文章编号: 1673- 0062(2009) 04- 0079- 05

基于 Z语言的互联网登陆系统的形式化规格与验证

闫仕宇

(南华大学 计算机科学与技术学院, 湖南 衡阳 421001)

摘 要: Z语言是一种基于集合和一阶谓词逻辑的模式规约语言, 可产生精确地需求规格说明. 本文用形式化语言 Z对互联网登陆系统的主要操作模式进行规格说明, 接着通过形式化验证, 证明设计的规格说明能够满足用户的需求, 提高了系统的可靠性和稳定性.

关键词: 登陆系统; Z语言; 形式化规格; 验证
中图分类号: TP391. 41 文献标识码: B

Formal Specification and Verification of the Internet Logged System Based on Z Language

YAN Shi-yu

(School of Computer Science and Technology, University of South China, Hengyang Hunan 421001, China)

Abstract: Z language is a model- based specification language based on set theory and first- order predicate logic. It can be used to precisely express requirement specifications. This paper formulated the requirements of software with Z language on main module of the Internet Logged system, then verified it in form. The results show that the specification can meet the users' requirements, advance the reliability and stability of the system.

Key words: logged system; Z language; formal specification; verification

形式化方法[1]是建立在严格的数学基础上的系统开发方法. 从广义的角度, 形式化方法是软件开发过程中分析、设计及实现的系统工程方法. 狭义的角度, 形式化方法是软件规格和验证的方法. 形式化规格是通过具有明确数学定义的文法和语义的方法或语言对软件的期望特征或者行为进行的精确、简洁描述. 形式化验证是基于已建立的形式化规格, 对软件的相关特征进行评价的数学分析和证明. 形式化方法是提高软件的正确性和可靠性的重要手段. 由于在软件开发过程中一般使用自然语言进行交流也为软件的开发增加了许多的歧义, 这些都使得软件的可靠性难以得到保证. 20世纪 90年代以来, 在国际上, 形式化方法已成为软件开发中重要的可信软件技术之一[2]. 实践证明, 通过形式规格确实可以增强对系统的理解从而发现了系统的错误, 通过形式化

收稿日期: 2009- 05- 08
作者简介: 闫仕宇 (1981-), 男, 湖南衡阳人, 南华大学计算机科学与技术学院助教, 硕士. 主要研究方向: 计算数学, 软件开发形式化方法等.

验证确实可以发现其他方法难以发现的错误. 由于编程的多样性也增加了系统集成的难度. 现在高级程序设计语言种类众多, 这些语言不但在表达能力, 适用范围上各不相同, 而且其结构及语义也是大相径庭的; 另外, 由于并发系统的大量出现使得软件的复杂性呈现出指数增长的趋势. 因此, 互联网登陆系统的形式化规格与正确性验证成了一个具有挑战性的课题.

软件工程形式化的基础是形式化的描述语言. 在众多的形式化语言中, Z语言是一种应用较为广泛的行为规格语言. 对互联网登录系统的形式化描述及其形式化验证, 目前国内进行这方面的研究还很少. 但是用 Z语言进行系统的形式化描述, 已经有很多成果[3]. 如 Agent系统 (A Conceptual Framework for Agent Definition and Development)[4], 飞行导航系统 (Formal Specification of a Flight Guidance System)[5]等. 这些系统使用了 Z语言对系统结构进行了描述, 较好地解决了相关领域中的问题, 取得了良好的效果.

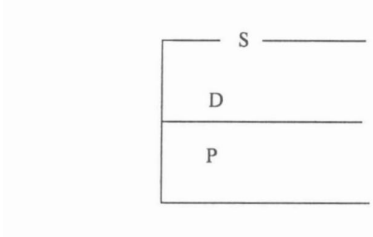
使用形式化方法设计互联网登陆系统, 能够很好地解决开发中诸如描述不一致, 标准不统一等问题, 同时通过形式化验证, 实现系统的可靠性和安全性. 本文用形式化描述语言—Z语言对互联网登陆系统进行了形式化规格, 并对主要的操作模式进行形式化验证.

1 Z语言简介

Z语言[6]是由英国 Oxford大学程序研究组 PGR的 Jean Raymond Abrial, Bernard Sufrin等人设计的一种基于一阶谓词逻辑和集合论的形式规格说明语言, 它采用了严格的数学理论, 可以产生简明、精确、无歧义且可证明的规格说明.

