VCSTalk: 基于 DEA 的开源软件开发人员效率评价工具*

童 杰[1,2], 王永吉[1,2], 张 曋[1,2]
(1. 中国科学院 软件研究所 互联网软件技术实验室, 北京 100080; 2. 中国科学院 研究生院, 北京 100049)

摘 要: 介绍了 VCSTalk 的具体设计和实现架构, 并展示了它用于评价若干开源软件项目中的开发人员效率的一个具体实例。

关键词: 数据包络分析; 开源软件; 开发人员效率评价
中图分类号: TP311.56 **文献标志码**: A **文章编号**: 1001-3695(2007)04-0054-04

VCSTalk: DEA-based Tool for Open Source Software Developer's Performance Evaluation

TONG Jie[1,2], WANG Yong-ji[1,2], ZHANG Shen[1,2]
(1. *Laboratory for Internet Software Technologies, Institute of Software, Chinese Academy of Sciences, Beijing* 100080, *China*; 2. *Graduate School, Chinese Academy of Sciences, Beijing* 100049, *China*)

Abstract: VCSTalk, a DEA-based evaluation tool was presented. And a set of open source software projects was applied, proved that the payoff of the evaluation tool is substantial.

Key words: data envelopment analysis(DEA); open source software; developer's performance evaluation

0 引言

开发人员效率评价即以统一的标准对一系列开发人员的效率进行评估, 以评估结果表明各个开发人员的相对效率高低。该评价工作对开源软件组织和个人很有意义。例如可以给出每个开发者的能力相对高低, 有助于开源软件组织更合理地分配资源和控制项目进度。此外, 通过效率评价还可以列举出具有相对高效率的开发者, 将其作为学习对象, 以帮助开源软件开发人员提高自身能力。

对开源软件开发人员效率进行评价存在如下困难: ①开源软件项目的过程数据往往未被集中收集管理, 而是散布于各种软件配置库如版本控制系统和缺陷管理系统中, 因此度量数据的收集比较困难。②不存在一种单一的指标可直观地表示出效率的高低。开发人员的生产率和一些其他因素, 如工作成果质量都会影响到效率高低。换言之, 该评价问题具有多变量输入/输出特性。此外文献[1]研究证明, 对个人软件过程能力的评价问题表现出可变规模收益特性, 该证明过程同样适用于对开发人员效率的评价。因此对开源软件开发人员的效率评价问题也具有可变规模收益特性(即输入和输出具有非线性关系, VRS)。该评价问题具有非线性和多变量输入/输出的特点。开源项目开发人员效率评价工具需要解决数据收集的困难, 如避免手工采集各个软件配置库中的过程数据; 还必须解决问题的多变量输入/输出和 VRS 特性。

针对上述问题设计并实现的评价工具 VCSTalk, 兼顾了上述两方面要求: 自动地从多种软件配置库(如版本控制系统、缺陷管理系统等)中获取项目数据用于执行评价工作, 在解决数据收集问题的同时降低了人力开销; 基于数据包络分析技术, 既能给出量化的、直观的评价结果, 又较好地解决了多变量和 VRS 特性问题。为验证其可用性, VCSTalk 被应用于 20 个开源软件项目中的 22 位开发人员, 得到了客观可靠的评价结果。

1 DEA及其经典模型

Charnes A. 和 Cooper W. W.[2] 于 1978 年提出的 DEA 是一种用于前沿估计的非参数化数学规划方法, 适用于评价一系列决策单元(Decision Make Unit, DMU)的相对效率。DMU 的效率可以看做是 DMU 的输出与输入之间的比值关系, 输入即 DMU 的消耗, 如耗费成本的数量; 输出即 DMU 的产值。作为一种极限点方法, DEA 将每个生产者(即 DMU)与最好的生产者加以比较, 从而得到生产者的相对效率。所谓最好的生产者往往是虚拟的, 由若干实际生产者组成, 这些相对效率高的 DMU 构成了效率前沿面(Efficiency Frontier)。

DEA 被广泛应用于各个领域的性能估计和评测[3], 在软件工程领域也同样有应用[4]: 展示了如何使用 DEA 从一系列 ERP 项目中识别出高效率的项目。前期研究成果[1] 则给出了

收稿日期: 2006-02-28; 修返日期: 2006-04-15 基金项目: 国家自然科学基金资助项目(60373053); 中国科学院"百人计划"留学回国人员科研启动基金资助项目([2003]406); 国家"863"计划资助项目(2002AA116060)

作者简介: 童杰(1982-), 男, 硕士研究生, 主要研究方向为软件工程(tongjie@itechs.iscas.ac.cn); 王永吉(1962-), 男, 辽宁盖州人, 研究员, 博导, 主要研究方向为实时系统与软件工程; 张曋(1981-), 男, 博士研究生, 主要研究方向为软件工程.

