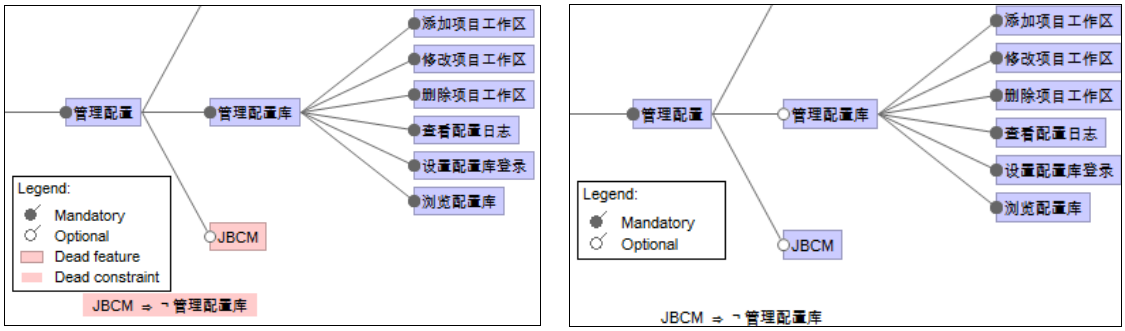
4.4 消解冲突

在发生配置约束关系变化的特征之间检测配置冲突,只有一个 Type II 类型的冲突被发现并被消解.图 9 所示例子即来源于此.

我们利用两个现有的工具 FeatureIDE[34]和 RequiLine[35]验证冲突消解的效果,这两个工具都可以对特征模型进行图形化编辑,检查模型是否存在配置冲突.我们分别将冲突消解前后的特征模型输入到工具中.图 10 显示了在 FeatureIDE 中消解冲突前后的特征模型片段.在冲突消解前,可选特征“JBCM”的插入导致其与一个完全必选特征“管理配置库”互斥.在图 10(a)中,FeatureIDE 识别出该冲突.经过冲突消解后,图 10(b)显示,该冲突已被移除.最后对特征模型所有特征的配置约束关系检查,不存在配置约束冲突.



(a) 冲突消解前 (b) 冲突消解后

Fig.10 Screenshots before and after conflict resolution in FeatureIDE

图 10 冲突消解前后的 FeatureIDE 截图

4.5 扩展效果分析

演化分析完成后,我们统计了每个分析阶段特征模型的变化,进一步分析扩展特征关联关系对演化分析的影响和效果.表 10 显示了每个分析阶段后特征模型中每种关联关系相关特征的数量,多数的变化都发生在前两个阶段,尤其是必选、可选、单选和相似关系特征的数量.由于输入的特征变更多以添加和删除类型为主,定位这些变更增加了可选、单选和相似特征,同时削减了的输入模型中的必选特征.而在共性提取和模型修剪中,对相似特征的重构增加必选特征的数量,并消除了这些特征间的相似性,减低了相似特征的数量.

Table 10 Statistics of the features according to different dependency types during the evolution analysis

表 10 演化分析中不同类型关联关系相关特征统计结果

关联关系	输入	phase 1	phase 2	phase 3
必选	345	290	345	345
可选	52	84	82	84
单选	14	134	140	140
多选	168	179	178	178
依赖	65	81	81	81
互斥	12	20	14	14
前置	130	146	146	146
限制	92	95	95	95
相似	100	226	116	116

为了验证模型扩展对演化分析的影响和效果,我们分别在扩展特征关联关系前后的特征模型基础上运行分析方法,并统计了所有分析阶段取得的配置约束关系相关的演化操作,如图 11 所示.对比两种场景可观察到:在多数演化操作类型上,扩展后分析取得的操作数量都大于扩展前.尤其在 InsMan,InsAlt,DelAlt 和 MovToAlt 数量上,前者显著大于后者.二者之间的差距主要由于扩展特征关联关系增强了“涟漪”效应的分析,从而发现了 17 个修改类型的特征变更.定位这些特征变更,使得更多的特征被插入到或者移动到单选特征组中.这些操作也

为共性提取提供了更多候选特征,使得更多的特征被重构,提高了必选特征的数量.

