一种分布式网络环境下虚拟结对编程的设计方案

洪 奎, 窦万峰

(南京师范大学 计算机科学与技术学院, 江苏 南京 210097)

[摘要] 为有效地支持软件分布式开发, 提高软件协同开发环境的易用性和有效性, 从协同理论和应用工具方面入手, 同时鉴于结对编程在软件协同开发应用中的实际价值以及在应用中出现的问题, 提出一种基于分布式网络环境下的虚拟结对编程方案. 该方案基于 Microsoft Windows平台和 Socket通信技术, 通过 C/S方式实现网络上两台机器之间通信.

[关键词] 软件工程, 结对编程, 协同开发

[中图分类号] TP311.5 [文献标识码] A [文章编号] 1672-1292(2010)01-0072-04

A Design of Virtual Pair Programming Based on Distributed Network Environment

Hong Kui, Dou Wanfeng

(School of Computer Science and Technology, Nanjing Normal University, Nanjing 210097, China)

Abstract In order to effectively support the distributed development of software, improve ease of use and effectiveness of software development environment, this paper, starting with theory of collaboration and application tool, and in view of the real value of pair programming in software collaborative development, as well as the problems in application, proposed a virtual pair programming program based on distributed network environment. The program is based on Microsoft Windows platform and Socket communications technology, through C/S approach to achieve network communication between two machines.

Key words software engineering; pair programming; collaborative development

结对编程(Pair Programming)是极限编程(Extreme Programming)的 12个实践之一. 不同的研究人员对结对编程进行的实践得出了不同的评估结果. 这主要是因为结对编程属于新方法学, 很多说法难以进行定量的评估. 但是, 不可否认, 采用结对编程带来很多好处: 代码具有更高的质量; 结对可以最大化地提高工作效率; 系统的每一部分至少有两个人熟悉详细的设计细节, 降低了团队成员变动给项目带来的风险; 团队成员间的合作更密切[1-4]. 但是结对编程是新的方法学, 在实践过程中难免会发生一些意想不到的情况: 坐在后面的人不习惯写代码人的代码风格, 导致写代码的人心里压力变大; 分布式环境下无法设施; 后面坐着的人跟不上写代码的人的思路, 写代码的人要不断对其讲解.

由于目前越来越多的项目都是交由区别地理位置的员工组成的虚拟团队来完成的, 很多开源软件都是由分布在世界各地的开发者共同完成的[5], 这使得结对编程很难应用到这样的虚拟团队. 为了有效地支持软件分布式开发, 提高软件协同开发环境的易用性和有效性, 本文提出一种基于分布式网络环境下的虚拟结对编程方案. 该方案基于 Microsoft Windows平台和 Socket通信技术, 通过客户/服务器方式实现网络上两台机器之间通信, 可有效地解决由于地理分布而造成的一些问题.

1 系统概述

依据结对编程的定义以及软件协同开发理论相关知识, 该方案需解决以下 4个重要的问题: 交流, 感知, 协作, 共构. 其工作流程示意图如图 1所示, 主要有两个模块: 实时语音通话和协同编程. 在语音通话模

收稿日期: 2009-06-12

通讯联系人: 窦万峰, 博士后, 副教授, 研究方向: 分布协同软件工程. E-mail: douwanfeng@njnu.edu.cn

块中, 针对两个用户, 以 P2P方式实现, 需要邀请对方, 并获得对方响应, 实时采集、传输、播放语音, 任何一方都可停止语音聊天 (此时, 另一方也会退出). 语音通话实现了交流的功能. 在协同编程模块中, 同样针对两个用户, 以 C /S方式实现, 实施过程中需要客户机邀请服务器, 并获得服务器响应, 由此就获得服务器的桌面, 针对编辑控件, 实时传输双方的光标位置和模拟键盘按键 (每次用户行为). 协同编程实现了感知、协作以及共构的功能.

2　实现方案

2 1　语音通话模块实现

语音通话模块对模拟语音信号经过模数转换, 进行编码压缩后, 按一定的打包规则将压缩帧转换成 IP数据包通过数据网进行传输, 在目的地经过数据解压、数模转换复原成话音, 从而达到语音通信的目的.

由于通话双方处于对等的地位, 因此语音通话模块必须将服务器端和客户端集中到一起, 因此语音通话模块必须采用 P2P方式实现. 语音通话模块工作流程如图 2所示.