一种基于 DEA 的个人软件过程能力评价方法, 但将 DEA 方法在实际工具中实现用来解决实际中的问题, VCSTalk 尚为首例。

从 DEA 方法提出至今, 研究者们已经提出了多种 DEA 模型[2,3,5~7]。VCSTalk 采用了两种经典的 DEA 模型, 即 Charnes 等人提出的 CCR 模型[2] 以及 Banker 等人提出的 BCC 模型[5]。CCR 模型和 BCC 模型的对偶规划形式如下:

CCR 模型

$$\min_{\theta,\lambda,s_i^+,s_k^-} \quad \theta - \varepsilon \left(\sum_{i=1}^{m} s_i^+ + \sum_{k=1}^{s} s_k^- \right)$$

s.t.

$$\theta x_{ij_0} - \sum_{j=1}^{n} x_{ij} \lambda_j - s_i^+ = 0, \; i = 1, \cdots, m$$

$$\sum_{j=1}^{n} y_{kj} \lambda_j - y_{kj_0} - s_k^- = 0, \; k = 1, \cdots, s$$

$$\lambda_j \geq 0, \quad j = 1, \cdots, n$$

$$s_i^+ \geq 0, \quad i = 1, \cdots, m$$

$$s_k^- \geq 0, \quad k = 1, \cdots, p$$

BCC 模型

$$\min_{\theta,\lambda,s_i^+,s_k^-} \quad \theta - \varepsilon \left(\sum_{i=1}^{m} s_i^+ + \sum_{k=1}^{s} s_k^- \right)$$

s.t.

$$\theta x_{ij_0} - \sum_{j=1}^{n} x_{ij} \lambda_j - s_i^+ = 0, \; i = 1, \cdots, m$$

$$\sum_{j=1}^{n} y_{kj} \lambda_j - y_{kj_0} - s_k^- = 0, \; k = 1, \cdots, s$$

$$\sum_{j=1}^{n} \lambda_j = 1$$

$$\lambda_j \geq 0, \quad j = 1, \cdots, n$$

$$s_i^+ \geq 0, \quad i = 1, \cdots, m$$

$$s_k^- \geq 0, \quad k = 1, \cdots, p$$

其中, θ表示 DMU_{j_0}的效率分值(Efficiency Score), 其取值范围为 0 到 1。效率分值为 1 的 DMU 为相对有效(Relative Efficient); 否则即为非相对有效(Relative Inefficient)。s_i^+($\theta x_{ij_0} - \sum_{j=1}^{n} x_{ij} \lambda_j$) 和 s_k^-($\sum_{j=1}^{n} y_{kj} \lambda_j - y_{kj_0}$) 分别是输入和输出的松弛与剩余变量。松弛变量表示对输入的过量使用, 而剩余变量则表示输出的不足。它们的数值大小表示使该 DMU 从非相对有效变为相对有效可作的改进及其幅度大小。在 VCSTalk 的评价结果中, 效率分值和剩余变量是最重要的数据。通过观察效率分值可以了解每个被评估开发人员的相对效率表现; 而通过对剩余变量进行分析可以得知特定开发人员应该通过改进哪些方面来改进自身的效率。文献[7] 对 CCR 和 BCC 模型进行了更细致的讨论。

2　VCSTalk 的设计与实现

2.1　系统工作流程

VCSTalk 的工作流程包括两个主要步骤: ①根据确立的效率评价指标从软件配置库中获得项目数据并转换为量化的指标值; ②采用 DEA 模型对收集到的指标数据进行计算得到评价结果。

(1) 指标选择和数据获取

Moser S 等人[8] 研究了面向对象方法与开发人员生产效率之间的关系, 他们采用功能点(FP) 作为评价指标。Ying 等人[9] 通过分析从 CVS[10] 中导出的代码行(Lines of Code, LOC) 和一些其他信息来理解学生在实践课程中的开发行为, 给出了一个非常好的示例说明可以从软件配置库中获取哪些信息用于评价开发人员的效率。Keir 等人[11] 从超过 200 个项目的 CVS 中提取出 LOC 和代码质量等指标进行统计学分析, 试图找到最直观合理的效率指标。这些研究表明 LOC 和开发者工作成果的质量(包括代码质量和软件质量) 可以作为评价开发人员效率的重要指标。因此选择了四类指标作为 VCSTalk 评价开发人员效率的依据(表 1)。