Z语言的核心是 Z模式, 它有两种模式: 状态模式和操作模式. 状态模式定义目标软件系统某一部分的状态空间及其约束特征. 操作模式描述了系统某一部分的行为特征, 它通过描述操作前该部分的状态值和操作后该部分状态值之间的关系来定义系统该部分的一种操作特征. 模式还可以修饰, 模式修饰的作用是将修饰应用到被修饰模式的声明部分中所有的变量.

Z模式是 Z语言的基本结构, 这种结构具有较强的描述软件系统的抽象状态和操作的功能. 一个模式包含模式的名字 (S)、声明部分 (D)、断言部分或者谓词部分 (P). 模式的名字在规格中随处调用, 也可以作为一个类型名; 模式的声明部分引入变量及其类型, 这些变量为该模式的局部变量; 断言部分描述了在这些局部变量之间, 或者局部变量与在该模式之间声明的全局变量 (常量)之间的不变式关系.

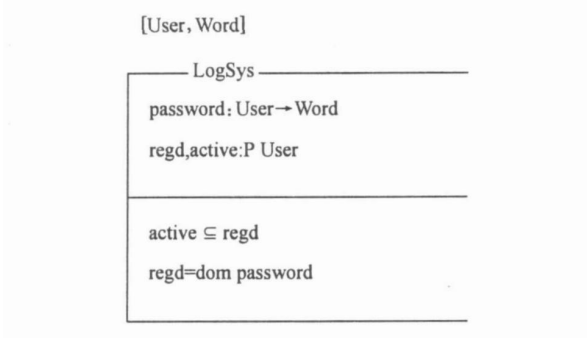


2 互联网登陆系统的形式化规格

互联网登陆系统一般包括 5个部分: 注册一个新的用户和口令; 注销一个用户及其口令; 让一个用户登陆; 让一个已登陆的用户修改其口令; 让一个已登陆的用户退出系统. 这 5个部分包含了 5个主要的操作. 为了实现上述模块的形式化描述, 需要对系统进行抽象化, 本文用 Z语言对互联网登陆系统进行分析, 对互联网登陆系统的主要操作问题进行形式化描述.

2.1 定义全局变量 User, Word

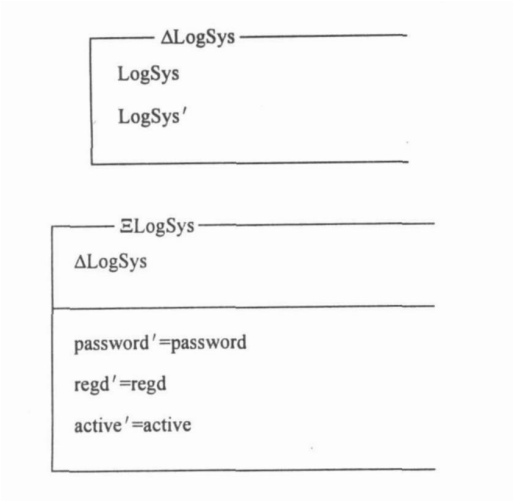
给系统抽象出两个变量. 一个是用户: User, 另一个是口令 Word 这个两个变量都是全局变量. 对于口令 password是一个由变量 User和变量 Word之间的部分函数关系 $User \rightarrow Word$, 用户可以分为在线用户 active, 注册用户 regd, 这两变量都是属于 User的幂集类型. 下面是对系统抽象状态的描述. 模式名为 LogSys, 定义模式的声明部分和断言部分. 如下,



2.2 定义 ΔLoySys, ΞLogSys和 InitLogSys

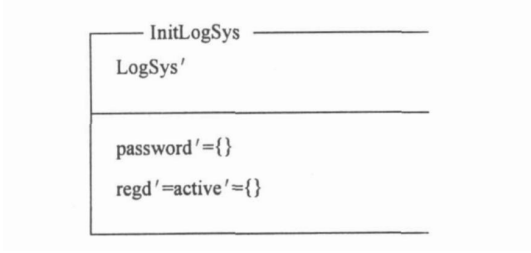
模式 Δ和 Ξ表示是对两类经常使用的模式的简写, 其目的是使用对模式的描述以及相应的规格说明更加简洁. 对状态模式 LogSys, ΔLoySys是由一个状态 LogSys和相应的后状态 LogSys′进行组合而成得到的模式. ΞLogSys是一个操作模

式, 该模式不引起系统状态的任何改变.



定义初始化模式 InitLogSys

初始化模式是对系统状态变量进行初始化, 对相应的状态变量赋初值, 但它不是通过对其他任何状态进行操作而得到的, 故没有前状态.

