计算机科学 1998 Vol. 25 №. 2

齐心协力

——1997年国际软件工程会议概况

Pulling Together

郑国梁　　邵维忠

（南京大学计算机科学技术系　南京 210093）（北京大学计算机科学技术系　北京 100871）

第19届国际软件工程会议于1997年5月17至22日在美国波士顿召开。来自世界各地30个国家和地区的代表，近900人参加了会议。会议的主题是“齐心协力”(pulling together)，是针对去年在柏林召开的第18届会议上，出现软件工程领域的“学院派”和“工程派”之争而提出的。这次会议的“齐心协力”意味着为了达到共同目标需要各方力量共同奋斗；也表明不同观点、不同基础和能力聚集一起，来寻找达到合作的途径和共同点。会议涉及面很广，发表论文50篇，其中包括学术研究论文41篇和实践经验报告9篇，按内容分为16个专题：开拓Internet、形式规格说明、可靠性、检测与评估、用户界面和规格说明、遗产系统和测试、静态分析、度量、过程、硬件及软件问题、逆向工程与程序理解、过程改进、C和C++的分析、经济与法律问题、面向对象技术、测试与分析。

大会组织了三个特邀报告，分别为：Ed Yourdon的特约报告，题目是Beyond Software Engineering：Ten Imperatives for the Successful Developer at the End of Decade；Guy Steele的特约报告，题目是Java and Evolutioin of the Web；Mark Weiser的特约报告，题目是Software Engineering that Matters to People。

大会前还组织了10多个专题讲座和专题讨论会。例如：第四届国际软件工程教学专题讨论会，第十届国际软件配置管理专题讨论会，并行和分布式系统软件工程专题讨论会和1997软件复用专题讨论会。大会期间还组织了软件产品和图书展示会。

会议发表的论文反映了国际软件工程领域的一些最新研究与实践成果，现简要介绍如下：

1. 开拓 Internet

美国哥伦比亚大学G. E. Kaiser等人在“一种基于WWW超级代码环境的体系结构”一文中指出，超级代码(Hypercode)软件工程环境涉及软件开发与演化过程中所有可能联机设置或生成的多媒体制品，包括源代码、形式化文档、数字存储资源、非形式的电子邮件以及记录等等。超级代码环境既支持这些制品件内部的(超文本)连接，也支持它们外部的(服务器)连接，将不断发现的有用连接加入系统；支持特定项目的超媒体查询与浏览；根据软件过程进行制品和超级链的自动创建；根据过程工作流的不同分别采用不同的制作工具；支持地理上分散的工作组之间的协作等等。基于WWW技术，作者提出了称为超媒体子网(subweb)的通用结构和运行在共享子网上的群组空间服务，它们可以应用到Internet上或Intranet内。

美国J. M. Perpich等人在“随时随地的代码检查：在大型软件开发中使用Web消除检查瓶颈口”一文中指出，在地理上分散的大型软件开发过程中，重要信息的传播和协作活动的同步是关键性问题。尽管这些问题并非不可解决，但由于时间、花费和效率之间的不同而使各种解决方案有不同的折衷。以往的研究表明，由于不同检查者之间存在着调度冲突，从而延迟了所需的检查结果，造成检查间隔的延长。作者在文中提出了一个采用Intranet Web的解决方案，该方案兼顾了信息传播的及时性和分布的检查者之间协作的有效性。首先，从一个试验中得出结论：检查结果的异步收集至少与同步收集同样有效。利用Web信息传播质量高、信息获取方便以及浏览器的平台无关性等优点，创建了一个耗资不大的工具并将其无缝集成到目前的开发过程中。

意大利A. Carzaniga等人在“利用可移动代码范型设计分布式应用”一文中指出，技术的革新和市场的需求使大规模分布式系统显得越发重要。然而，由于小型分布式系统的开发方法通常难以向上适应，使得大型系统的开发尚无完善的技术和方法学上的支持。近来，有人提出用可移动代码语言(MCLs)作为一种可行的解决方案。本文抽象掉语言的具体细节，利用代码的可移动性提出一种独立于任何特定技术的设计范型。同时，作者还讨论了这种设计范型的特征和应用领域，并针对一个给定的分布式应用，阐述了如何选取正确的范型。

