软件过程中问题跟踪和分析系统的研究

汤效琴[1]，戴汝源[2]

（1. 宁夏大学 数学计算机学院，宁夏 银川 750021；
2. 宁夏大学 物理电气信息学院，宁夏 银川 750021）

摘 要：在软件项目的实施过程中，开发和管理人员不可避免地会碰到各种各样的问题，如何及时、有效地解决这些问题，对一个软件项目最终的成功与否有着极大的影响。经大量的研究工作后，开发了问题跟踪和分析系统。介绍了该系统中对问题进行定义、跟踪、管理和解决的工具——问题跟踪工具的实现方法，以及对问题跟踪工具应用后积累的历史数据进行挖掘分析的方法。该系统的应用对提高软件生产率和软件质量起到了一定积极有效的作用。

关键词：问题跟踪；问题分析；实例化视图；分类规则

中图分类号：TP311　　**文献标识码**：A　　**文章编号**：1673－629X(2007)04－0192－04

Problem Tracking and Analysis System for Software Processes

TANG Xiao-qin[1], DAI Ru-yuan[2]

(1. School of Mathematics & Computing Engineering, Ningxia University, Yinchuan 750021, China;
2. School of Physics Electric & Info., Ningxia University, Yinchuan 750021, China)

Abstract: Currently the software development and maintaining is an extremely complex process. In this process, various problems may emerge. How to solve these problems timely and effectively is a critical issue. Addresses a fundamental issue, the problems tracking and analysis, in facilitating software development and maintaining where the conduct of processes depends on the tools and system to the relevant developer and maintainer. Discuss system implement method to provide a computing framework for ease of tracking and analysis of problems by taking into consideration the state－of－the－art software technologies. By doing so, the efficient of software development and quality will be enhanced largely.

Key words: problem tracking; problem analysis; materialized view; classification rules

0 引 言

在软件过程的实施过程中，开发和管理人员不可避免地会碰到各种各样的问题，软件项目的成功与否很大程度上依赖于这些问题的及时、有效和合理的解决。当前，人们对问题的研究一方面集中于跟踪问题的处理过程，通过定义问题的处理过程和追踪问题的状态变化来达到问题的及时、有效和合理的解决[1]。另一方面是问题的分析，通过对本项目中产生问题的分析和度量来对项目进行总结，更进一步，通过对使用同一软件过程的众多历史项目中的问题信息进行分析，可以优化和改进软件过程[2]。

在这一领域笔者参与做了大量的研究工作，开发了问题跟踪和分析系统。该系统分为两个部分，其中问题跟踪工具（Issue Tracking Tool，ITT）和软件工程支持环境紧密地结合，能够追踪当前执行项目中问题的处理过程，实现问题信息管理自动化、追踪软件错误的解决方案、监控问题处理进度、量化软件质量管理，把问题的处理流程置于严格的监控之下；问题分析系统（Issue Analysis System，IAS）将数据挖掘、OLAP 等新技术应用于历史问题库，对问题进行分类、聚类操作和多维数据分析，从中发现规则、模式和统计信息，用来改进企业的软件过程、对正在实施的项目进行更合理的评估和支持企业的决策分析。

1 问题跟踪工具（ITT）

在软件开发过程中，人们往往会由于出现问题而受挫，但是一旦找到问题并解决了它，那么一切就变得十分顺利了。过去，常常采用书面的方式来定义、控制和追踪问题。但是这样做既不方便，也不容易控制。ITT 使用计算机化的方式来定义、控制和追踪问题。

收稿日期：2006－07－22

基金项目：宁夏自然科学基金项目（D0004(2004)）

作者简介：汤效琴（1970－），女，宁夏固原人，硕士，讲师，研究方向为程序设计语言、数据库和信息管理系统等。

它帮助开发人员和分析人员有序地管理、方便地跟踪所有的问题。在这里,问题数据库构成了项目成员之间正式和有效交流的基础。同时,ITT 的实现是基于 Web 方式的,这使得异地开发的人员之间进行合作成为可能。此外,通过问题来改进软件开发过程,可以使软件企业达到更高软件开发能力与成熟度(Capability Maturity Model,CMM)层次。