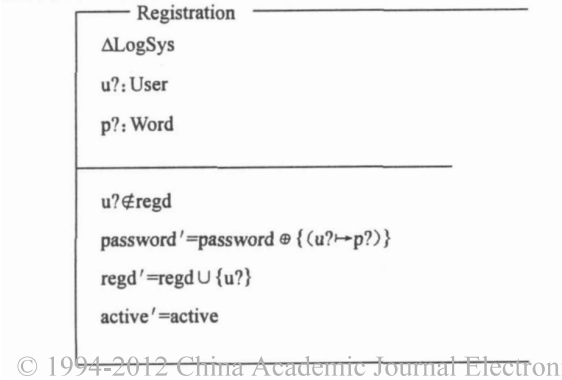


2 3 操作模式

从操作上看, 对系统的操作可为 5个方面: 用户注册; 注销一个用户及其口令; 让一个用户登陆; 让一个已登陆的用户修改其口令; 让一个已登陆的用户退出系统.

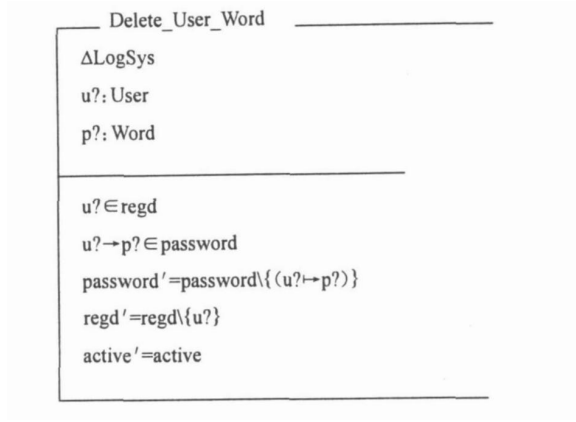
2 3 1 用户注册模式 Registration

该操作模式输入变量 $u?$, $p?$, $u?$ 表示注册用户名, $p?$ 表示注册用户口令, 执行该操作的前置条件是, 注册的用户是不是已注册的用户, $u? \notin regd$ 口令 password和注册用户 regd的后状态发生变化, 在线用户 active未发生变化.



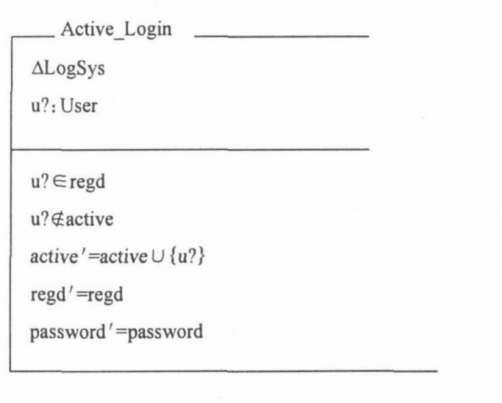
2 3 2 注销用户和口令操作模式 Delete_User_Word

在该模式的断言部分, 执行该操作的一个前置条件是注销的用户 $u?$ 一定是属于已注册的用户 regd, $u? \in regd$



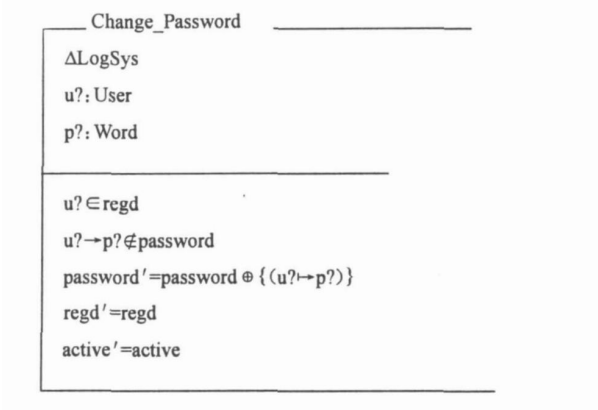
2 3 3 定义用户登陆模式 Active_Login

用户对登陆模式需要满足两个前置条件: 用户是已经注册用户 $u? \in regd$, 且登陆用户不是在线用户 $u? \notin active$



2 3 4 已登陆用户修改其口令模式 Change_Password

执行该操作需要满足两个前置条件: 用户肯定是属于已注册用户 $u? \in regd$, 用户 $u?$ 与口令 $p?$ 部分函数关系发生更改.