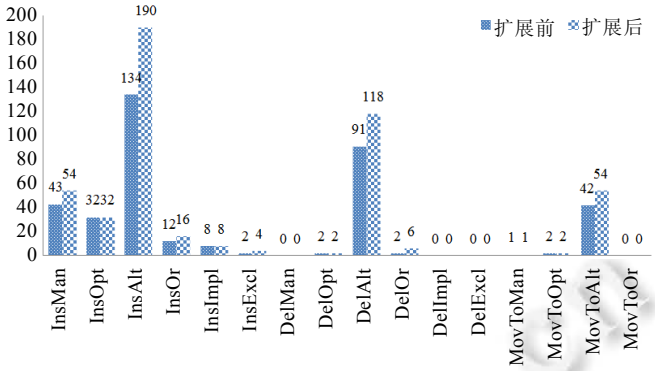


Fig.11 Model operations obtained by the evolution analysis based on the feature models before and after extending the feature dependency types

图 11 在扩展特征关联关系前后的特征模型基础上进行演化分析得到的模型操作

上述分析结果表明了扩展特征关联关系对演化分析的影响,同时也验证了扩展对于演化分析的必要性.即,在扩展基础上分析特征模型演化,可得到更为全面的演化操作.

5 讨 论

尽管案例分析验证了方法的可用性和有效性,但是该方法在某些方面仍然存在局限性.此外,在案例分析过程中,通过和项目人员的讨论,我们也得到一些经验.

5.1 方法的局限性

方法的局限性主要体现在 3 个方面:扩展特征关联关系、“涟漪”效应分析和冲突消解策略.

(1) 扩展特征关联关系与特征配置约束关系的相关性

对比配置约束关系和扩展关联关系,可以发现依赖与前置或者限制之间可存在一定的相关性.对于具有前置或者限制关系的特征,其可能在配置上也存在依赖关系.例如在图 3(a)中,完全必选特征“设置数据库”前置于可选特征“备份数据”,“备份数据”也依赖于“设置数据库”.而在一些研究[9,34]中,若可选特征依赖于完全必选特征,则该依赖关系被视为冗余,需在特征模型中消除.因此,图 3(a)中并没有显示二者间的依赖关系.但是,如果在“涟漪”效应分析只采用依赖关系,某些特征变更将无法被发现.例如,修改“设置数据库”增加系统支持的数据库类型,可引起“备份数据”的变更,如果不补充前置关系,则无法识别该类变更传播.因此,补充前置和限制关系对于“涟漪”效应分析是必要的.

(2) “涟漪”效应分析

由于特征是系统的高层抽象,其变更带来的“涟漪”效应具有不确定性[6,29,30].虽然我们设计了规则发现可能受影响的特征和特征变更的类型,但是为了提高分析的准确性,我们建议在利用规则分析后,将分析结果提交给分析人员,通过人工检查的方式保证结果的准确性.

此外,在分析“涟漪”效应时,我们只考虑从扩展的 3 种关联关系出发.实际上,父子特征之间也可能存在变更的传播.由于特征模型的树形结构,父子特征间的变更相比扩展关联关系上的变更传播更为复杂.修改模型高层的特征,可能导致其所有后继特征发生变化.本文中不考虑父子之间的变更传播,如有父特征变更影响子特征的场景,需在输入中逐一为受影响的子特征定义特征变更请求.

(3) 冲突消解策略

我们根据不同的配置冲突类型设计了消解策略,除了这些策略以外,还可能存在其他的选择,例如,将 Type I 中的非完全必选特征更新为完全必选特征.这样的操作虽然解决了配置冲突,但是该特征并不是所有产品都需

要的,违背其实际的配置要求.因此,我们并未将类似的策略纳入方法中.

在消解配置冲突时,我们为分析人员提供了多种消解策略.最终选择哪一种策略,则需要分析人员根据实际情况做出决策.由于执行不同消解策略花费的工作量不同,对系统和定制产品造成的影响也不尽相同.因此,分析人员可通过收集影响因素建立统一的评价方法,对每一种消解策略进行影响分析,从而选择合适的策略消解冲突.

5.2 案例分析经验

进行案例分析后,我们与 Qone 项目组探讨方法的使用和应用结果时,得到一些意见和建议.综合这些意见和建议,我们得到如下几点经验:

(1) 特征重构的价值分析

解析和重构相似特征是共性提取和模型修剪的重要步骤,在案例分析中,约有 126 个特征需要进行解析和重构.由于执行这些分析需要花费一定的工作量,对于定制规模较小的软件而言,重构虽然提升了复用,但是重构带来的开销可能高于收益.对于一些需要快速交付的产品,先重构后开发的方式也可能延迟交付时间.因此,在进行特征解析和重构时需要进行价值分析,平衡解析和重构的开销和收益.