1.1　ITT 的体系结构

ITT 使用基于 Web 的客户端/服务器模型,其体系结构如图 1 所示。

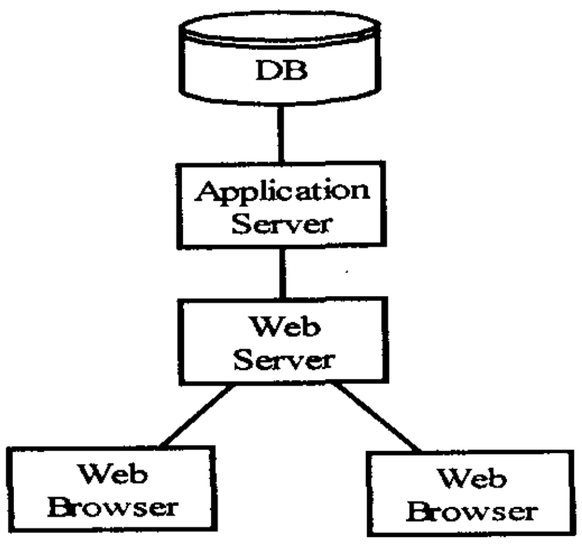

图 1　ITT 的体系结构

它包括一个问题库、一个执行应用程序的应用服务器、一个 Web 服务器和多个作为客户端的 Web 浏览器。这种结构是基于 Web 方式的,用户可以通过 Internet 来操作该工具,这使得用户可以随时随地使用该工具。同时也较好地支持项目的异地开发。

1.2　问题的属性

问题的属性是在提出问题的时候必须指定的,它不仅将在问题处理过程中起作用,而且在问题的统计分析中,它也将作为分析过程中的各个维起重要作用。我们对问题进行了分析,确定了以下属性:主题、所属项目、类型、严重程度、状态、关键程度、可视范围、阶段、影响范围、责任人、父问题、希望解决时间和描述。这些属性能全面描述问题在整个周期中所有情况。

1.3　问题的跟踪

把每个问题的处理过程都记录在问题的解决日志中,所以能通过问题解决过程的日志来追踪问题。通过分析问题日志中的信息,可以评估问题对开发进度和产品质量的影响,同时也可以统计解决该问题所需要的代价。问题追踪也能跟踪到相关联的问题,比如通过子问题,能够向后跟踪到触发它的父问题。同样,通过父问题,能够向前跟踪到它触发的所有子问题。一个父问题只有在它所有子问题都关闭的情况下,它自己才能被关闭。

1.4　问题的生命周期

问题的生命周期描述如图 2 所示,下面解释问题状态变迁的规则。

1) 识别并提出问题。

当项目成员识别并提出一个问题,该问题的状态是初始状态。提出的问题必须包含下列属性:该问题所属的项目、问题主题、问题类型、严重等级、关键级别、可视范围、问题的责任人、问题所属阶段、问题影响的阶段、希望解决的时间点和问题的描述。如果该问题由别的问题触发,则需指定它的父问题。如果提出问题的人不知道谁是问题的责任人,那么该问题就交给问题所属阶段的负责人和项目经理,由阶段负责人或项目经理来指定解决这个问题的责任人。

2) 审核和发布问题。

根据所递交问题的不同类型来决定该问题发布前是否需要通过审核。如果问题需要通过审核,那么问题将发送给项目经理,由项目经理来决定该问题是发布、延迟或是抛弃。如果问题被发布,那么问题的状态将变成激活状态;如果问题被延迟,问题的状态将变为延迟状态;最后如果问题被抛弃,那么问题将被关闭。