2 3 5 已登陆的用户退出系统模式 active_Logout

已登陆的用户退出系统,在线用户减少,后状态 active′发生变化.

Active_Logout

ΔLogSys

u?: User

u? ∈ regd

active′ = active \ {u?}

password′ = password

regd′ = regd

那么一个完整的操作描述为

LogSys≅ Registration ∨ Delete_User_Word ∨ Active_Login ∨ Change_Password ∨ Active_Logout

3 形式化验证

这里我们用 Z/EVES2.0工具[7]来验证第 2部分定义的抽象类型以及操作模式, Z/EVES是个很好用的 Z语法、类型检查和证明辅助工具. 第 2部分中我们用了 Z语法对登陆系统形式化规格描述,在写出规格说明后,需要对规格说明的严密性进行证明,定理证明则可以消除规格说明中的模糊性和不一致性,从而验证规格说明是否满足用户需求. 在这一部分我们首先利用 Z/EVES2.0工具来实现类型检查验证,再进行形式化推理证明,主要包括初始化定理及其证明,前置条件验证.

3.1 类型检查

例如,第 2部分中定义抽象模式 LogSys,定义如下,

LogSys

password: User ⇸ Word

regd, active: P User

active ∈ regd

regd = dom password

通过 Z/EVES2.0工具证明功能,发现类型关系出现错误. 工具界面会出现错误提示: type of local regd is not a power set of type of local active, 根据错误提示就知道了,在线用户 active变量类型不属于注册用户的一个元素,而是一个子集. 修改类型关系,通过 Z/EVES2.0工具验证. 正确结果如图 1.

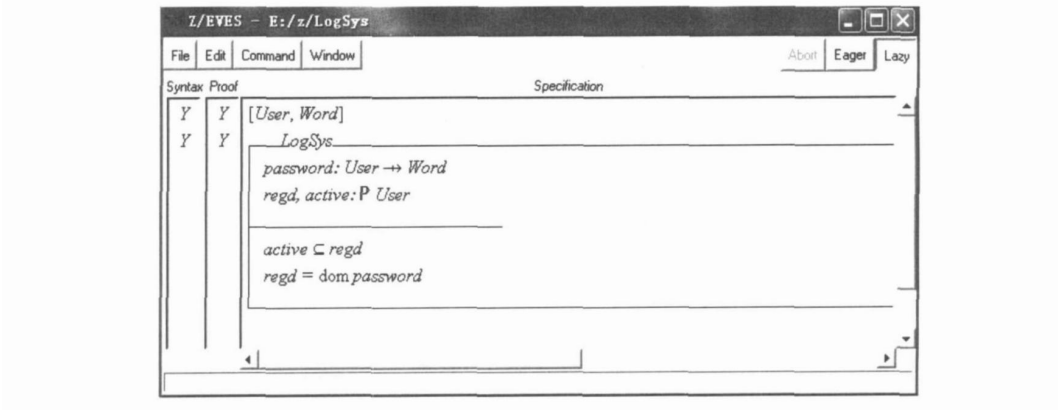


图 1 Z/EVES2.0工具形式化验证结果

Fig.1 The result of formal verification by the Z/EVES2.0

通过以上的方法,对第 2部分的操作模式可以逐一验证,找出其类型不一致的错误和语法上的错误,加以验证和修正.

3.2 初始状态存在验证

在进行初始状态推理证明之前,需要用到一下定律规则,这些定律规则都已经得到了证明[8]:

规则 1 $\dfrac{\exists x \bullet S(y \wedge x = t)}{t \in S \wedge y[t/x]}$ (点规则)

规则 2 $(\exists x \mid P \bullet Q) \Leftrightarrow (\exists x \bullet P \wedge Q)$

规则 3 $\mathrm{dom}\,\Phi = \Phi$

规则 4 $\mathrm{ran}\,\Phi = \Phi$

规则 5 $\Phi \in PS$ (表示集合 S的幂集)

规则 6 $\Phi \in S \quad T$

规则 7 $S \cap \Phi = \Phi$

对登陆系统的状态模式, 初始状态的系统 InitLogSys, 证明初始化状态存在, 先给出初始化定理.

定理　⊢ ∃ LogSys′• InitLogSys

这个定理的含义是确实存在一个 LogSys′系统状态满足 InitLogSys的谓词.

