质 Φ 可以被态射 $(f,$

$F)$ 反射, 如果对于源有色 Petri 网 (N, M) 下式成立

$$(N', M') \models_{f\theta} \Phi \Rightarrow (N, M) \models_{\theta} \Phi$$

性质 Φ 可被态射 (f, F) 反射, 当且仅当

$$\forall \Upsilon: f\Upsilon \in \parallel \Phi \parallel_{f\theta} \Rightarrow \Upsilon \in \parallel \Phi \parallel_{\theta}$$

其中, $\Upsilon = \Upsilon_0, \Upsilon_1, \dots, \Upsilon_n$ 为有限计算序列.

4 计算反射及需求过程演化建模

4.1 计算反射

1982 年 Smith 最早介绍了程序反射的概念并提出了反射假说:“既然我们可以构造包含负责管理有关某个外部世界的表示的内部过程的计算过程, 并通过它来对那个外部世界进行推理, 那么也可以构造能够对自身进行推理的计算过程, 它包含负责管理有关自身的操作和结构的表示的内部过程.”也就是说, 借助反射计算过程能够访问、推理和改变有关自身的表示.

1987 年 Maes 首先系统地阐述了计算反射和反射系统的概念^[7], Maes 对计算反射的定义为: 计算反射是反射系统在运行中所做的有关自身的计算和控制. 反射系统是关于系统本身的系统, 而且系统中用于描述系统的数据结构和系统本身之间存在因果联系, 即如果系统或者有关系统的描述发生了改变, 那么有关系统的描述或者系统本身就会发生相应的改变. 在反射系统中, 用于描述系统本身的数据结构称为系统的自描述信息, 它们是实现反射的基础, 反射系统通过维护正确的自描述信息来实现运行时对系统行为的监视, 并根据环境和应用程序的需要对系统进行重新配置.

一个反射机制通常由两个部分组成: 元层和基层. 基层是被反射的层次, 是元层计算的数据, 元层是进行反射的层次, 通过“具体化”把元层映射到基层, 通过“反射”把基层映射到元层.

反射是一种系统的推理和作用于自身的能力. 反射具有两种功能: 内省和调解. 所谓内省, 指的是元层对象能查看并控制基层对象的相关信息, 如基层结构和行为方面的信息. 而调解指的是修改一个对象或者类的属性的能力. 由于反射系统可以提供自表示能力, 以及具有参照这种自表示能力对自身进行修改的能力, 因此, 将反射系统引入到需求工程过程中可以提高过程的演化性.

4.2 需求过程演化建模

需求工程过程的主要开发活动包括: 需求获取、需求分析、需求规格说明、需求验证. 为了尽量不给设计、实现、测试等后继开发活动带来不必要的影

响,需求规格说明文档中定义的需求必须正确、准确地反映客户的意图.在需求工程过程中,通过与涉众面对面交流获取的涉众素材验证需求规格说明文档,并迭代、增量、反馈修改需求规格说明,直到需求规格说明尽可能准确、一致地反映涉众需求为止.

软件系统的需求可以分为功能需求和非功能需求,根据系统的反射机制,建立反射式需求过程演化模型,如图1所示.将软件系统非功能需求作为反射系统的元层,需求工程过程的主要开发活动作为基层,实现软件系统的功能需求,元层对象和基层对象因果关联,用元层的非功能需求性质去控制和调节基层的需求工程过程开发活动,实现需求工程过程开发活动的不断演化,元层中软件系统的非功能需求性质的模拟、保持和反射,用有色 Petri 网 N 的范畴 MN^+ 表示^[8].同时,需求工程过程开发活动的演化又映射为元层中软件系统的非功能需求性质,这样,反复不断地控制、调节、映射,使需求工程的最终结果——需求规格说明文档中定义的需求更加正确、准确地反映涉众的意图,真正做到需求文档的无二义、完备、正确、可理解、一致、精确和可靠.

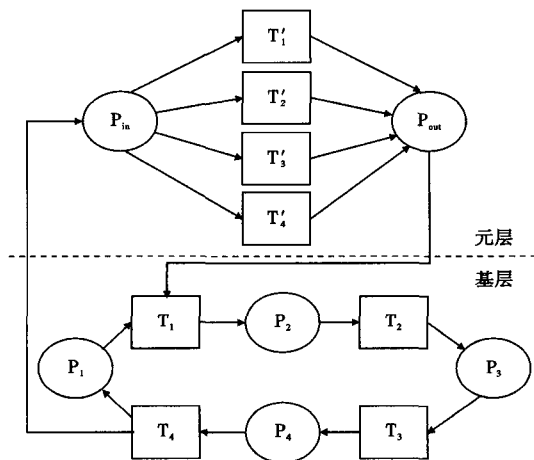


图1 反射式需求过程演化模型

在反射式需求过程演化模型中,元层中的库所 P_{in} 、 P_{out} 分别表示元层有色 Petri 网的输入、输出接口,元层中的变迁 T'_1 、 T'_2 、 T'_3 、 T'_4 分别表示软件系统的非功能需求活动:性能需求、质量属性、对外接口、约束;基层中的变迁 T_1 、 T_2 、 T_3 、 T_4 分别表示需求获取、需求分析、需求规格说明、需求验证.

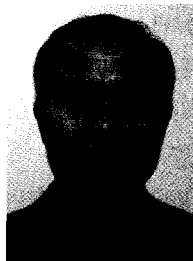
5 结束语

本文引入反射机制建模,将模型分为两层:元层和基层.需求工程过程的开发活动实现系统的功能需求作为反射系统的基层,用非功能需求作为反射系统的元层,对需求工程过程开发活动进行控制,同时,将非功能需求反射到功能需求.定义了有色 Petri 网实现反射式需求过程演化形式化模型,用有色 Petri 网范畴表示软件系统的非功能需求性质的模拟、保持和反射,较好地模拟了软件需求工程过程中功能需求和非功能需求的迭代、增量、演化.

参考文献:

- [1] 张国生. 敏捷需求过程建模[J]. 微电子学与计算机, 2012,29(2):27-30.
- [2] Brooks F P. No silver bullet: essence and accidents of software engineering[J]. IEEE Computer, 1987, 20(4): 10-19.
- [3] Lehman M M. Feedback in the software evolution process[J]. Information and Software Technology, 1996(38): 681-686.
- [4] 张国生. 需求演化过程建模[J]. 微电子学与计算机, 2012,29(5):54-57.
- [5] IEEE. IEEE STD 830-1998 IEEE recommended practice for software specifications[S]. Institute of Electrical and Electronics Engineering Inc, 1998.
- [6] Michael Baar, Wells Charics. Category theory for computing science[M]. Englewood Cliffs, NJ: Prentice Hall, 1990.
- [7] Maes P. Concepts and experiments in computational reflection[J]. ACM SIGPLAN Notices, 1987, 22(10): 147-155.
- [8] Claude Girault, Rudiger Valk. Petri nets for systems engineering: a guide to modeling, verification, and applications[M]. Springer-Verlag, 2003.

作者简介:



张国生 男,(1968—),硕士,副教授,研究方向为软件工程及其形式化.