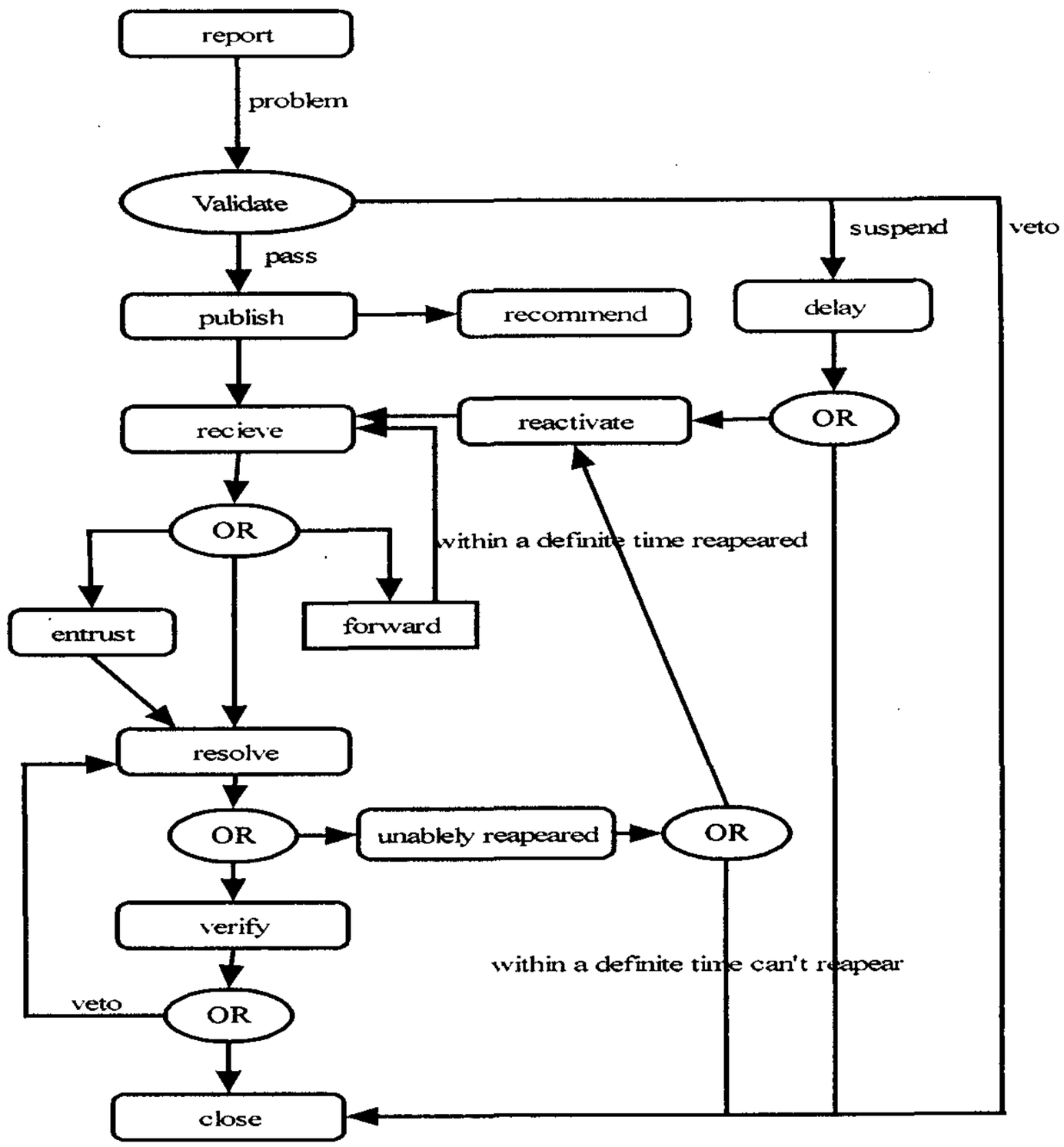

图 2　问题的生命周期

反之,如果问题不需要通过审核,那么问题的状态将直接变成激活状态。

3) 问题的解决过程。

当问题发布后,将致力于问题的解决过程。责任人收到问题后,分以下几种情况:

