

软件工程中的几个热点问题

余 金 山

(华侨大学信息科学与工程学院, 福建 泉州 362011)

摘要 讨论软件工程中的几个热点问题,研究其现状和发展方向. 内容包括:(1) 软件体系结构;(2) 软件重用;(3) 网络软件工程;(4) 分布计算和分布对象技术;(5) 软件工程教育.

关键词 软件工程, 软件体系结构, 软件重用, 网络软件工程, 分布计算, 分布对象技术, 软件工程教育

中图分类号 TP 311.5

文献标识码 A

软件工程是计算机学科中一个年轻且充满活力的研究领域. 自 20 世纪 60 年代末以来,人们为了克服“软件危机”在这一领域做了大量的工作,逐渐形成比较系统的软件开发理论、技术和方法. 它们在软件开发实践中发挥了重要作用. 今天,现代科学技术将人类带入信息社会. 计算机软件扮演着十分重要的角色,软件工程已成为信息社会高技术竞争的关键领域之一. 本文就当前软件工程中,有关软件体系结构、软件重用、网络软件工程、分布式计算和分布对象技术,以及软件工程教育等几个热点问题进行讨论,并介绍其研究现状和发展方向.

1 软件体系结构

1.1 所谓软件体系结构

何为软件体系结构,至今尚无一个能被广泛接受的定义. 但是,我们可以从下面几个方面加以理解.

(1) 软件体系结构,包括一个或一组软件部件(Components)、软件部件的外部的可见特性及其相互关系.
(2) 一个系统可以包括多于一个结构,而且没有哪一个可以说自己就是完整的软件体系结构. 软件体系结构本身并不说明什么是部件、什么是部件的相互关系. 某个软件部件可以是一个对象、一个进程、一个文档、一个商业产品、一个数据库或其他更广泛的概念.
(3) 描述部件的信息大致有 4 类:计算功能、额外功能特性、结构特性和家族特性. 计算功能就是指部件所实现的整体功能. 额外功能特性描述了部件的执行效率、处理能力、环境假设和整体特性等方面的要求. 它们大都可对时间要求、空间要求、精确度、安全性、保密性、带宽、吞吐率、最低软硬件要求等,进行定量描述. 结构特性描述了特定部件如何与其它部件组织在一起,以构成整个系统的信息. 这种特性是软件体系结构中最重要. 有关它的比较重要的方面是部件类型和部件所支持的相互作用. 家族特性描述了相似部件之间的关系. 因此,体系结构设计要考虑的问题包括^[1]:在总体结构上,如何把系统组织成部件之间的集成;全局控制结构;通讯协议、同步和数据存取;给设计元素分配功能;设计元素之间的组装;物理分布;伸缩性和性能问题;不同设计方案间的选择等. 有代表性的、著名的软件体系结构,包括(1) Pipes and Filters,(2) Layered Systems,(3) Virtual Machines,(4) Client/ Server Systems 和(5) 以数据为中心的系统. 例如,仓库系统、超文本系统、黑板系统等.

1.2 体系结构的重要性

上面列出几类大家至少有些熟悉的体系结构. 根据它们的影响和作用,我们已经能基本上体会到体系结构的重要性. 随着软件系统大小和复杂性的增加,在软件设计过程中人们所面临的问题不再是考虑

收稿日期 2003-11-09

作者简介 余金山(1952-),男,教授,主要从事软件工程的研究. E-mail: jinshanyu@hqu.edu.cn

基金项目 福建省自然科学基金资助项目(A0210018)

软件系统的功能问题,而是面临要解决更难处理的非功能性需求.比如,系统性能问题、可适应性问题、可靠性问题、可复用性问题等等.要在系统设计的最初阶段决策系统设计的总原则和确定整个系统的总体框架,以指导系统设计的开展,保证开发工作能达到项目的预定目标.同时,能在软件系统的整个生命期中,保持系统体系结构可以很方便地进行维护和调整,以适应所发生的变化.