2. 形式规格说明

日本NEC公司S. Nakajima等人在“CafeOBJ

代数规格说明的面向对象建模方法”一文中针对代数规格说明提出了一种基于脚本的面向对象建模方法。本方法的基础是一种新的代数规格说明语言CafeOBJ和多范例设计表示法GILO-2(Generic Interaction Language for Objects)的集成。CafeOBJ是从代数规格说明语言OBJ发展而来的,基于隐藏的有序分类重写逻辑(hidden order sorted rewriting logic)的面向对象形式规格说明。GILO-2提供了协同功能并可在面向对象建模中获取脚本行为的对象类。对于给定的被开发系统的问题描述,该方法通过采用GILO-2进行基于脚本的面向对象设计,为将问题分解成可运行的CafeOBJ规格说明模块提供指导,分解反应了问题域的结构。

美国密执安州立大学E. Y. WANG等人在“OMT中动态模型的形式化和集成化”一文中指出,对象建模技术(OMT)的对象模型、动态模型和功能模型三种模型的集成缺乏良好定义的语义,因此妨碍了全面的开发过程,特别是在设计阶段。以前,作者曾采用代数规格说明术语对对象模型进行了形式化。但是,代数规格说明仅能获取静态性质和一个系统的结构性质,不能显式地描述行为,而这对于系统开发是很关键的,特别是在设计阶段,为了使用严格的技术说明并验证系统行为,采用结构描述术语对动态模型进行形式化是十分必要的。对于对象和动态模型以及它们的集成,文中提出了一种具有良好定义的形式模型。该模型是采用著名的规格说明语言LOTOS术语描述的。图形表示法的形式化使得许多自动化处理和分析任务可行,例如行为模拟和规格说明的各层次间的一致性检查。

意大利L. Baresi等人在“工业实践中引人形式规格说明方法”一文中指出,尽管形式规格说明方法具有优点并且其理论和工具都比较成熟,但是它在工业项目中并没有经常得到应用。开发人员对形式符号的不熟悉及使用中的困难是限制形式规格说明方法成功应用的主要原因。基于用形式定义的普通前端符号映射形式模型的方法能够解决实际问题。但是,迄今为止所提出的映像都缺乏灵活性,使得这些方法没有应用到适合这些解决方案的环境中。文中提出了一种基于形式化的新的解决方案,它定义了从前端符号到形式模型的映射。这个框架适合不同的前端符号和形式模型,支持将由形式模型得到的分析结果映射到开发人员选择的前端符号。该方法在工业环境实验应用实例中已得到确认。文中给出了一些迄今为止所获得的工业结果和正在进行的试验。

3. 可靠性

美国纽约州布鲁克林市工业大学P. Frank等人在“选择测试方法支持可靠性”一文在一个发现程序失败(failure)并修改软件而消除失败的模型中,比较了几种测试方法。所考虑的问题是,使用测试实例来发现失败(“调试测试”)或通过模拟使用并在使用过程中发现失败(可运行的测试)是不是最好的。“最好”的含义是用在所有测试到的失败都消除以后获得的可靠性来度量的。文中的比较是对以往工作的扩展,其中度量是指发现一个失败的概率。该文在理论上的处理就是概率和分析。文中揭示了一些特别的情况,其中每一类测试都是优良的。

美国凯斯西部保留地大学A. Podgurski等人在“维护之后的软件可靠性重新评估”一文中指出,回归测试中测试实例复用与修改后软件可靠性的评估是不相容的,因为软件修改和以往测试结果之间的依赖性使得评估出现偏差。随机输入的统计测试能够有效地评估可靠性,但是其开销很大,除非考虑了软件变化的性质。作者提出了一种在某些普通的环境下评估修改后软件可靠性的比较经济的方法。该方法通过对连续软件版本性能差别的评估,更新以前的可靠性评估结果。老版本作为一种相对正确的结果来降低为保持与需求一致而进行人工检查的运行次数。该方法是合理的,它只采用了基础概率,不需要假设可靠性增长。

日本奈良科技学院K. Shima等人在“不同软件错误的失败密度研究”一文中介绍了软件可靠性增长模型中失败密度分布的试验研究情况,作者发现常用模型中关于失败密度遵从γ-分布的假设并非永真。文中新的软件可靠性模型中没有采用这种假设,而是从失败数据中计算失败密度。这种新模型比常用模型更准确地预测了发现错误的数目。

4. 检测与评估

美国马里兰大学C. B. Seaman和V. R. Basili在“代码检测中人员沟通的经验性研究”一文中介绍了作者对软件开发组成员之间沟通问题的经验性研究,特别是在一个软件开发项目中有关代码检测数据的收集。作者感兴趣的问题是组织结构(开发者之间的关系网)是否对开发者之间信息交流的开销有影响。采用的方法既有定量的,也有定性的,该项研究的贡献是为进一步的调查提供一系列很好的假设。