(1)责任人是自己,而且由自己来解决,则填写好处理信息,按“提交”按钮提交以待确认。

(2)责任人是自己,但自己比较忙,委托给他人处理,则按“委托”按钮,选择委托人。

(3)对责任人不是自己的问题则转发给该问题的真正责任人处理。

委托和转发的不同是转发的话责任人就变了,而委托的责任人不变。如果问题被解决,问题的状态将变为已解决状态。处于无法再现状态的问题可以在规定的时间段内再重新被激活,成为激活状态。如果无法再现状态的问题在规定的时间段内还没有重新激活,该问题将自动被关闭,状态变为关闭状态。

4)问题的审核和关闭。

已解决的问题需要在评审会议上审核,如果问题的解决方案在评审会议上通过,它的状态将变为已关闭状态。否则,该问题将被重新激活,状态再次变为激活状态,表示该问题被返回给责任人去重新解决。

5) 提建议。

一个问题发布后,在它没有被关闭之前,每个有权限看到该问题的人都可以对问题的解决方案提出意见。这些建议可能会帮助责任人解决问题。

6) 问题的查询。

每个用户都可以按照各种条件组合查找他们有权限看到的问题。系统提供了以下条件的组合查询:问题 ID 号、项目名称、阶段名称、受影响的阶段、问题严重程度、问题的状态、问题的类型、问题的提出者、问题的责任人和问题的提出时间。各个项之间在查询的时候是与的关系,如果某个项为空,表示对该项不作限制。如果所有项都为空,则显示该用户有权限看到的所有问题。

2 问题分析系统(IAS)

ITT 的使用可以将一个软件公司在长期运作过程中积累的大量问题资料保存起来,这些历史数据反映了公司的“成长历程”,蕴藏着公司宝贵的知识财富。对这些历史数据进行科学有效的分析,能够为软件公司提供有力的决策支持信息。

2.1 问题分析系统的体系结构

以数据仓库、OLAP、数据挖掘等相关技术结合起来所开发形成的问题分析系统是软件企业决策支持系统的新的支持形式。其中数据仓库中存储了获取满足以上决策需求的数据源,从而实现对决策主题数据的存储和综合;OLAP 则实现对主题数据的多维分析,而通过数据挖掘提取决策支持的相关知识。各相关技术相互补充,互相依存,最终发挥各自的辅助决策支持优势,使最终决策更有效。

图 3 是问题分析系统的体系结构图。

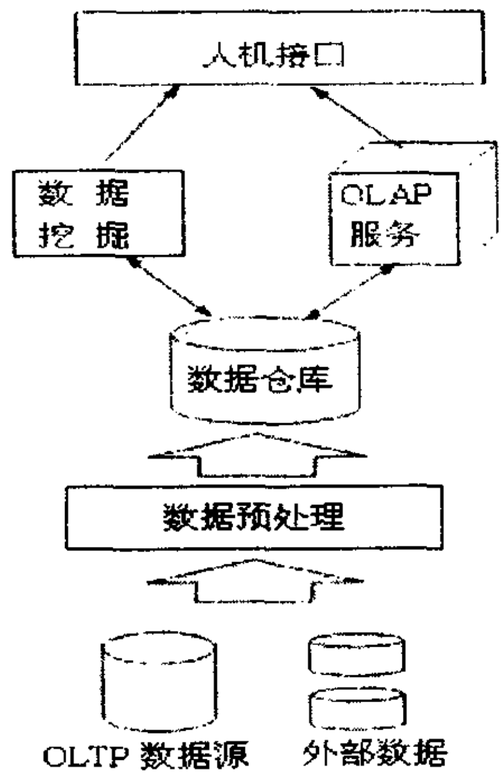

图 3 问题分析系统的体系结构

问题分析系统的人机接口向用户提供管理数据和管理数据挖掘任务的操作界面。

2.2 数据仓库设计

通过和软件公司各级管理层的探讨、分析,我们了解到,对一个软件企业的决策层来讲,他们希望获得以下一些分析结果和预测信息:

1)分析一个项目开发中严重问题的数量、严重问题的总计题龄(一个已解决问题的题龄指从提出问题到解决经历的时间,一个尚未解决问题的题龄指从提出问题至今经历的时间)、项目的规模同项目的完成质量之间的关系。一个已完成项目中的问题题龄,即是问题从提出到解决所经历的时间。

