一种基于层次聚类的软件架构恢复方法

李 寒[1,2] 佟 宁[3] 陈 峰[4]
(北方工业大学计算机学院 北京 100144)[1]
(大规模流数据集成与分析技术北京市重点实验室 北京 100144)[2]
(大连交通大学软件学院 大连 116052)[3] (德蒙特福德大学计算机与信息工程学院 莱斯特 LE1 9BH)[4]

摘 要 针对软件聚类侧重相似度测度而欠缺考虑实体和特征的特性的问题,提出一种基于层次聚类的软件架构恢复方法(HCSAR)。该方法有针对性地选取实体和特征,提出特征的多重加权策略,采用信息丢失度作为相似度测度,选取和设计软件聚类的客观和主观评估准则。与目前效果较好的软件聚类方法相比,HCSAR在聚类中期能生成更多的簇,主观判定数更低,能够通过调整关注点获得不同的聚类结果,使用设计的评估准则分析聚类结果还能有效辅助系统划分。

关键词 层次聚类,架构恢复,面向对象,面向过程,系统划分
中图法分类号 TP301 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2017.04.016

Hierarchical Clustering Based Software Architecture Recovery Approach

LI Han[1,2] TONG Ning[3] CHEN Feng[4]
(College of Computer Science, North China University of Technology, Beijing 100144, China)[1]
(Beijing Key Laboratory on Integration and Analysis of Large-scale Stream Data, Beijing 100144, China)[2]
(Software Technology Institute, Dalian Jiaotong University, Dalian 116052, China)[3]
(School of Computer Science and Informatics, De Montfort University, Leicester LE1 9BH, UK)[4]

Abstract To solve the problem of the lack of consideration of the characteristics of entities and features, a hierarchical clustering based software architecture recovery approach was proposed. Targeted entities and features were selected, weight scheme was proposed to generate feature vectors, information loss was considered as the similarity metric, and objective and subjective evaluation measures were respectively chosen and given. Compared with superior agglomerative software clustering approaches, HCSAR is more cohesive, requires less arbitrary decisions, is flexible to adjust the focus of software clustering, and is able to assist to generate more accurate system partition.

Keywords Hierarchical clustering, Architecture recovery, Object orientation, Procedure orientation, System partition

1 引言

软件架构恢复是通过系统的低层次描述提取高层次信息的过程[1]。近年来,遗传算法[2]、模式匹配[3]、聚类等许多技术被应用于软件架构恢复。其中,聚类被视为准自动方法[4],称为软件聚类。层次聚类是软件聚类的关键技术,尤其以凝聚式层次聚类最受认可[5]。近年来,许多凝聚式层次聚类算法被应用于软件架构恢复。单连接聚类方法(SLA)、全连接聚类方法(CLA)、平均连接聚类方法(ULA)和加权平均连接聚类方法(WLA)是经典的软件聚类方法,主要存在主观判定数高的问题[6]。针对该问题,出现了两段式的聚类算法。联合聚类方法(CA)为两阶段式聚类,其通过保留更多的信息来减少主观判定数,能够提高聚类质量,但聚集算法未利用簇与实体之间的定量信息[7]。Maqbool和Barbi在CA方法的基础上提出了WCA方法[8]。WCA将特征向量加权,并采用非二进制系数计算实体相似度。其比CA,SLA,CLA,WLA和ULA的内聚度更高,聚类结果更准确。此后,Andritsos和Tzerpos在凝聚式信息瓶颈AIB方法的基础上提出了LIMBO方法[9]。LIMBO采用信息丢失度度量实体之间的相似度,支持不同的特征加权策略,适用于形式化描述特征和非形式化描述特征。与SLA,CLA,WLA,ULA相比,其聚类结果等同或优于其他方法;该方法在保留的特征信息、主观判定数、可扩展性方面均具有优势。

综上所述,软件聚类的研究较少关注实体和特征的特性[4]。针对该问题,本文充分考虑实体和特征的特性在改善聚类质量、增强聚类适应性和辅助系统划分方面的作用,提出

到稿日期:2015-11-30 返修日期:2016-02-28 本文受北京市教委科技计划面上项目(KM2015_10009007),北京市优秀人才培养资助青年骨干个人项目(2014000020124G011),北方工业大学科研启动基金项目资助。