表 1　VCSTalk 所使用的四类指标及其含义和量化表示

指标名称	含义	量化表示
TIME CONSUMED	特定开发人员为项目投入的时间	SUM(该开发人员在项目中的活跃时间段)
CODE QUANTITY	特定开发人员为项目贡献的代码数量	SUM(该开发人员增加的代码行数) + SUM(该开发人员改的代码行数)
CODE QUALITY	特定开发人员为项目编写的代码质量高低	CODE QUANTITY/开发人员引入项目中的违例数目
DEFECT IMMUNITY	特定开发人员的代码是否容易为项目引入 BUG	CODE QUANTITY/开发人员负责的 BUG 数目

TIME CONSUMED 以小时为单位, VCSTalk 从 CVS 的日志中获取该指标值。由于该指标的精确性受到许多外在因素的影响, 如开源软件开发人员往往要从事其他工作, 或同时参与多个软件项目, 这会带来相当大的偏差。为了在获取数据时尽量消除对时间估计上的偏差, VCSTalk 采用了去除非活跃时间的方法。从 CVS 日志中可以得到开发者行为的记录, 当某开发者两次动作的间隔长短超过某一阈值时, VCSTalk 认为该开发者没有将该时间段投入到该项目中去。该时间段即为非活跃时间。

CVS 日志记录了任意一行代码是由哪位开发者添加到项目中来的。通过分析 CVS 日志能够得到 CODE QUANTITY。该指标表示为增加的代码行数与修改的代码行数之和。值得注意的是删除的代码行已经不再是项目的一部分, 因而删除的代码行数不计入公式。

CODE QUALITY 在项目中反映为开发人员编写代码中违例的数量多少。违例多则质量低。在这里, 违例所指的是代码中的各种缺陷, 如在一行中书写多个语句, 出现定义但未被使用的变量、没有仔细处理的异常等。这些违例可以通过对源代码进行自动静态分析得到。通过结合代码分析结果和 CVS 日志可以得到某特定开发人员引入的违例数目。

DEFECT IMMUNITY 表现为该开发人员负责的 BUG 数量和密度。该指标可从项目的缺陷管理系统中获得。

上述四类指标数据的采集过程完全由程序自动进行, 避免了手工收集数据的人力开销。

(2) DEA 评价计算

在数据采集阶段完成之后, VCSTalk 使用 DEA 的 CCR 模型和 BCC 模型进行计算。在计算过程中, 将每个开发人员作为一个 DMU, 指标 TIME CONSUMED 作为 DMU 的输入, 其余三个指标作为 DMU 的输出。实际计算中使用 CCR 和 BCC 的对偶规划形式。

采用 DEA 模型进行计算后得到的量化结果中, θ和剩余变量对评价工作最有意义。θ为 DMU(在本文的问题中指开

发人员) 的效率分值, 效率分值为 1 的开发者是相对有效的, 换言之就是具有较高的效率。剩余变量表示把 DMU 从非相对有效变成相对有效所能进行的改进及可能的改进程度。举例来说, 如果开发人员 A 的指标 CODE QUALITY 对应的剩余变量较大, 则他/她应该通过提高自己的代码质量来改进效率。

2.2 系统架构与实现

如前所述, VCSTalk 的工作流程是一个典型的流水线式的过程。因此 VCSTalk 的系统架构被设计成管道结构(图 1): 数据采集组件(DCC)、中间数据库组件(IDB)、指标导出组件(MEC) 和 DEA 分析其(DA) 四个组件各自都将前一个组件的输出作为自己的输入并把自身输出作为后续组件的输入。控制器组件负责协同四个组件的行为来处理一个给定的效率评价任务。DCC 从软件配置库中收集原始数据输入到 IDB 中; IDB 管理原始数据并响应 MEC 的查询请求; MEC 根据 IDB 提供的数据生成四种指标(第 2 章中定义的) 的指标值并传递给 DA。特别值得一提的是 MEC 通过调用代码静态分析工具(如 PMD[12]) 分析项目源代码来获得 CODE QUALITY 指标数据。DA 生成最终的评价结果, 其输入和输出均为表格形式。VCSTalk 系统架构详图如图 2 所示。