2)项目的类型、项目的规模同项目总收益和平均人月效益比之间的关系。项目总收益指项目的所得经费减去支出。

3)项目的类型、规模、项目负责人的专业背景同严重问题总数、严重问题的总计题龄之间的关系。

4)项目的类型、项目的规模、严重问题总数、严重问题的总计题龄同项目延时的关系。

5)从项目的完成部门、项目的类型、项目的规模级别、项目完成时间(可以是按年度、季度或月份)等角度,统计总收益、人均收益(总收益/部门内总人数)等,进行公司的经营状况的趋势分析。

6)从项目的负责人、项目的类型、项目的规模级别、项目完成时间、项目的完成质量等角度,统计完成项目的累计数量。

对某个开发部门来说,他们希望获得以下一些分析结果和预测信息:

(1)从项目的类型、项目的规模级别、项目中阶段、项目的完成质量等多个角度统计平均问题数量、平均题龄等聚集信息。这样,在实施一个新项目时,就可以根据正在执行的一个阶段中出现的问题数量和总的题龄,评估项目中这一阶段的完成状况。同样,也可以根据当前项目中已经完成和正在执行阶段中出现的问题总数量和总的题龄,评估这一项目到目前为止的完成状况。

(2)从项目类型、项目中阶段、问题责任人等角度,统计解决问题的总数、平均题龄等信息可以帮助项目管理人员决定把问题交给谁去解决最合适(当然,也要看该人当时的工作负荷)。

(3)从项目的规模级别、项目类型、使用的过程模型、过程模型中的阶段(即活动)、时间段等角度,统计平均的问题数量、平均的问题题龄等信息,发现软件过程模型的演化是否有效地导致问题数量的减少和问题平均题龄的缩短。

(4)从项目的规模、使用的过程模型、过程模型中的阶段(即活动)、时间段等角度,找出前三个问题最严重的阶段,目的是发现过程模型中的关键瓶颈。其中"严重程度"由该阶段中问题的数量和总的题龄经加权算出。

根据以上需求分析,确定了以下两个主题域:

①项目分析,以项目表为核心,进行在线数据分析和数据挖掘(主要是发现分类规则);

②问题分析,以问题表为核心,进行在线数据分析(OLAP)。

数据仓库的设计中,维表和事实表的设计是关键问题,维表和事实表设计的好坏直接影响到数据仓库的响应时间和分析的效果。维是决策者观察分析对象的角度,所以维的设计最能反映决策者的分析意图和角度。维的设计必须体现出数据仓库中数据的不同级别,也就是数据的粒度。数据的粒度越大,数据的综合程度就越高;数据的粒度越小,数据就越体现细节。在项目分析中,事实表为 ProjectForAnalysis,相应的维表为 Departments(部门维表)、TimeProjct(时间维表)、ProjectType(项目类型维表)、Project_sizeLevel(项目级别表)。"项目分析数据仓库"的雪花模式如图 4 所示。

在问题分析中,事实表为 ProblemForAnalysis,相应的维表为 Phases(阶段维表)、Projects(项目维表)、ProcessModel(过程模型维表)、Project_type(项目类型维表)、Agents(责任人维表)、TimeProblem(时间维表)。"问题分析数据仓库"的雪花模式如图 5 所示。