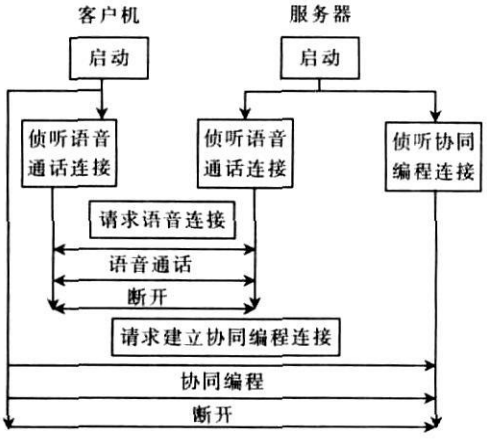

图 1　系统工作流程示意图

Fig.1　Workflow diagram of the system

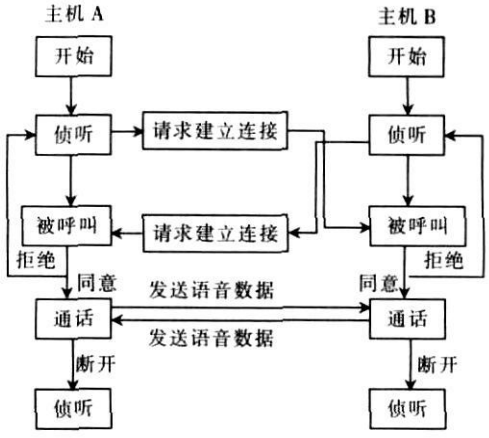

图 2　语音通话模块工作流程示意图

Fig.2　Workflow diagram of the voice module

在 W indow s中, 很容易实现音频采样和播放. 语音采集模块将模拟语音信号转换为数字信号, 在程序中为了减少通话的延迟时间, 采用直接分配内存块, 即将采集的数据放在内存块中, 以便压缩或传输, 而不是存为语音文件. 而播放语音时, 也是直接播放语音数据, 这样的好处是省略了读写硬盘的费时操作, 提高了语音通话的实时性.

该模块采用 TCP / IP协议来进行数据语音的网络传输, 并将服务器端和客户端集中到了一起. 分析图 2, 对于主机 A 传送到主机 B的音频数据信息这条数据通路, 主机 B 作为服务器端; 对于主机 B 传送到主机 A 的音频数据信息这条数据通路, 主机 A 作为服务器端. 因此双方通话时, 主机 A 和主机 B同时处于服务器状态又处于客户端状态, 当主机 A 呼叫主机 B时, 主机 A 就进入客户端状态, 主机 B收到呼叫信息, 进入服务器状态, 若主机 B同意通话, 则两机都进入通话状态, 双方可以通过上述两条数据通路进行网络通话, 若其中有一台机器挂断, 则两机都返回侦听状态, 等待呼叫或被呼叫.

2 2　协同编程模块实现

要实现软件分布式协同开发, 根据协同理论和极限编程的 12大实践, 该开发环境必须要实现 3个重要功能: 协同感知, 协作, 协同编辑 (共构).

协同感知的目的是模拟现实世界的协作过程, 让参与结对人员在网络环境下感知对方的活动, 从而为自己的活动提供一个上下文 (context)环境, 消除由于空间上的分布而带来的割裂感, 更好地发挥个体的主动性和积极性[6]. 根据应用的需求, 必须提供多层次的感知信息. 首先分布式结对编程要求参与结对编程的双方必须能感知到对方的状态. 其次分布式结对编程要求参与结对编程的双方都能对文档的进度有共同的掌握, 对于一方正在编写的代码, 另一方能够及时地感知. 这样才能使协作的过程更自然. 极限编程的一个重要实践就是团队代码, 因此协同感知是分布式结对编程的重要组成部分.

协作和协同编辑在分布式协同开发中表现为协同编程. 结对双方可以同时打开同一份文件, 并可以同时对文件进行编辑. 实时协同编辑要保证系统在具有比较好的响应时间和通知时间的前提下, 复制到各站点处的数据对象一致, 实现用户之间方便、灵活、自由地交互. 协同编程允许不同或相同时间不同地点的用户协同完成代码的编辑[7], 从而大大提高群体工作的效率. 工作模式如图 3所示, 该模块使用 P2P技术, 支持在虚拟的协作空间中进行协同编辑操作以及协同感知. 并发控制算法是协同编辑的核心, 主要解决两人同时编辑同一代码时保证代码的一致性. 一般区分为悲观的和乐观的并发控制两种, 其中基于版本控制的乐观并发控制算法被认为是最有希望实现自由流畅交互的并发控制算法[8].