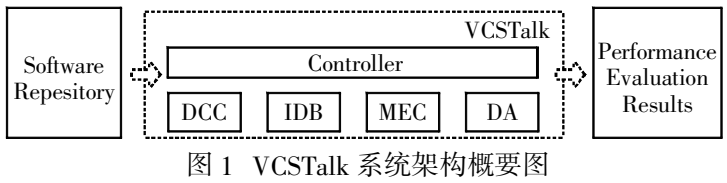

图 1 VCSTalk 系统架构概要图

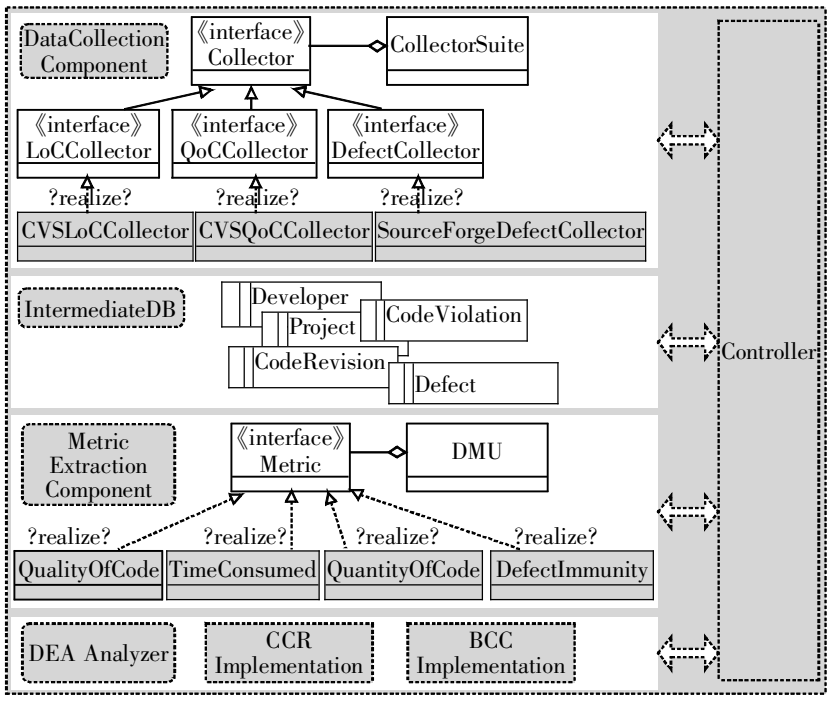

图 2 VCSTalk 系统架构详图

现有版本的 VCSTalk 实现了三种收集器对象和四种指标对象(对应四种指标)。CVSLoCCollector 对象从 CVS 中采集代码量信息并导入到 IDB 的 CodeRevision 对象中。CVSQoCCollector 对象结合 PMD 作源代码分析和 CVS 日志分析, 得到代码中的各种违例并导入到 IDB 的 CodeViolation 对象中。SourceForgeDefectCollector 实现了一个网页爬虫, 它从 SourceForge[13] 网站上抓取特定项目的 BUG 信息并导入到 IDB 的 Defect 对象中。

VCSTalk 的 DEA 分析器模块用 Mathematica[14] 开发, 其余部分用 Java 语言实现。由于目前版本调用 PMD 进行代码静态分析, 而 PMD 仅支持 Java 源文件分析, 尚不能对其他语言类型的项目开发人员进行评价, 可对 CVSQoCController 进行扩展以支持其他语言项目。VCSTalk 的源代码可从地址 http://file.ttthree.com/下载。

3 应用实例及结果讨论

为了验证 VCSTalk 的可用性和有效性, 本文从 SourceForge 选取了 20 个 Java 语言项目。在参与这些项目的 180 多位开发人员中选取了 22 位活跃开发人员, 使用 VCSTalk 对他们的开发效率进行评价。作为开发人员选取的标准, 活跃意味着频繁参加项目开发(从 CVS 日志中可以统计出参与开发的频度)。

VCSTalk 运行得到的 CCR 和 BCC 模型计算结果如表 2、3 所示。受篇幅所限, 除效率分值和剩余变量外, 其他数据均被略去, DEA 计算的原始数据也未在文中给出(附于源程序包中, 可从 http://file.ttthree.com/下载)。