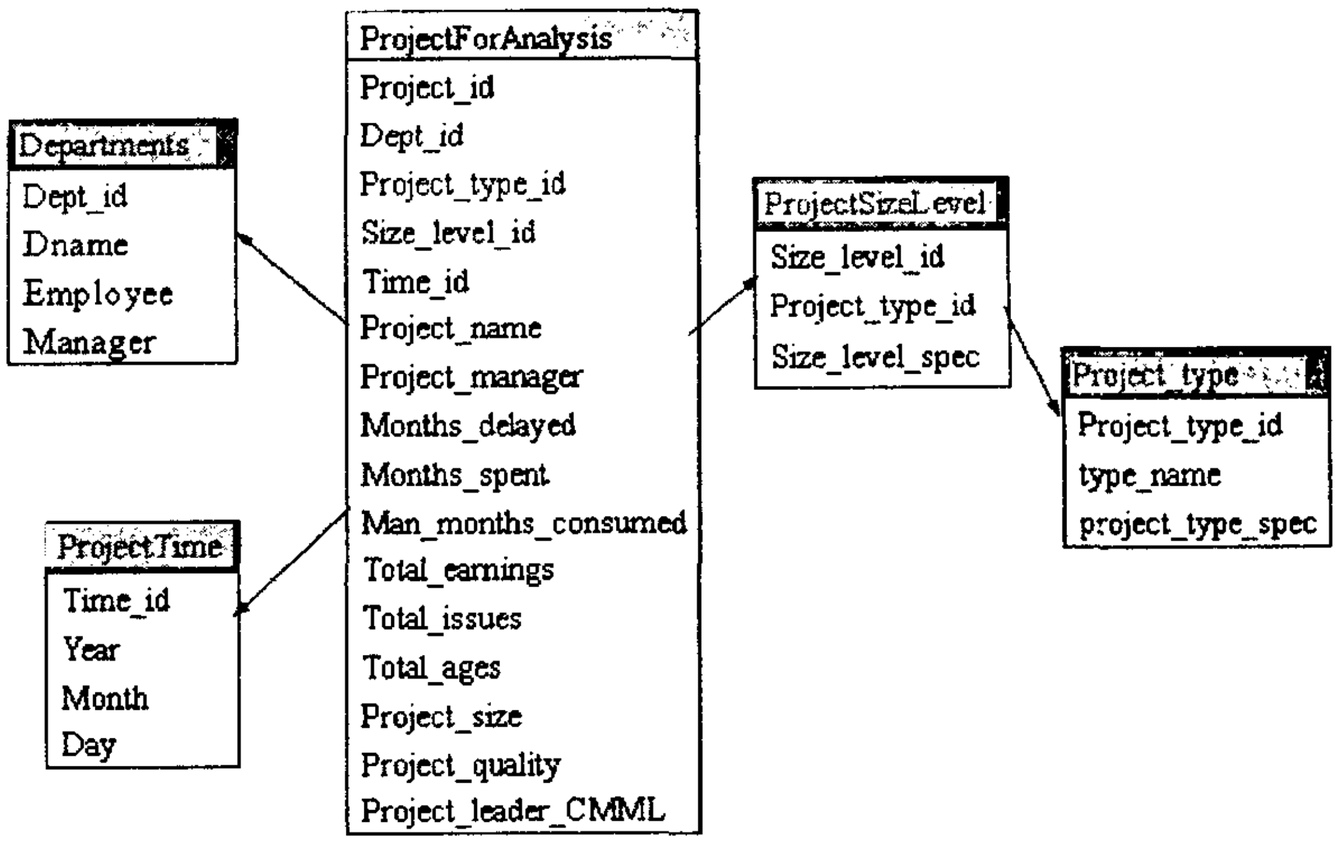

图 4　"项目分析数据仓库"的雪花模式

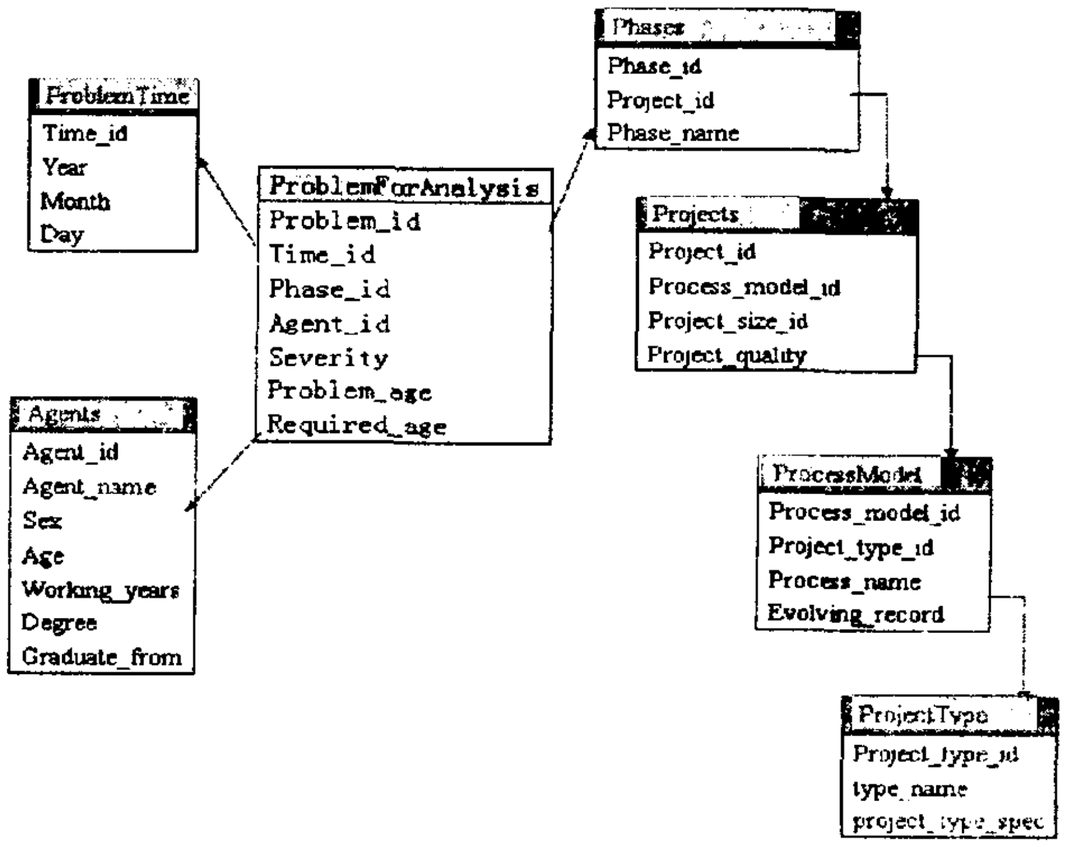

图 5　"问题分析数据仓库"的雪花模式

每个主题都是由维表和事实表组成的,而且某个主题中的维表很可能在另一个主题中也会用到,不需要重复创建这些维表,这些维表可以共用(如图 4 和图 5 共享 Project_type 维表)。

基于上述数据仓库模型,就可以使用通常关系数据库中的 DDL 语言(数据定义语言),分别为各个事实表和维表建立相应的关系模式,然后再将 OLTP 数据源(即问题跟踪数据库中的数据)中的数据存储到数据仓库中。

2.3　基于数据仓库的 OLAP 分析与数据挖掘

问题分析系统中的 OLAP 分析主要是对问题的不同属性从不同角度、不同层次进行统计分析,所以要用到大量的分组聚集(group－by),而计算多维集合上的聚集是 OLAP 应用的性能瓶颈。我们在 Cube 计算的优化方面做了研究,分析了在高维 Cube 算子计算中传统的流水线方法的不足之处[3],提出了通过有选择地

(下转第 200 页)

学习,根据以往的经验提出一个进化的基准。

除了以上提到的外,在软件进化中还存在一些其他问题,如随着软件的进化,如果没有主动的策略,软件系统的质量会逐渐下降,这在很大程度上是由外部的原因导致的(比如经济压力等)。再如一些技术方面的问题等。因此,对于存在的这些问题,需要基本的关于形式和理论的科学研究来帮助理解、分析和管理软件的改变。需要语言、工具、方法和技术的发展来提供对于在软件开发过程中软件改变的显性的支持,这是需要人们做大量研究工作的。

6 结 论

尽管人们对于软件工程的研究使得软件的开发从各个方面有了提高,但是,软件的质量仍然达不到人们的期望,软件要不断面临新的需求和环境而改变。因此,软件进化一直是人们研究的一个重要领域。笔者简单介绍了软件进化的几个方面,由于进化及其过程涉及到了很多的方面,因此,对其的研究必须覆盖各个方面,如理论模型的研究、过程模型的进化、技术工具方面的研究等。同时,人们应该使其在传统软件开发过程中得到重视,对其提供各种显示的和直接的支持研究。

参考文献:

[1] Bennett K. Software evolution: past, present and future[J]. Information and software technology, 1996,38:673 - 680.