李 寒(1981—),女,博士,助理研究员,主要研究方向为软件演化、软件体系结构、云计算,E-mail:lihan@ncut.edu.cn;佟 宁(1981—),女,博士,讲师,主要研究方向为无线自组网、软件工程;陈 峰(1969—),男,博士,副教授,主要研究方向为软件工程、分布式计算。

一种基于层次聚类的软件架构恢复方法。

2 基于层次聚类的软件架构恢复方法

2.1 实体和特征的选取

文献[5]发现全面且恰当的特征会获得理想的聚类结果。本文将函数(过程)和类分别定义为面向过程和面向对象软件系统的实体,以基于间接关联的聚集策略和实体功能相关性为基础,定义如下特征选取原则。

(1)数据是软件系统的核心,与功能相关。若两个实体使用相同的数据,则它们具有相关功能,可能属于同一子系统。因此,与实体相关的数据被选为特征。

(2)实体之间的关联也是决定系统划分的重要因素。若两个实体都与相同的实体具有关联,则它们具有相关功能,可能属于同一子系统。因此,与实体相关的实体被选为特征。

遵循上述选取原则,实体引用的用户自定义数据类型、数据文件、局部变量和全局变量被定义为软件系统的共有特征;头文件、宏和实体间的调用关系被定义为面向过程的软件系统的特征;头文件或包、实现的接口、实体间的继承关系、实体间的依赖关系和实体间的关联关系被定义为面向对象的软件系统的特征。本文选取的特征包含常用的数据和实体,也可根据实际系统对特征进行删减和扩展,且可使用 EA(Enterprise Architect)等反向工具获取系统 UML 图,导出 XMI 文档,再采用 XSLT 提取实体及特征信息。

2.2 实体的特征向量

本文将实体的特征分类,特征向量由各类特征的子向量构成。令 V_i 为实体 S_i 的特征向量,式(1)中,V_i^j 为第 i 个实体的第 j 类特征子向量,包含 m_j 个特征;v_{ik}^j 为 V_i^j 的第 k 个特征,$v_{ik}^j \in [0,+\infty)$。

$$\begin{aligned} \boldsymbol{V}_i &= (\boldsymbol{V}_i^1, \boldsymbol{V}_i^2, \cdots, \boldsymbol{V}_i^j, \cdots, \boldsymbol{V}_i^n) \\ \boldsymbol{V}_i^j &= (v_{i1}^j, v_{i2}^j, \cdots, v_{ik}^j, \cdots, v_{im_j}^j) \end{aligned} \tag{1}$$

不同的特征不仅在聚类过程中呈现出不同的重要程度[5],而且与实体的关联程度也存在差异。针对上述特点,本文首次提出为实体特征设置如下 3 类特征权重。

(1)类别差异权重:不同类别的特征展现不同的信息,不同的信息与实体的相关程度存在差异。此外,用户对不同特征类别的关注程度不同。因此,应为特征类别添加权量,以区别特征类别的重要性程度,并将主观因素引入软件架构恢复中。令 wg 表示类别差异权重,增设类别差异权重的实体特征向量如式(2)所示。

$$\boldsymbol{V}_i = (wg_1 \times \boldsymbol{V}_i^1, wg_2 \times \boldsymbol{V}_i^2, \cdots, wg_j \times \boldsymbol{V}_i^j, \cdots, wg_n \times \boldsymbol{V}_i^n) \tag{2}$$

(2)特征依赖度权重:特征被实体引用的次数越多,实体对其依赖性往往越强。将特征被当前实体以某类型使用的次数与当前实体使用特征的总次数的比值作为特征依赖度权重。v_{ik}^j 的特征依赖度权重为 wd_{ik}^j,增设特征依赖度权重的特征如式(3)所示,$N_{v_{ik}^j}$ 为 S_i 的第 j 类第 k 个特征被使用的次数,m_s 为第 s 类特征的特征数。

$$v_{ik}^j = wd_{ik}^j \times v_{ik}^j = \frac{N_{v_{ik}^j}}{\sum_{t=1}^{s}\sum_{p=1}^{m_s} N_{v_{ip}^t}} \times v_{ik}^j \tag{3}$$