1.3 当前对软件体系结构的研究

(1) 体系结构描述.表达体系结构的符号系统与语言、表达方式是研究的重要内容.这种描述语言的目的就是提供一种规范化的体系结构描述,从而使得体系结构的自动化分析成为可能.(2) 提供体系结构设计技术的形式化的基础.它的目的是对体系结构设计人员在实践过程中,总结出来的一些设计的经验和方法加以总结、概括.从而,形成一个形式化的描述,形成一定的理论基础,以代替当前的不精确的研究.(3) 研究体系结构的分析技术.通过分析来预见软件的质量,通过分析来创建、选择、评估与比较不同的体系结构.(4) 设计规则与开发方法的研究.体系结构设计作为一种设计方法,它应该能够被推广从而被更多的人所掌握.(5) 设计工具和环境.软件体系结构设计既然作为软件工程的一部分,它的计算机辅助实现手段是相当重要的.我们应当开发出一些软件工具来实现体系的描述和分析.(6) 体系结构的发现与再造工程.从现有的系统中总结出系统的体系结构.为了提高其性能需要找出不相容的部件,对其进行桥接,或改变其体系结构.使之达到新的水平,并为一些特定领域的应用提供一些设计框架.(7) 体系结构的分类、编目及使用指南.(8) 实例分析.

2 软件重用技术

软件重用指重复利用现有的且已验证过的代码、设计、需求规格说明、测试数据与测试过程,以及各种有关的规范文档来构造或集成新的软件系统,

而不是诸事都从头做起.软件重用模型 SRM 是一种以可重用软件库为核心,把重用技术应用于整个软件生命周期的模型.图 1 给出了 SRM 的一种典型示例.在重用库的支持下,用重用技术实现快速原型方法是当今较为常见的另一种软件重用模式.软件重用的思想最早可追溯到 50 年代提出的所谓“黑箱重用”和“白箱重用”.标准子程序可看作是软件重用的最典型的例子和具有代表性的方法之一.重用方法在其它工程领域中的应用是十分普遍且极为成功的,即使在计算机硬件工程中也是如此.由以上简单的分析即可看到软件重用对软件生产的明显的、革命性的作用.因此,软件重用被称为是改进软件生产过程与其最终产品的最有前途的办法.软件重用的意义是突出的、深远的,但存在的问题也是尖锐的.软件重用在技术、管理和决策方面都存在许多未能解决的难题.

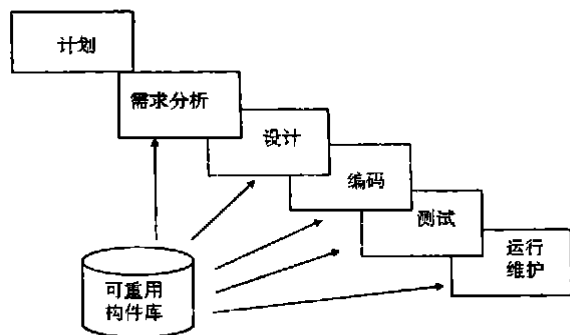


图 1 软件重用模型

由以上简单的分析即可看到软件重用对软件生产的明显的、革命性的作用.因此,软件重用被称为是改进软件生产过程与其最终产品的最有前途的办法.软件重用的意义是突出的、深远的,但存在的问题也是尖锐的.软件重用在技术、管理和决策方面都存在许多未能解决的难题.

2.1 技术难题

(1) 构造和识别技术.即如何构造或如何识别可重用的软件成分.(2) 分类和查找技术.对已建造出的可重用构件应按什么方法、标准进行分类和组织,以及如何有效地从重用库中找出完全符合或最大程度上符合问题要求的构件,进行讨论.(3) 理解与修改技术.对需要进行适当修改方可重用的软件,如何对原构件的结构、功能、风格等特征进行理解,并在理解的基础上如何进行修改.(4) 组装与合成技术.如何把构件组装成或与新开发的子系统合成为所需要的目标系统.

2.2 管理与决策问题

(1) “两难”问题.即如何解决通用性与处理能力的矛盾、构件大小与可重用潜力的矛盾、建立可重用库的投资和重用收益的矛盾.(2) 标准化问题.(3) 配置管理问题.这也是一个颇难解决的大问题.例如,是否必要把每个可重用构件作为一个配置项(CSCI)来管理,对同一构件的不同裁剪版本又该如何管理,等等.软构件技术是支持软件重用的核心技术,也是目前实现软件重用的现实的和主流的途径.它已

被提升到一定的高度,加以认识 CBSE(Component-Based Software Engineering)^[2].