图 3　协同编程模块工作模式示意图

Fig.3　Work model diagram of collaborative programming module

2.2.1　协同感知和协作

现实生活中的结对编程首先需要人与人之间的交流, 这种私下的交流非常必要[9]. 支持这种两两间的对话, 可以明确感知对方的状态, 同时也方便了解文档的开发进度. 其次由于本设计采用中央代码库的方式管理文档, 一个工程的所有文档都要在中央代码库有一个最新版本的备份, 由中央代码库将文档分发给下属的两个或多个节点, 在节点　节点间形成对等控制方式. 每个节点都保存一定时期内的所有版本, 但只将最新版本提交给中央代码库. 节点可以对前期版本提出查询申请, 在这种方式中集中地体现了 CSCW 中的协调性, 同时也能让参与结对双方感知文档的最新状态. 采用这种方式的优势在于减少了域中心节点层之间的数据交换, 从而避免了不必要的网络拥塞, 而前提是各模块之间要具有相互的独立性. 对于简单模块的代码, 参与结对编程的双方可以通过实时远程获得对方屏幕的办法感知开发的进度, 配合语音模块可以进行实时交流. 对于需要另一方的协作的情况, 本地机器可以将控制权交给远端.

2.2.2　协同编辑

实时协同编辑要保证系统在具有比较好的响应时间和通知时间的前提下, 复制到各站点处的数据对象一致, 实现用户之间方便、灵活、自由地交互. 通常协同编程采用 3种方式管理共享数据: 集中式体系结构、复制式体系结构和混合式体系结构. 这 3类体系结构下的协同编辑器在设计思想、应用场合、传输机制上都有着很多不同.

Patterson模型是第一个描述实时协同系统的结构模型, 它将实时协同系统分为 4层: 用户界面层(Display)　直接和用户交互; 视图层(View)　包含显示层的逻辑表示; 数据模型层(Model)　与协同应用语义相关, 描述协同系统与应用相关的共享状态和过渡状态; 文件层　描述协同应用系统的持久状态. Patterson模型可以说明哪几层集中驻留在服务器, 哪几层复制分布在用户站点, 以及分布复制层之间的同步关系. 图 4表示 Patterson模型分别描述集中、复制和混合等 3种协同系统体系结构. 在图 4(a)所示的状态集中共享结构中, 模型状态和文件状态是共享的, 驻留在中央服务器; 而两个用户的视图和显示界面不同步, 对于相同的共享数据, 给用户提供独立的视图. 图 4(b)所示的状态同步复制结构, 所有层都分布在用户站点, 用同步机制实现数据状态和视图同步, 给用户提供完全一致的共享信息和视图. 用户将可以看到完全相同的视图和窗口界面　也称为严格 WYSIWIS. 而在图 4(a)中, 对于相同共享状态, 用户看到的视图可以有所不同. 图 4(c)是混合结构, 其中模型是共享的, 而视图是同步的, 这种结构提供了保证模型状态一致的简单机制, 且允许用户根据需要切换视图共享或不共享.

结对编程主要是面对满足两个人同时编程的实践, 综合 3种协同体系结构的特点, 协同编辑模块采用集中式体系结构, 集中式体系结构将文档数据存放在一个中央服务器. 其工作流程如图 5所示. 客户端通过网络与此服务器应用程序完成计算功能后将共享数据发送给各个客户端, 这样每个客户本地的共享数据显示得到更新. 集中式体系结构下只有共享文档的一个副本, 因此基于版本控制的乐观并发控制算法比较简单, 易于实现存取管理和保证数据的一致性. 同时, 因为服务器能够实时地将某个作者对文档的修改反映到所有协作参与者, 开发人员可以进行虚拟结对编程, 而不需要真正地坐在一起. 同时实现基于代码的协同工作, 两个开发人员可以同时编辑一个源代码文件, 一个人对源代码做的修改会被实时地反应到另外一个人的视野里, 这保证了两个人看到的确实是同一份源代码文件, 就像两人坐在同一机器前一样, 而

实际上两人确是各自坐在自己的座位前,用着自己的开发工具.因此可以真正地实现群件系统中的 你见即我见 .