(3)特殊关注点权重:某些情况下,具有特殊含义的特征虽然被引用的次数较少,软件聚类对其依赖程度却很大。例如,尽管某机密数据被引用次数较少,但与其相关的实体具有较高的功能相关性。令 ws_{ik}^j 为实体 S_i 第 j 类特征中第 k 个特征的特殊关注点权重,增设特殊关注点权重的实体特征如式(4)所示。

$$v_{ik}^j = wd_{ik}^j \times ws_{ik}^j \times v_{ik}^j = \frac{N_{v_{ik}^j}}{\sum_{t=1}^{s}\sum_{p=1}^{m_s} N_{v_{ip}^t}} \times ws_{ik}^j \times v_{ik}^j \tag{4}$$

为避免产生冗余信息,在构造的最终特征向量中将某特征所有特征值的和作为其特征值。

2.3 相似度测度的选取

研究表明基于信息论的软件聚类方法的适用性更强[9],本文采用适用性更强的基于信息论的方法作为相似度测度,如式(5)所示。令实体 S_i 和 S_j 聚集的簇为 S_{ij},实体被选择的概率 $p(S_i)=p(S_j)=1/n$,$p(S_{ij})=p(S_i)+p(S_j)$,n 为实体总数,m 为特征总数。依据信息论,实体的聚合会损失信息,因此合并信息损失最小的实体。

$$\begin{aligned} &\delta I(S_i,S_j) = [p(S_i) + p(S_j)] \times Djs[\boldsymbol{V}_i, \boldsymbol{V}_j] \\ &Djs[\boldsymbol{V}_i,\boldsymbol{V}_j] = \frac{p(S_i)}{p(S_{ij})} \times Dkl[\boldsymbol{V}_i \parallel \boldsymbol{V}_{ij}] + \frac{p(S_j)}{p(S_{ij})} \times \\ &\qquad Dkl[\boldsymbol{V}_j \parallel \boldsymbol{V}_{ij}] \\ &Dkl[\boldsymbol{V}_i \parallel \boldsymbol{V}_j] = \sum_{k=1}^{m} v_i^k \log \frac{v_i^k}{v_j^k} \end{aligned} \tag{5}$$

综上,HCSAR 包含选取实体特征、构造特征向量和确定相似度测度的内容,由生成实体、计算实体相似度、生成簇和判定终止条件 4 个步骤构成,后 3 个步骤迭代至达到预期的聚类数。

3 实验与结果分析

本文以基于 Java 的开源 WAP 浏览器 j2wap 和基于 C 的开源标准通用编码库 MPEG4 为实例,将 HCSAR 与目前效果较好的 WCA,WCAW 和 LIMBO[7] 进行对比以验证其有效性,WCAW 为采用本文加权策略的 WCA 方法。

3.1 评估准则

3.1.1 客观评估准则

(1)簇的数目:若聚类在中期生成大量规模较小且相似的簇,则聚类具有更高的内聚度,结果更加准确[8]。

(2)主观判定数:尽管主观判定能够解决具有相同相似度的实体或簇对的冲突,但它会降低软件聚类的效率,若不准确还会影响聚类的执行和质量。因此,应尽量降低主观判定数以提升聚类效率和准确度。

3.1.2 主观评估准则

本文以实体特征的冗余性、实体间的多重关联性[10]为依据,设计如下主观评估准则。

(1)簇的稳定度:当输入数据存在小规模差异时,聚类结果相对稳定是聚类的特性[9]。不同聚类结果的簇 C_i 和 C_j' 间的稳定度 $D_{Stability}(C_i,C_j')$ 被定义为属于 C_i 且属于 C_j' 的实体数与属于 C_i 的实体数的比值,如式(6)所示。将簇的稳定度应用于聚类结果中能发现相对稳定的簇对,辅助系统划分。

$$D_{stability}(C_i,C_j') = \frac{|C_i \cap C_j'|}{|C_i|} \tag{6}$$

(2)聚类相似度:专家给定的和聚类生成的系统划分分别侧重主观和客观情况,将其分别称为子系统和聚类簇,聚类相似度 $D_{Similiarity}$ 被定义为子系统与聚类簇的最大实体重叠率的均值,如式(7)所示,C_i 为子系统,C_i 与聚类簇 C_j' 具有最大的相交实体集,$MaxC_i$ 为最大相交实体集的实体数,n 为子系统数。