表 2 22 位开发人员的 CCR 模型效率分值及剩余变量数据

DMU	θ	s_2^-	s_3^-	DMU	θ	s_2^-	s_3^-
1	0.16	8 819.77	414.82	12	0.05	1 844.93	94.93
2	0.49	0	1 907.73	13	0.05	249.08	58.95
3	0.06	0	4 411.40	14	0.04	0	0.73
4	0.04	0	44.74	15	0.04	417.02	28.92
5	1.00	0	0	16	0.05	1 009.83	133.90
6	0.21	395.32	48.06	17	1.00	0	0
7	0.09	0	308.76	18	0.11	0	0
8	0.06	0	514.25	19	1.00	0	0
9	0.21	3 257.44	155.99	20	0.05	1 671.17	104.73
10	0.08	2 175.53	97.34	21	0.12	0	4 918.32
11	0.33	151.78	0	22	0.53	130 000	0

表 3 22 位开发人员的 BCC 模型效率分值及剩余变量数据

DMU	θ	s_2^-	s_3^-	DMU	θ	s_2^-	s_3^-
1	1.00	0	0	12	0.24	0	2.35
2	0.55	0	338.38	13	0.15	0	95 705.42
3	0.08	914.23	9 331.13	14	0.06	2 126.5	11 458.43
4	0.13	0	95 000.2	15	0.13	0	48 630.56
5	1.00	0	0	16	0.25	0	1 540.75
6	0.71	2 403.14	156 916	17	1.00	0	0
7	0.22	3 858.24	39 136	18	0.22	0	0
8	0.21	0	118 421	19	1.00	0	0
9	0.71	493.20	255 216	20	0.26	0	11.46
10	0.26	244.88	178 242	21	0.13	0	923.04
11	1.00	0	0	22	1.00	0	0

对上述结果进一步讨论:

(1) 在 CCR 模型的计算结果中, 开发者 5、17、19 构成了效率前沿面, 换言之, 这三位开发者与其他开发人员相比具有较高的效率。而在 BCC 模型计算结果中, 效率前沿面由开发者 1、5、11、17、19、22 构成。该计算结果的合理性可以通过对原始数据的人工分析得到验证: 开发人员 5、17、19 在相对短的时间内为他们所在的项目贡献了较多的代码且代码中违例少, 引入的 BUG 也相对少。开发人员 22 与其他开发人员相比贡献的代码数量少, 但其代码质量特别高且引入的 BUG 非常少。开发人员 1 和 11 代码中违例数量较多但他们为项目完成的代码量特别大, 所以他们也处于效率前沿面。

(2) 对于那些效率分值低于 1 的开发人员来说, 上表中所列的剩余变量表明他们可以通过改进相应的指标来提高效率。剩余变量的数值大小表示可供改进的余地大小。表 3 和 4 中

s_2^- 和 s_3^- 分别对应 CODE QUALITY 指标和 DEFECT IMMUNITY 指标。根据两表结果, 开发人员可以知道他们应该通过改进哪些方面来提高效率以及相应可改进的幅度。

(3) VCSTalk 的计算结果中各开发人员的效率分值波动较大, 其根本原因在于 TIME CONSUMED 指标的精确程度不足, 在后续研究及开发中有待改进。

4　总结

对开源软件开发人员的效率评价问题具有多变量输入/输出及 VRS 特性, 且由于开源软件的过程数据有其自身特点而较为困难。本文介绍的基于 DEA 方法的开源软件开发人员效率评价工具 VCSTalk 在相当程度上解决了上述困难。VCSTalk 提供了一种全新的开发人员效率评价方式, 其优点包括: ①自动数据收集机制减少了人工开销; ②考虑多种指标以及给出直观的量化结果; ③评价结果可以对开发人员改进效率提供指导。VCSTalk 同样适用于非开源软件组织。VCSTalk 所能收集到的信息越详细则评价结果越客观和可靠。由于商业软件组织通常有较为规范的过程数据和严格定义的软件过程, VCSTalk 的应用效果会更佳。

后续的研究工作将关注以下方面: ①评价结果的可视化, 以可视化的形式代替目前的表格形式, 使得评价结果更为直观, 便于分析。②更广泛和细粒度的数据采集, 目前 VCSTalk 仅引入了四项指标, 更多指标的引入以及指标数据获取的精确度提高, 都有助于提高评价的质量。③评价的实时化。目前的评价是在事后进行的, 实现评价的实时化, 即在项目进行过程中持续地给出评价结果将有助于项目及开发人员的持续改进。

参考文献:

[1] ING Liping, YANG Qiusong, SUN Liang, *et al*. Evaluation of the capability of personal software process based on data envelopment analysis: proc. of the International Software Process Workshop on Unifying the Software Process Spectrum: Lecture Notes in Computer Science, Vol. 3840[C]. Beijing: Springer-Verlag GmbH, 2005: 235-248.