3 网络软件工程

Internet/ Intranet 的发展与普及一方面给软件工程带来了某些积极作用^[3],另一方面也给软件工程提出了许多新的问题.认识这些问题,分析并解决这些问题具有更加重要的意义.(1) 现有系统/ 信息环境与 Internet/ Intranet 的联接/ 集成. 这个问题或许是近期最迫切需要解决问题. 信息技术发展到现在,或更准确地讲,到 Internet/ Intranet 的广泛应用时为止,各个行业,各个部门都或多或少,或大或少地建立了某些计算机应用系统.如何才能平滑,高效地把现有系统与 Internet/ Intranet 集成具有高度的现实意义、在技术上涉及的面也很广.(2) 安全性问题. Internet 是一个全球性的互联网,它不但对安全性有更高的要求,而且由于它的软/ 硬环境异常的复杂和多样化,从而使得已有的许多安全技术已不安全.因此,研制适用于 Internet/ Intranet 的安全机制是另一个迫切需要的问题.(3) 基本支撑软件的选择问题.最基本的支撑软件,包括网络协议、浏览器、服务器和应用软件开发平台.(4) 性能问题.(5) 标准与技术更新的矛盾. 这个问题包括两个方面,一是与 Internet/ Intranet 直接相关的新概念、新技术的规范化与标准化问题,二是一些已制定的规范与标准在 Internet/ Intranet 环境下可能已不合适,必须重新制定.(6) 用户与开发者的关系问题.传统的软件开发模式是用户提出软件需求,软件人员按照需求开发软件.开发完后,再把该软件交给用户使用.这种模式存在的致命问题——用户需求的不可满足问题,以及由此带来的软件维护的困难性与开销、人员的紧缺等等问题,在基于 Internet/ Intranet 的应用中,更加突出.文[4]认为,在 21 世纪中,很快即将形成一种以用户为控制中心的程序设计新时代.该文的作者还认为,软件工程的基本概念应重新颠倒过来,称为软件艺术.在这种模式中,软件艺术师(生产者 producer)建造针对具体应用领域的基本框架(Framework)、公用部件(Component)和 Applet,而让用户(消费者 Consumer)根据需要裁剪/ 装配自己的系统.如果能做到这点,那当然是天大的好事.但是存在一个根本问题:我们有没有办法开发出装配起来“足够容易”的软框架、软部件和 Applet,以及我们将要求用户裁剪/ 装配系统时应具有什么知识和技巧简单地讲,这存在的问题仍然是用户和开发者的关系问题.(7) 体系结构.基于 Internet/ Intranet 的应用,显然不能照搬传统软件的体系结构.这也涉及应该采用什么样的体系结构.(8) CASE.基于 Internet 的 CASE,至少在下列 3 个方面有别于传统的 CASE.(a) 它本身是基于 Internet 的.(b) 能支持协同工作(Cooperative Work),特别地要能支持跨地域的协同工作.(c) 能充分利用 Internet 的优点和网上资源.(9) 软件过程模型.基于 Internet 的应用,或称网络计算(Network Computing)及其开发模式与传统软件与传统开发模式,具有许多截然不同之处.这就存在传统的过程模型是否还能运用,以及合适的过程模型又是什么的问题而值得探讨.

4 分布计算和分布对象技术

4.1 分布计算/ 分布系统

分布计算或称分布系统是近 20 年来影响计算技术发展的最活跃的因素之一.一个分布式计算系统是由许多计算资源,以一定的互联方式组成的、开放、多平台的、可互操作的、合作的系统.它能够为用户提供一定范围的服务.或者说任何用户只要具有关于所用系统的最基本的知识,就可以在任何地点、任何时候,以应用目的可接受的响应时间,访问并使用系统中的任何资源,得到所需要的服务.分布计算至少具有如下几点优势.(1) 系统性能可提高几个数量级.(2) 计算资源、用户信息共享.(3) 与应用本身的自然结构和需求相适应.例如,银行、连锁店等本身就是分布的,工业生产的体系结构也从树形变为网状,贸易的全球化,等等.因此,分布系统是发展的必然.