$$D_{Similiarity}=\frac{\sum_{i=1}^{n}\frac{MaxC_i}{|C_i|}}{n} \tag{7}$$

3.2 结果分析

(1)簇的数目

图 1 为 WCA,WCAW,LIMBO 和 HCSAR 聚集 j2wap 和 MPEG4 形成的簇的数目。

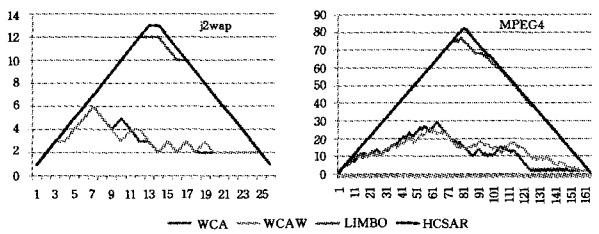

图 1　聚类过程中簇的数目

如图 1 所示,HCSAR 在 j2wap 和 MPEG4 聚类中期形成簇的最大数大约是 WCA 的两倍和 WCAW 的 3 倍,趋势与 LIMBO 一致,且其最大数高于 LIMBO 的最大数。结果表明 HCSAR 在聚类中期提取了更多的小规模簇,内聚度更高。

(2)主观判定数

将聚类过程平均分为 5 个阶段,图 2 示出了各阶段的主观判定数。WCA,WCAW,LIMBO 和 HCSAR 的主观判定总数依次降低。在第一阶段,4 种方法的主观判定数均较多,但采用本文加权策略的 WCAW 的主观判定数比 WCA 明显减少,HCSAR 的主观判定数比 LIMBO 明显减少。

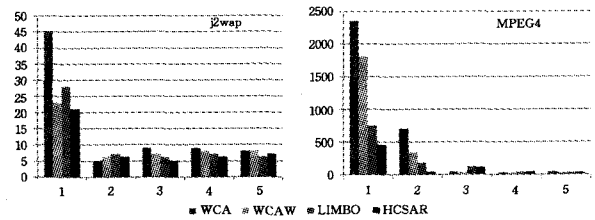

图 2　聚类各阶段的主观判定数

(3)簇的稳定度

图 3(a)和图 3(b)为当依赖关系类别差异权重为 1 和 10 时的 j2wap 聚类结果。26 个实体在聚类后期被划分为 3 个簇。(C_1,C_1'),(C_2,C_2'),(C_3,C_1')和(C_2,C_3') 的簇的稳定度较高,分别为 0.875,0.5,0.5 和 0.375。因此,稳定簇对为 $(C_1\cup C_3,C_1')$,$(C_2,C_2'\cup C_3')$,可作为 j2wap 的系统划分。