(2) 在分析前对特征变更聚类

由于领域内包含众多的定制产品,可能出现对同一特征不同语义格的变更请求或者同一语义格的不同请求.针对此类变更请求,项目人员建议我们在演化分析前对变更进行聚类,从而提高分析的效率.

6 相关工作

本文的相关工作主要涉及两个方面:特征关联关系的扩展和特征模型的演化分析.

(1) 特征关联关系的扩展

特征模型最早由 Kang 在 1998 年提出[3],作为面向特征的领域分析(feature oriented domain analysis)方法的核心.由于特征建模具有简单、灵活、易理解等优势,使其一经提出就吸引了众多的研究人员,并逐步成为重要的共性和可变性建模方法之一.此后,很多研究在该模型的基础上进行扩展,以满足不同的应用场景.综合特征模型分析的系统化调研[18]和特征模型扩展工作的总结[2,36],可将扩展归纳为 3 个方面:特征的属性、特征的基数和特征的关联关系.

本文所提出的扩展从属于特征关联关系方面,因此,本节主要总结该方面的研究,并进行对比.

特征关联关系的扩展工作中,Lee 等人根据特征模型在设计可适应(adaptable)和可配置(configurable)资产时存在的缺陷,从特征操作的角度引入了多种关联关系[19].补充的特征关联关系包括使用(usage)、修改(modification)和激活(activation).Zhang 等人面向特征分析了需求间的关联关系,增加了静态和动态特征关联关系[20].其中,静态关联关系包括精化(refinement)和限制(constraint),动态特征关联关系主要为影响(influence)和交互(interaction).Peng 等人对特征模型进行了更严格地定义,给出了 3 种特征关联关系:使用(usage)、决定(decide)和配置依赖(configdepend)[21].通过这 3 种关联关系,更为细致地描述运行时和绑定时(binding)特征间的交互性及配置约束.

上述研究工作进一步增强了特征模型对特征相关性的表达能力,然而对于“涟漪”效应分析而言,关键是识别不同层次和类型关联关系上的特征变更传播.这些扩展只是对特征在配置和重用上相关性的进一步刻画,并没有从其他角度进行补充.将这些扩展得到关联关系与已有变更传播研究[24]对比可发现,采用这些扩展的特征模型仍然不足以支撑“涟漪”效应的分析.与这些研究不同,本文所提出的扩展主要面向特征变更的“涟漪”效应,分析特征间不同抽象层次或者不同方面的关联关系,从中选择可用于“涟漪”效应的关联关系类型补充到特征模型中,实现了在特征模型基础上分析特征变更“涟漪”效应.此外,扩展的关联关系还可用于识别潜在的产品共性,为共性提取和模型修剪提供更多的候选特征.

(2) 特征模型的演化分析

特征模型演化分析的研究主要包括演化影响分析、重构和模型诊断等多个方面.

在演化影响分析方面:Riebisch 主要以特征模型为桥梁建立需求与其他软件制品的跟踪关系,通过需求的变更分析演化对特征模型以及其他制品造成的影响[11];Peng 等人分析了影响特征模型演化的因素,识别可能发生的模型变化和受影响的特征,并对模型变化进行价值分析和排序[12].这些方法主要从需求工程的角度分析特征模型演化.

其他方面,Botterweck 等人归纳了特征模型演化的元操作集合 EvoOperators[25];Hwan 等人提出多种特征模型的操作,例如删除节点、删除子树[26];Thum 等人引入了重构、归纳(generalization)、特化(specialization)和随意性编辑(arbitrary edit)描述演化前后特征模型的变化[37];Alves 等人基于特征模型提出了产品线重构模式[38].还有许多研究人员采用 SAT(satisfiability solving),BDD(binary decision diagram)和 CSP(constraint satisfaction problem)对模型的有效性进行诊断[15,16,39].Guo 等人提出在演化时通过设定策略修复不一致的特征模型配置[10].