4.2 互操作性

分布系统必须解决两个最基本的问题.(1) 逻辑互联方式.用什么样的逻辑互联方式,实现进程间的通讯.也即,不同的软件和硬件平台上的用户应该如何通信、共享信息.(2) 系统的互操作性.所谓两种或多种资源是可以互操作的,是指它们能够交互地共同执行任务,互相提供服务.除此之外,一个开放的分布系统还面临着如下两个难题.(1) 高度异构的分布环境.(2) 系统管理和实用化支持.(a) 安全

性、可靠性、效率等系统性能问题. (b) 公共设施,包括用以支持分布式应用系统的开发、使用和维护的基本工具,基本的系统及服务 (GUI,信息管理、系统管理、电子邮件等). (c) 系统的可伸缩性 (Scalability) 等. 传统的远过程调用和 C/S 模型在应付这些复杂问题时已是力不从心. 研究和实践证明,解决这些问题的合适且有效的办法是采用分布对象技术 (DOT) 和中间件技术.

4.3 DOT 与中间件 (Middleware)

DOT 的主要思想是在分布式系统中,引入一种可分布的、可互操作的对象机制,并且把分布于网络上可用的所有资源看作公共可存取的对象集合. 它使得不同的对象可以集成存在一起. 此外,一个对象客户能通过定义在分布对象模型上的接口,访问分布系统的可用对象. 通常将这些可用对象称为对象组件 (或构件或部件). 分布对象存在于网络的任何地方,可被远程客户应用以方法调用的形式访问. 至于分布对象是用何种语言和编译器所创建,对客户对象来说是透明的. 客户应用无须知道它所访问的分布对象在网络中的具体位置以及运行在何种 OS 上. 分布对象组件的设计目标是使得用户和开发者能象使用硬件那样“即插即用”. 由于对象自身并不能提供这样的内部结构,使得不同厂商提供的软件可以在同一地址空间相互作用,更不用说跨越不同地址空间或网络的交互了. 解决的办法是为对象组件提供标准底层运行环境的支持,即对象总线 (也称软件总线) 和一些系统服务组件——中间件图 2. 中间件是一

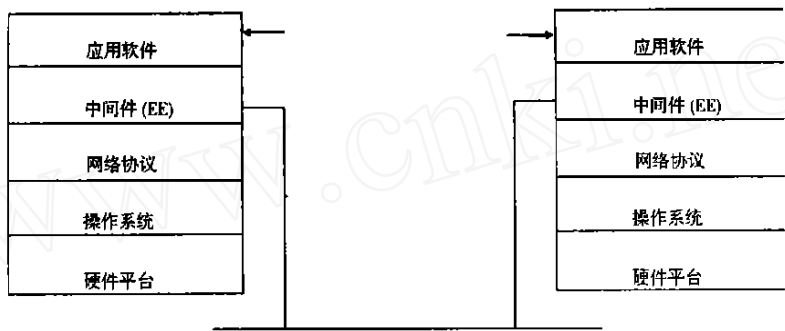


图 2 中间件作用示意图

种支撑性软件. 一个中间件应具有以下两部分^[5]. (1) 执行环境 (EE). 网络上各节点安装了 EE 软件,各节点上的应用软件之间就可协同工作. 也即 EE 使得在其以下各个层次的异构性,对应用软件而言成为透明的. 所以 EE 是实现互操作的关键,是中间件的主体. (2) 应用开发工具 (AD). 应用软件要能透明地动用合作方的资源,该软件中当然要有作出此种动用的指示. 为此必定要有一组工具,备有专用语言和有关的编辑器,它们可用来开发含“透明动用对方成分”的应用软件. 所以 AD 工具是一个完善的中间件必备的部分.

4.4 当前的主流技术及其发展

表 1 所示,给出了分布计算的 3 个发展阶段. 当前主流技术: (1) CORBA (Common Object Request Broker Architecture); (2) Java/ RMI; (3) Active/ DCOM; (4) ODP (Open Distributed Processing, ISO/ ITU).