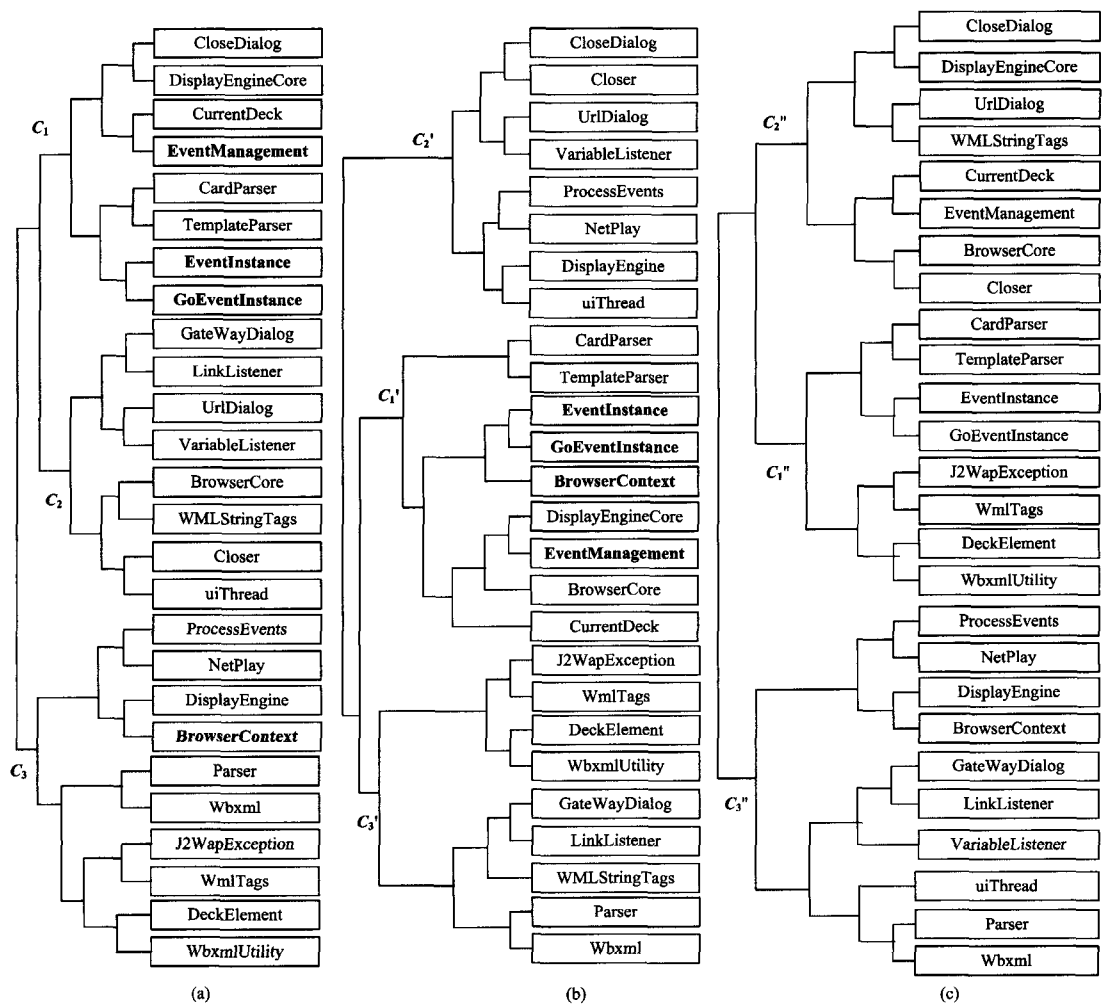

图 3　HCSAR 和 LIMBO 的 j2wap 聚类结果

采用 1 和 20 作为头文件类别差异权重和文件 trace. txt 的特殊关注点权重对 MPEG4 聚类，聚类后期，165 个实体被划分为 5 个簇，经合并及拆分，簇的稳定度由 5 个稳定簇对的 0.865，0.906，0.563，0.563 和 1 提高为 4 个稳定簇对的 0.865，0.906，1 和 0.922 或 3 个簇对的 0.656，1 和 0.875，且得到相应的系统划分。

(4)聚类相似度

图 3(c)为 LIMBO 聚集 j2wap 的结果，C_1''，C_2''和 C_3''与 C_1，C_2 和 C_3 级别相同，合并 C_1''，C_2''和 C_3''中任意两个簇，(C_1''，$C_2''\cup C_3''$)的聚类相似度最大。HCSAR 和 LIMBO 的不同系统划分的聚类相似度区间分别为[0.811，0.9]和[0.744，0.844]。由数据可知，HCSAR 的聚类相似度优于 LIMBO 的，其相应的系统划分更准确。

HCSAR 和 LIMBO 聚集 MPEG4 的结果与 j2wap 相似，聚类相似度区间为[0.83，0.907]，LIMBO 仅为 0.663。由此可知，相对稳定的簇对构造的系统划分的聚类相似度更高，HCSAR 能辅助构造准确度更高的系统划分。

结束语 本文针对软件聚类欠缺考虑实体和特征的特性的问题，提出了一种基于层次聚类的软件架构恢复方法(HCSAR)。该方法针对软件系统类型选取实体和特征，首次提出特征的多重加权策略，采用信息丢失度来衡量实体相似度，并选取和设计软件聚类的客观和主观评估准则。与目前效果较好的方法对比，HCSAR 能增加聚类中期簇的数目，提高结果的内聚度；能降低主观判定数，提高效率和准确度；能灵活地调整聚类关注点，获得不同的聚类结果；使用设计的评估准则分析关注点不同的聚类结果，能辅助构造准确度更高的系统划分。

参 考 文 献

[1] BIBI M, MAQBOOL O. Version information support for software architecture recovery[C]// Proceedings of the 7th International Conference on Emerging Technologies, 2011. Islamabad: IEEE, 2011: 1-6.