上述研究中,其中一部分以需求规约或者需求模型分析特征模型的演化.这些研究只是在需求层面上进行演化分析,通过特征模型表达共性和可变性的变化,并非是从特征模型本身的角度考虑演化.由于需求模型缺乏对特征和特征关联关系的描述,分析过程中可能忽略特征变更的“涟漪”效应,无法发现由此引发的共性和可变性变化.与这些工作不同的是,本文通过扩展特征关联关系实现了在特征模型基础上进行特征变更的“涟漪”效应分析,并通过共性提取和模型修剪发现更多的产品共性.此外,本文提出的方法还考虑了模型配置冲突的发现和消解.与上述研究中的模型诊断的研究相比,本文仅从发生配置约束变化的特征出发检测配置冲突.与 Guo 等人的研究类似,本文同样假设输入的特征模型不存在任何异常,配置冲突只可由模型演化操作引发,并不需要遍历整个特征模型进行检测.但是对于冲突消解,上述工作并未说明如何消解不同类型的配置冲突,Guo 等人的研究也仅描述如何修复删除特征造成的配置异常.本文则定义了详细的策略应对每一种配置冲突.

由于识别共性和可变性的变化是特征模型演化分析重要组成,因此我们在研究过程中也参考了共性和可变性分析的相关工作.分析相似特征时,采用 Guo 等人分析功能性需求的方法[33],通过“格语法”对特征进行语义分解.在对可选特征进行共性分析时,我们总结了相关工作中分析产品共性的决策因素[7,17,33,40],在这些因素基础上设计了规则辅助开发人员决策.

7 总 结

本文在分析当前特征模型演化分析的挑战和现有方法存在的问题基础上,提出了一种软件特征模型演化分析方法.该方法扩展了特征关联关系和模型演化元操作,通过这些扩展,支持在特征模型基础上分析和识别特征变更的“涟漪”效应引发的特征变更,发现和提取潜在的产品共性.同时,该方法还对演化过程中出现的配置冲突进行消解.为了验证该方法的可用性和有效性,实现了半自动化的分析工具,并选取了一个具有代表性的软件系统 Qone 进行案例分析.分析结果表明,我们所提出的方法可以用于实际的项目并辅助项目人员进行特征模型演化分析.

在研究过程中,我们也发现了该方法存在的不足,并在案例分析中收获了一些经验教训.这些都为我们指明了下一步的研究方向,包括:对特征变更进行聚类分析,提高方法的效率;对特征解析和重构进行价值分析;建立冲突消解策略选择方法;评价特征模型演化对软件开发的影响.此外,我们也将持续关注 Qone 项目,并计划采用更多的方式验证方法的有效性.同时,也希望在更多的软件系统和业务场景中进行应用,不断改进方法.

References:

[1] Apel S, Kastner C. An overview of feature-oriented software development. Journal of Object Technology, 2009,8(4):1−36. [doi: 10.5381/jot.2009.8.5.c5]

[2] Zhang W, Mei H. Feature-Oriented software reuse technology—State of the art. Chinese Science Bulletin, 2014,59(1):21−42 (in Chinese with English abstract).

[3] Kang K, Cohen S, Hess J, Novak W, Peterson A. Feature-Oriented domain analysis (FODA) feasibility study. Technical Report, CMU/SEI-90-TR-21, Software Institute, Carnegie Mellon University, 1990.

[4] Czarnecki K, Helsen S, Eisenecker U. Staged configuration using feature models. In: Proc. of the 3rd Int'l Conf. on Software Product Lines. Berlin: Springer-Verlag, 2004. 266−283. [doi: 10.1007/978-3-540-28630-1_17]

[5] Yau SS, Collofello JS, MacGregor T. Ripple effect analysis of software maintenance. In: Proc. of the 2nd Int'l Conf. on Computer Software and Application. Los Alamitos: IEEE Computer Society, 1978. 60−65. [doi: 10.1109/CMPSAC.1978.810308]

[6] Zhang L, Qian GQ, Li L. Software stability analysis based on change impact simulation. Chinese Journal of Computers, 2010,33(3): 440−451 (in Chinese with English abstract). [doi: 10.3724/SP.J.1016.2010.00440]

[7] Pohl K, Bockle G, Van Der Linden F. Software Product Line Engineering: Foundations, Principles, and Techniques. Berlin: Springer-Verlag, 2005. [doi: 10.1007/3-540-28901-1]

[8] Ardis M, Weiss DM. Defining families: The commonality analysis. In: Proc. of the 21st Int'l Conf. on Software Engineering. Los Alamitos: IEEE Computer Society, 1999. 671−672. [doi: 10.1145/302405.302929]

[9] Maben T, Lichter H. Deficiencies in feature models. In: Proc. of the Workshop on Software Variability Management for Product Derivation-Towards Tool Support. Berlin: Springer-Verlag, 2004.