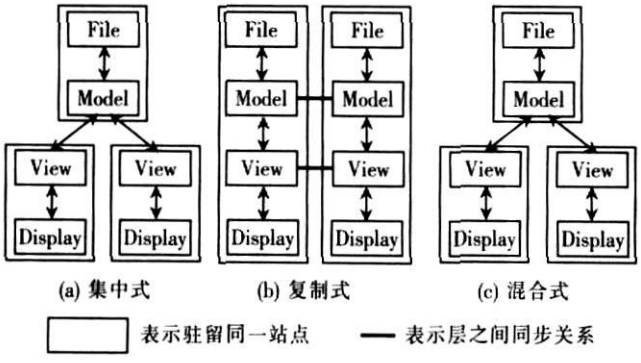

图 4　Patterson 描述的 3 种协同系统体系结构

Fig.4　Three kinds of collaborative system architecture which Patterson described

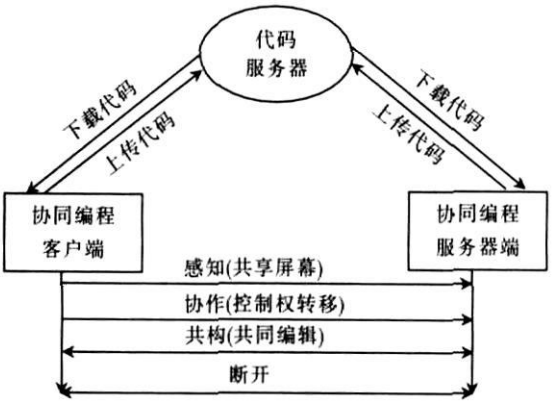

图 5　协同编程模块工作流程示意图

Fig.5　Workflow diagram of collaborative programming module

3　结语

本文在研究现有结对编程优势和应用中遇到的问题的基础上,提出了一种新的基于网络环境下结对编程方案,并且基于该方案设计了一个分布式结对编程协同开发系统原型,该系统可以为分布式虚拟结对编程提供支持.协同编辑的设计和实现是本文关注的另外一个焦点.本文分别对协作通信、会话管理、协同感知以及系统构建的技术进行了研究.在协作通信技术上,采用 Socket通信技术,并在此基础上设计了一个可进行消息可靠传输的机制.会话管理是协同编辑中要解决的另一个问题,本文设计了一个基于发起人模式的会话管理,该会话管理能够支持一种特定的分布式结对编程模式.

[参考文献] (References)

[1] 史美林. 计算机支持的协同工作　概念、技术、应用[M]. 北京:电子工业出版社, 2000: 12-120.
ShiMeilin. Computer Supported Collaborative Work　Conception, Technology and Application[M]. Beijing: Electronics Industry Press, 2000: 12-120.

[2] PeteMcBreen, Kent Beck. Questioning Extreme Programming[M]. Beijing: Pearson Education Press, 2002: 5-17.

[3] 陈斐. 结对编程技术[M]. 北京: 机械工业出版社, 2004: 20-100.
Chen Fei. Pair Programming[M]. Beijing: Machinery Industry Press, 2004: 20-100. (in Chinese)

[4] StapelK, LubkeD, KnaussE. Best practices in extreme programming course design[C]// Proceeding of the 30th International Conference on Software Engineering. New York: ACM, 2008: 769-775.

[5] LaurieWilliams, Robert Kessler. Pair Programming Illuminated[M]. Beijing: Machinery Industry Press, 2004: 70-108.

[6] Toll T V, Roger Lee, Thomas Ahlswede. Evaluating the usefulness of pair programming in a classroom setting[C]// 6th IEEE/ACIS International Conference on Computer and Information Science. Los Alamitos: IEEE Comp Soc Press, 2007: 302-308.

[7] Mendes E, AL-Fakhri L, Luxton-Reilly A. A replicated experiment of pair programming in a 2nd-year software development and design computer science course[J]. SIGCSE Bulletin, 2006, 38(3): 108-112.

[8] The ACM-ICPC International Collegiate Programming Contest[EB/OL]. [2005-11-26] http://icpc.baylor.edu/icpc/.

[9] Vidot N, Cart M, Ferrie J, et al. Copies convergence in a distributed real time collaborative environment[C]// Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work. New York: ACM, 2000: 171-178.

[责任编辑: 严海琳]