证　展开该定理, 有

⊢ ∃ password′: User → Word;

regd′: P User;

active′ P User | active′ ⊆ regd′ ∧ regd′ = dom password′• active′ = Φ∧ regd′ = Φ∧ password′ = Φ

利用规则 2可去掉“|　”, 利用规则 1对谓词中的变量进行替换, 可得到如下等价式:

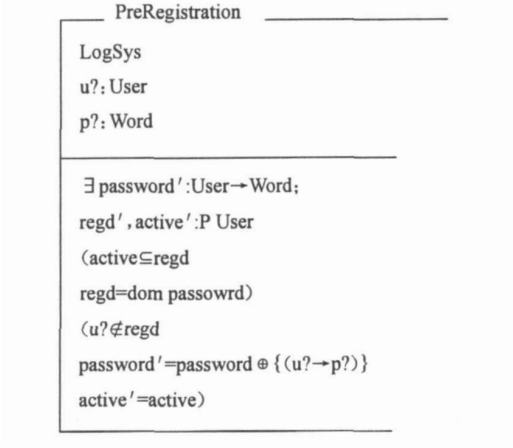
⊢ Φ∈ P User∧ Φ∈ P User∧ Φ∈ User → word∧ Φ∩ Φ = Φ∧ Φ = domΦ

利用以上规则 1– 规则 6, 可得到, Φ∈ P User, Φ∈ User → password, domΦ = Φ

由此可知, 初始状态存在.

3. 3　前置条件的验证

对于前置条件验证, 我们必须先求出相应的操作前置条件, 再与需求规格相比较. 如果相符, 则此操作形式规格说明合理, 否则不符合相应的需求. 对于用户注册模式 Registration, 推导一个操作的前置条件至关重要, 其方法是: 从描述操作模式的说明部分中删除后状态变量和输出变量, 然后在谓词部分多这些变量用存在量词量化, 于是可得到前置条件 PreRegistration模式.



展开 PreRegistration中的谓词, 可得到:

∃ password′: User → Word; regd′, active′: P User • (active′⊆ regd∧ regd′ = dom password) ∧ (u? ∉ regd∧ password′ = password Ý {(u? → p?)}) ∧ active′ = active

利用点规则, 并去除沉余部分, 上式可简化为

active⊆ P User∧ regd = dom password∧ u? ∉ regd∧ active = active

active⊆ P User和 regd = dom password在 LogSys说明中都为真, 且 active前后状态未发生变化, 故可得到简化后的前置条件模式谓词为: u? ∉ regd, 即 PreRegistration模式中要成功操作的前提条件是必须注册的用户不是已注册的用户, 符合直观认定.

其它的操作模式也可以类似地讨论证明之.

4　结束语

本文用形式化描述语言 Z语言对互联网登陆系统主要操作进行了形式化规格, 列出了五种主要操作模式, 并对其规格说明进行形式化验证, 证明了该系统能够符合设计要求, 而且通过形式化方法的设计, 提高了软件的可靠性, 使得软件的设计上更加规范统一的标准. 同时系统性质定理证明以及软件的求精还需进一步研究和探讨.

参考文献:

[1] 古天龙. 软件开发形式化方法 [M]. 北京: 高等教育出版社, 2005.

[2] Stephenne A, Champacgne B. Cepstral prefiltering for time delay estimation in reverberant environment[C] // proc of IEEE ICASSP'95 Detroit [s. n.], 1995: 3055– 3058.

[3] 崔　继, 周竹荣. 基于 Z规格的答疑系统标准检查 [J]. 西南大学学报 (自然科学版), 2007, 29(11): 133– 137.

[4] Luck M, D'Inverno M. A conceptual framework for agent definition and development[J]. The Computer Journal, 2001, 44(1): 1– 20.

[5] Francis Fung, Damir Jamsek. Formal Specification of Flight Guidance System (1998) [EB/OL]. http: //citeseer ist psu. edu/fung98formal html, 2005.

[6] Miao Yongwu. Design and implementation of a collaborative virtual problem– based learning environment[D]. Darmstadt technische university darmstadt, 2000: 58– 95.

[7] Meisels Irwin, Saaltink Mark. The Z/EVES reference manual[EB/OL]. https: //www. se. auckland. ac. nz/courses/SOFTENG460 /resources/zeves/Reference-Manual. pdf

[8] 缪淮扣, 李　刚, 朱关铭. 软件工程语言 – Z[M]. 上海: 上海科学技术文献出版社, 1994.

[9] 夏建勋, 唐红武. 需求分析的 Z语言形式化方法 [J]. 科学技术与工程, 2008, 8(8): 2246– 2248.