[2] CHARNES A, COOPER W, RHODES E. Measuring the efficiency of decision making units[J]. **European Journal of Operation Research,** 1978, **2**(6): 429-444.

[3] WEI Quanling. Use DEA to evaluate the relative efficiency: a new filed of operational research[M]. Beijing: China Renmin University Press, 1987: 10-13.

[4] STENSRUD E, MYRTVEIT I. Identifying high performance ERP projects[J]. **IEEE Transaction on Software Engineering,** 2003, **29**(5): 387-416.

[5] BANKER R D, CHARNES A, COOPER W. Some models for estimating technical and scale inefficiencies in data envelopment analysis [J]. **Management Science,** 1984, **30**(9): 1078-1092.

[6] HUANG Zhimin, LI S X. Dominance stochastic model in data envelopment analysis[J]. **European Journal of Operational Research,** 1996, **95**(2): 390-403.

[7] HUANG Z, SUN D B, Wei Q L. Theories and application of the compositive data envelopment analysis model with cone structure[J]. **SCI-TECH Information Services,** 1995, **6**: 57-73.

[8] MOSER S, NIERSTRASZ O. The effect of object-oriented frameworks on developer productivity[J]. **Computer,** 1996, **29**(9): 45-51.

[9] YING L, ELENI S, KEN W, *et al*. Using CVS historical information to understand how students develop software: proc. of the 1st International Workshop on Mining Software Repositories[C]. Edinburgh: [s. n.], 2004: 32-36.

[10] CVS, 一种使用广泛的版本控制系统, 用于对管理代码、文档进行版本控制[EB/OL]. http://www.cvs.org.

[11] KEIR M, KEVIN L, SAM R, *et al*. Mining student CVS repositories for performance indicators: proc. of the 2nd International Workshop on Mining Software Repositories[C]. Saint Louis: [s. n.], 2005: 41-47.

[12] PMD, 程序静态分析工具, 用于检查 Java 语言程序代码中的缺陷[EB/OL]. http://pmd.sourceforge.net.

[13] SourceForge, 世界上最大的开源软件项目集散站点之一[EB/OL]. http://sourceforge.net.

[14] Mathematica, 功能强大的数学程序开发环境, 由 Wolfram 公司开发[EB/OL]. http://www.wolfram.com.

(上接第 28 页)

[4] ALONE T W, CROWSTON K. The interdisciplinary study of coordination[J]. **ACM Computing Surveys,** 1994, **26**(1): 87-119.

[5] THOMPSON J D. Organizations in action: social science bases of administrative theory[M]. New York: McGraw-Hill, 1967.

[6] ALLEN J F. Towards a general theory of action and time[J]. **Artificial Intelligence,** 1984, **23**(2): 123-154.

[7] MALONE T W, CROWSTON K. Tools for inventing organizations: toward a handbook of organizational processes[J]. **Management Science,** 1999, **45**(3): 425-443.

[8] VON M F. Multiagent plans relationships: proceedings of the Ninth Workshop on Distributed Artificial Intelligence[C]. Rosario Resort: [s. n.], 1989: 59-72.

[9] DECKER K, LESSER V. Designing a family of coordination algorithms: proceedings of the First International Conference on Multi-Agent Systems(ICMAS'95)[C]. San Francisco: [s. n.], 1995.

[10] OMICINI A, OSSOWSKI S. Objective versus subjective coordination in the engineering of agent systems[C]//KLUSCH M, BERGAMASCHI S, EDWARDS P, *et al*. Intelligent information agents: an agentLink perspective, volume 2586 of LNAI: state-of-the-art survey: [S. l.]: Springer-Verlag, 2003: 179-202.

[11] RAPOSO A B, MAGALHAES L P, *et al*. Coordination of collaborative activities: a framework for the definition of tasks interdependencies: proceedings of the Seventh international Workshop on Groupware-CRIWG2001[C]. Darmstandt: [s. n.], 2001.

[12] 马巧云, 陈学广, 洪流. 一种基于协调知识水平的协调问题描述方法[J]. 系统工程理论与实践, 2005, **25**(7): 87-92.

[13] 范玉顺. 工作流管理技术基础[M]. 北京: 清华大学出版社, 2001: 151-153.

[14] 袁崇义. Petri 网原理[M]. 北京: 电子工业出版社, 1998.

[15] RAPOSO A B, FUKS H. Defining task interdependencies and coordination mechanisms for collaborative systems[J]. **Frontiers in Artificial Intelligence and Applications,** 2002(34): 88-103.