[2] Lehman M M, Ramil J F. Software evolution - Background, theory, practice [J]. Information Processing Letters, 2003, 88(1/2):33 - 44.

[3] Lehman M M, Ramil J F. Software evolution and software evolution process[J]. Annals of software Engineering, 2002, 14:275 - 309.

[4] Belady L A, Lehman M M. A model of large program development[J]. IBM Systems Journal, 1976,15(1):225 - 252.

[5] Lehman M M. Programs, Cities, Students, Limits to Growth? [J]. Imperial College of Science and Technology Inaugural Lecture Series, 1974,9:211 - 229.

[6] Lehman M M, Ramil J F. Tutorial on Software Evolution: its Source, Nature and Control[C] // ICSM 2002. Canada:[s. n.],2002.

[7] Lehman M M. Programs, lifecycles and the Laws of Software Evolution[J]. Proc. IEEE,1980,68(9):1060 - 1076.

[8] Lehman M M. Laws of Software Evolution Revisited[C] // EWSPT. France:[s. n.],1996.

[9] Lehman M M. Feedback in the Software Evolution Process [C] // CSR Eleventh Annual Workshop on Software Evolution: Models and Metrics. Dublin:[s. n.],1994.

[10] Lehman M M, Ramil J F, Kahen G. Evolution as a Noun and Evolution as a Verb[C] // SOCE 2000 Workshop on Software and Organization Co - evolution, Imp. Col., London:[s. n.],2000.

[11] Ward M, Bennett KH. Formal Methods to Aid the Evolution of Software[J]. International journal of software engineering and knowledge engineering, 1995, 5(1):25 - 47.

[12] Heisel M, von Schwichow C. A method for guiding software evolution[C] // The IASTED International Conference on Software Engineering. Innsbruck, Austria:[s. n.], 2004.

[13] 唐亚哲,陈传峰,李增智. 基于开放实现与反射的软件进化模型[J]. 小型微型计算机系统,2003,24(11):1978 - 1981.

[14] Mens T. Challenges in Software Evolution[C] // ECRIM - ESF workshop ChaSE. Bern, Switzerland:[s. n.], 2005.

(上接第 195 页)

实例化(预计算并存储结果)Cube 中的部分节点以提高 OLAP 性能的解决方案。

利用决策树分类方法[4]和 Bayes 分类器[5]等数据挖掘方法对数据仓库进行挖掘分析,得出有用的规则,用来改进企业的软件过程、对正在实施的项目进行更合理的评估和支持企业的决策分析。

3 结束语

问题跟踪工具为软件企业的软件开发过程中问题的解决提供一个良好的平台,问题分析系统为企业提供了有用的决策信息。问题跟踪和问题分析系统对提高软件生产率和软件质量起到了一定积极有效的作用。相信随着该系统的不断完善,高质量的分析型数据的日积月累,问题分析系统将会发挥越来越大的作用。

参考文献:

[1] Lonchamp J. A Structured Conceptual and Terminological Framework for Software Process Engineering[C] // In 2nd International Conference on Software Process (ICSP2). Berlin, Germany:[s. n.],1993:41 - 53.

[2] Grimshaw A S. Easy - to - Use object - oriented parallel processing with Mentat[J]. IEEE Computer, 1993,26(5):39 - 51.

[3] 于 波,赵 征,唐世渭. OLAP 中的 CUBE 计算问题[J]. 计算机应用,2003,23(1):1 - 3.

[4] Han Jiawei, Kamber M. 数据挖掘:概念与技术[M]. 北京:机械工业出版社, 2001:188 - 195.

[5] 边肇琪, 张学工. 模式识别[M]. 北京:清华大学出版社, 2002:9 - 43.