[2] LI Q S, CHEN P. Implementing architecture recovery by using improved genetic Algorithm[J]. Journal of Software, 2003, 14(7): 1221-1228. (in Chinese)
李青山，陈平. 用改进的遗传算法实现架构恢复[J]. 软件学报，2003，14(7)：1221-1228.

[3] YANG Q, ZHANG L P. The research on software architecture recovery based on pattern matching[J]. Computer Application and Software, 2006, 23(4): 27-29. (in Chinese)
杨清，张礼平. 基于模式匹配的软件架构恢复的研究[J]. 计算机应用与软件，2006，23(4)：27-29.

[4] SIDDIQUE F, MAQBOOL O. Enhancing comprehensibility of software clustering results[J]. Software, IET, 2012, 6(4): 283-295.

[5] MAQBOOL O, BABRI H. Hierarchical clustering for software architecture recovery[J]. IEEE Transactions on Software Engineering, 2007, 33(11): 759-780.

[6] CHOT S, CHA S, TAPPERT C. A survey of binary similarity distance measures[J]. Journal of Systemics, Cybernetics and Informatics, 2010, 8(1): 43-48.

[7] SAEED M, MAQBOOL O, BABRI H, et al. Software clustering techniques and the use of the combined algorithm[J]. Journal of Software Maintenance and Evolution: Research and Practice, 2003, 16(4-5): 301-306.

[8] MAQBOOL O, BABRI H. The weighted combined algorithm: a linkage algorithm for software clustering[C]// Proceedings of the 8th European Conference on Software Maintenance and Reengineering, 2004. Los Alamitos, Calif.: IEEE Computer Society, 2004: 15-24.

[9] ANDRITSOS P, TZPERPOS V. Information theoretic software clustering[J]. IEEE Transactions on Software Engineering, 2005, 31(2): 150-165.

[10] NASEEM R, BIN M, MAQBOOL O. Software modulariza-tion using combination of multiple clustering[C]// Proceedings of IEEE 17th International Conference on Multi-Topic Conference, 2004. Karachi: IEEE, 2014: 277-281.

(上接第 71 页)

[12] BÉRARD B, HADDAD S, SASSOLAS M. Interrupt Timed Automata: verification and expressiveness[J]. Formal Methods in System Design, 2012, 40(1): 41-87.

[13] FLORIAN M, GAMBLE E, HOLZMANN G. Logic Model Checking of Time-Periodic Real-Time Systems[C]// Infotech@aerospace. 2013: 1-43.

[14] SENN E, LAURENT J, JUIN E, et al. Refining Power Consumption Estimations in the Component-based AADL Design Flow[C]// Forum on Specification, Verification and Design Languages, 2008(FDL 2008). IEEE, 2008: 173-178.

[15] GLUCK P R, HOLZMANN G J. Using SPIN model checking for flight software verification[C]// Aerospace Conference Proceedings, 2002. IEEE, 2002: 105-113.

[16] CLARKE E M, ZULIANI P. Statistical Model Checking for Cyber-Physical Systems[C]// International Conference on Automated Technology for Verification and Analysis. Springer-verlag, 2011: 1-12.

[17] HÅKAN L, YOUNES S. Ymer: A Statistical Model Checker [M]// Computer Aided Verification. Springer Berlin Heidelberg, 2005: 429-433.

[18] DE I I D G, WEYNS D. Guaranteeing robustness in a mobile learning Application using formally verified MAPE loops[C]// Proceedings of the 2013 8th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS). IEEE Computer Society, 2013: 83-92.

[19] BARTELS B, KLEINE M. A CSP-based framework for the specification, verification, and implementation of adaptive systems[C]// Proceedings of the 6th International Symposium on Software Engineering for Adaptive and Self-Managing Systems. ACM, 2011: 158-167.

[20] RAMIREZ A J, CHENG B H C. Verifying and Analyzing Adaptive Logic through UML State Models[C]// International Conference on Software Testing, Verification, and Validation. IEEE, 2008: 529-532.

[21] LUCKEY M, THANOS C, GERTH C, et al. Multi-Staged Quality Assurance for Self-Adaptive Systems[C]// 2013 IEEE 7th International Conference on Self-Adaptation and Self-Organizing Systems Workshops. IEEE, 2012: 111-118.