表 1 分布计算的不同年代比较表

项 目	第一代 (20 世纪 80 年代 ~ 90 年代初)	第二代 (20 世纪 90 年代)	第三代 (21 世纪初)
面 向	信息 (资源) 共享	异构环境下的应用互操作	智能化协同工作
体系结构	经典的 C/S 模型	OO 多层 C/S 模型	多 agent 模型
关键技术	传统技术与设施	OO 技术	面向 agent 的拟人化交互环境

5 软件工程教育

鉴于软件的重要性和软件开发的特殊性,国际上的一些著名大学、研究机构和学术团体,特别是 IEEE,ACM 和美国的 SEI 研究所,从 20 世纪 90 年代初就开始对软件教育的研究,并积极促使软件工程成为独立的专业. 在国内,我们也曾在一些重要的报刊、核心杂志和全国学术会议上发表了一系列关于软件工程教育的文章 (有些曾被多方转载).

软件工程学科的逐渐成熟、社会的需要和软件产业的发展再次表明,软件工程从研究生教育普及到

本科教育和职业教育的必然趋势^[6]。现在,国内外的许多大学都纷纷开设了软件工程的本科专业。

由于软件工程是一门新兴的学科,而且有它的许多特殊性。因此,有关软件工程教育的问题,目前仍备受重视。其中,最值得关注是由 IEEE 和 ACM,于 1997 年联合组织的负责研究软件工程教育的专门机构 SWECC(Software Engineering Coordination Committee)的研究项目和工作成果。到目前为止,在开展并已有阶段性成果的工作有 3 项:软件工程专业课程计划,软件工程知识体,软件工程道德法律和专业实践。其总体研究项目称为 SWEEP(Software Engineering Education Project)。

SWEEP 从教师、教学计划(课程设置)、教学环境(实验室和计算机资源)、教学对象(学生)和实施制度等方面,制定了软件工程专业认证标准。该标准主要针对软件工程本科学位计划。

SWEBOK 项目的目的是为了建立一组标准和规范,作为软件工程专业实践中进行决策、专业认证和制定教育课程计划的基础。该目标包括 5 点^[7]。(1) 促进国际上对软件工程的一致理解。为此目的, SWECC 聘请了 42 个国家的近 500 名专家,承担了该项目的评审工作。(2) 确定软件工程相对于其它学科,如认知科学和人类因素、管理和科学、计算机科学、工程管理、计算机工程和数学等的地位和范围。(3) 描述软件工程学科内容和特征。(4) 提供获取软件工程知识体相关信息的目录。(5) 提供软件工程专业课程计划的开发、个人专业证书和专业许可的依据。

6 结束语

软件技术的迅猛发展以及它和社会环境的高度相关性,要求我们必须密切注意软件工程的发展动态和前沿。本文讨论介绍了软件工程中的若干热点问题。所给出的热点不是全面的,但其有代表性。深入地讨论研究将带出其它的热点问题。本文给出的每个热点问题也不是孤立的,而是互有联系的。

参 考 文 献

- 1 Shaw M, Carlan D. Software architecture: Perspectives on an emerging discipline[M]. Englewood Cliffs: Prentice Hall, INC., 1996. 1 ~ 32
- 2 Browns A W, Wallnau K C. The current state of CBSE[J]. IEEE Software, 1998, 15(5): 37 ~ 46
- 3 余金山. 基于 Internet 的软件工程[J]. 计算机系统应用, 1999, (2): 29 ~ 34
- 4 Lewis T. The next 10 000₂ years: part II[J]. Computer, 1996, 29(5): 78 ~ 86
- 5 刘锦德, 唐雪飞. 开放系统中互操作技术的发展和前景[J]. 计算机科学, 2000, 27(10): 27 ~ 30
- 6 Saiedian H. Software engineering education and training for the next millennium[J]. The Journal of System and Software, 1999, 49(2/3): 133 ~ 115
- 7 白 征. SWEBOK: 软件工程知识体[J]. 计算机科学, 2001, 28(7): 108 ~ 111

Several Hot Spots of Software Engineering

Yu Jinshan

(College of Info. Sci. & Eng., Huaqiao Univ., 362011, Quanzhou, China)

Abstract With regard to following hot spots of software engineering, the author presents the present state of studies and the orientation of development. These hot spots include: (1) software architecture (2) software reuse; (3) network software engineering; (4) distributed computing and technology of distributed object; and (5) software engineering education.

Keywords software engineering, software architecture, software reuse, network software engineering, distributed computing and technology of distributed object, software engineering education