[10] Guo JM, Wang YY, Trinidad P, Benavides D. Consistency maintenance for evolving feature models. Expert Systems with Applications, 2012,39(5):4987−4998. [doi: 10.1016/j.eswa.2011.10.014]

[11] Riebisch M. Supporting evolutionary development by feature models and traceability links. In: Proc. of the 11th IEEE Int'l Conf. and Workshop on the Engineering of Computer-Based Systems. Los Alamitos: IEEE Computer Society, 2004. 370−377. [doi: 10.1109/ECBS.2004.1316721]

[12] Peng X, Yu YJ, Zhao WY. Analyzing evolution of variability in a software product line: From contexts and requirements to features. Journal of Information of Software Technology, 2011,53(7):707−721. [doi: 10.1016/j.infsof.2011.01.001]

[13] Asadi M, Bagheri E, Mohabbati B, Gasevic D. Requirements engineering in feature oriented software product lines: An initial analytical study. In: Proc. of the 16th Int'l Conf. on Software Product Line. New York: ACM Press, 2012. 36−44. [doi: 10.1145/2364412.2364419]

[14] Buhne S, Lauenroth K, Pohl K. Why is it not sufficient to model requirements variability with feature models. In: Proc. of the Workshop on Automotive Requirements Engineering. 2004. 5−12.

[15] White J, Benavides D, Schmidt CD, Trinidad P, Dougherty B, Ruiz-Cortes A. Automated diagnosis of feature model configurations. Journal of Systems and Software, 2010,83(7):1094−1107. [doi: 10.1016/j.jss.2010.02.017]

[16] Benavides D, Trinidad P, Ruiz-Cortes A. Automated reasoning on feature models. In: Proc. of the 17th Int'l Conf. on Advanced Information Systems Engineering. Berlin: Springer-Verlag, 2005. 491−503. [doi: 10.1007/11431855_34]

[17] Chen K, Zhang W, Zhao HY, Mei H. An approach to constructing feature models based on requirements clustering. In: Proc. of the 13th Int'l Conf. on Requirements Engineering. Los Alamitos: IEEE Computer Society, 2005. 31−40. [doi: 10.1109/RE.2005.9]

[18] Benavides D, Segura S, Ruiz-Cortes A. Automated analysis of feature models 20 years later: A literature review. Information Systems, 2010,35(6):615−636. [doi: 10.1016/j.is.2010.01.001]

[19] Lee K, Kang KC. Feature dependency analysis for product line component design. In: Proc. of the 8th Int'l Conf. on Software Reuse. Berlin: Springer-Verlag, 2004. 69−85. [doi: 10.1007/978-3-540-27799-6_7]

[20] Zhang W, Mei H. Feature-Driven requirement dependency analysis and high-level software design. Requirements Engineering, 2006,3(11):205−220. [doi: 10.1007/s00766-006-0033-x]

[21] Peng X, Zhao WY, Xue YJ, Wu YJ. Ontology-Based feature modeling and application-oriented tailoring. In: Proc. of the 9th Int'l Conf. on Software Reuse. Berlin: Springer-Verlag, 2006. 87−100. [doi: 10.1007/11763864_7]

[22] Dahlstedt AG, Persson A. Requirements interdependencies: State of the art and future challenges. In: Proc. of the Engineering and Managing Software Requirements. Berlin: Springer-Verlag, 2005. 95−116. [doi: 10.1007/3-540-28244-0_5]

[23] Pohl K. Process-Centered Requirements Engineering. New York: John Wiley & Sons, Inc., 1997.

[24] Zhang H, Li J, Zhu LM, Jeffery R, Liu Y, Wang Q, Li MS. Investigating dependencies in software requirements for change propagation analysis. Journal of Information and Software Technology, 2014,56(1):40−53. [doi: 10.1016/j.infsof.2013.07.001]

[25] Botterweck G, Pleuss A, Dhungana D, Polzer A, Kowalewski S. Evofm: Feature-Driven planning of product-line evolution. In: Proc. of the ICSE Workshop on Product Line Approaches in Software Engineering. New York: ACM Press, 2010. 24−31. [doi: 10.1145/1808937.1808941]

[26] Hwan C, Kim P, Czarnecki K. Synchronizing cardinality-based feature models and their specializations. In: Proc. of the 1st European Conf. on Model Driven Architecture—Foundations and Applications. Berlin: Springer-Verlag, 2005. 331−348. [doi: 10.1007/11581741_24]

[27] Lehnert S. A taxonomy for software change impact analysis. In: Proc. of the 12th Int'l Workshop on Principles of Software Evolution and the 7th Annual ERCIM Workshop on Software Evolution. New York: ACM Press, 2011. 41−50. [doi: 10.1145/2024445.2024454]

[28] Li Y, Li J, Yang Y, Li MS. Requirement-Centric traceability for change impact analysis: A case study. In: Proc. of the Int'l Conf. on Software Process. Berlin: Springer-Verlag, 2008. 100−111. [doi: 10.1007/978-3-540-79588-9_10]

[29] Lock S, Kotonya G. An integrated probabilistic framework for requirement change impact analysis. Australasian Journal of Information Systems, 2007,6(2):38−63.

[30] Fu Y, Li MQ, Chen FZ. Impact propagation and risk assessment of requirement changes for software development projects based on design structure matrix. Int'l Journal of Project Management, 2012,30(3):363−373. [doi: 10.1016/j.ijproman.2011.08.004]

[31] Fillmore CJ. The case for case. In: Universals in Linguistic Theory. Rinehart & Winston, 1968. http://www.researchgate.net/publication/230875990_The_Case_for_Case

[32] Niu N, Easterbrook S. Extracting and modeling product line functional requirements. In: Proc. of the 16th Int'l Conf. on Requirements Engineering. Los Alamitos: IEEE Computer Society, 2008. 155−164. [doi: 10.1109/RE.2008.49]

[33] Guo JM, Wang YL, Zhang Z, Nummenmaa J, Niu N. Model driven approach to developing domain functional requirements in software product lines. IET Software, 2012,6(4):391−401. [doi: 10.1049/iet-sen.2010.0072]

[34] Thum T, Kastner C, Benduhn F, Meinicke J, Saake G, Leich T. FeatureIDE: An extensible framework for feature-oriented software development. Experimental Software and Toolkits, 2014,79(1):70−85. [doi: 10.1016/j.scico.2012.06.002]

[35] Massen T, Lichter H. RequiLine: A requirements engineering tool for software product lines. In: Proc. of the Workshop on Software Product-Family Engineering. Berlin: Springer-Verlag, 2004. 168−180. [doi: 10.1007/978-3-540-24667-1_13]

[36] Nie KM, Zhang L, Fan ZQ. Systematic literarture review of software product line variability modeling techniques. Ruan Jian Xue Bao/Journal of Software, 2013,24(9):2001−2019 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/4433.htm [doi: 10.3724/SP.J.1001.2013.04433]

[37] Thum T, Batory D, Kastner C. Reasoning about edits to feature models. In: Proc. of the 31st Int'l Conf. on Software Engineering. Los Alamitos: IEEE Computer Society, 2009. 254−264. [doi: 10.1109/ICSE.2009.5070526]

[38] Alves V, Gheyi R, Massoni T, Kulesza U, Borba P, Lucena C. Refactoring product lines. In: Proc. of the 5th Int'l Conf. on Generative Programming and Component Engineering. New York: ACM Press, 2006. 201−210. [doi: 10.1145/1173706.1173737]

[39] Czarnecki K, Wasowski A. Feature diagrams and logics: There and back again. In: Proc. of the 11th Int'l Conf. on Software Product Line. Alamitos: IEEE Computer Society, 2007. 23−34. [doi: 10.1109/SPLINE.2007.24]

[40] Moon M, Yeom K, Chae HS. An approach to developing domain requirements as a core asset based on commonality and variability analysis in a product line. IEEE Trans. on Software Engineering, 2005,31(7):551−569. [doi: 10.1109/TSE.2005.76]

附中文参考文献:

[2] 张伟,梅宏.面向特征的软件复用技术——发展与现状.科学通报,2014,59(1):21−42. [doi: 10.1360/972013-341]

[6] 张莉,钱冠群,李琳.基于变更传播仿真的软件稳定性法分析.计算机学报,2010.33(3):440−451. [doi: 10.3724/SP.J.1016.2010.00440]

[36] 聂坤明,张莉,樊志强.软件产品线可变性建模技术系统综述.软件学报,2013,24(9):2001−2019. http://www.jos.org.cn/1000-9825/4433.htm [doi: 10.3724/SP.J.1001.2013.04433]

胡洁(1982－),女,安徽芜湖人,博士,主要研究领域为软件定制化开发,软件需求演化.

王青(1964－),女,博士,研究员,博士生导师,CCF 高级会员,主要研究领域为软件质量管理,过程建模,知识管理,需求管理,软件协同工作.