美国明尼苏达大学M. Stein等人在“分布式异步软件检测的实例研究”一文中指出,传统的软件检测要求参加者同时集中在同一个地点进行,而分布式的异步检测允许参加者独立于时间、地点进行会面,从而使检测更加方便。作者报告了一个工业化的研究实例,该实例使用了一个分布式的异步软件检测工具。试验表明分布式的异步软件检测是地理上

分散的工作组进行合作的、有效的方法。

5. 用户界面和规格说明

法国 J. P. Jacquot 和 D. Quesnot 在"用户界面早期规格说明：采用形式方法"一文中介绍了他们在用户的形式规格说明领域所做的工作。作者认为，工程用户界面将因使用形式方法而获益匪浅。为此，要求形式框架具有两个特性：首先，它必须基于一个能表示相关属性的形式模型；其次，必须能实现用于帮助验证规格说明的工具。我们用 VDM 形式地定义了一个基于规划表示的对话结构模型。该模型为一个规范语言(用来对用户界面进行初始形式描述)提供了语义基础；反过来，根据这一语义实现了一个原型工具，它与对话结构的交互可视化结合可产生界面的可执行实例。该原型工具可用于规格说明的验证。

美国 W. M. Wilson 等人在"需求规格说明自动分析"一文中指出，一般认为，不好的需求报告会导致软件功能上的缺陷。GSFC 软件技术中心(SATC)开发了一个早期生命周期工具，用来评测用自然语言书写的需求报告。该工具在文档中搜索被 SATC 规定为质量标识(比如，弱语句)的项。工具产生的结构用来指明需求文档中的规格说明语句和需要改进的结构化区域。工程管理人员可用这种度量方法来辨识、排除规格说明中潜在的风险。

加拿大约克大学 T. C. N. Graham 和 T. Urnes 在"支持时序媒体集成图形用户界面的体系结构"一文中指出，随着计算机越来越多地被用来作为传递信息的途径，多媒体逐渐成为交互应用的重要组成部分。有效地对多媒体信息进行通信，要求媒体的紧致集成、对多媒体的高度交互控制，以及用多媒体进行群体交互。文中介绍了如何对用于用户界面开发的 MVC 扩充以支持时序媒体，该方法允许对集成媒体的交互应用(包括多媒体群件)进行简单的规格说明。文中的所有实例都采用 Clock(一种新型编程环境)实现，运行在 IBM、SGI 和 SUN 工作站上。

6. 遗产系统和测试

美国 Mitre 公司的 A. S. Yeh 等人在"处理重获的软件体系结构示图"一文中指出，对于很多软件工程任务来说，从遗产软件中自动地恢复软件体系结构示图(逆向工程)是有价值的，尽管直接得到的示图提供的信息，要么太琐碎，要么太复杂，不合适实际应用。本文提出了使这些表示更加有用的方法。该方法自动地合并与简化了原来的示图，来生成层次结构，混合结构和抽象。

法国 G. Bernot 等人在"关于概率功能测试的一个理论"一文中为'概率功能测试'提供了一个框架。一个测试数据集合的成功对被测试系统的正确性的信任程度，可以描述成两个参数的函数，一个是对可信性的估计，另一个是当卖主告诉客户这个可信性百分比时所冒的风险。这个结果基于本文提出的"公式测试"理论。作者还提供了一个工具的原型，该工具能够根据这个理论辅助生成测试案例。

7. 静态分析

美国马萨诸塞大学 G. S. Avrunin 等人在"分析部分实现的实时系统"一文中提出了一个用来分析部分实现的实时系统的方法。作者用该方法分析了一个实时并发系统，这个系统的一些组成部分用 ADA 实现的，还有一些部分是用正则表达式和图形间隔逻辑(GIL，一个实时时序逻辑)说明的。文章介绍了如何建立部分实现的系统的模型并且考虑到诸如运行超时，进程调度等的性质，同时还考虑到相当规模程序的可跟踪分析。这个方法可以是完全自动的。

意大利 M. Pezze 和 M. Young 在"构建多形式状态空间分析工具——使用规则来说明模型的动态语义"一文中指出，人们已经对并发系统的几种表示方式开发了状态空间分析技术，但是每一种工具或技术都是针对一个单一的设计或程序表达的。文章给出了一种方法来对异构系统构建多形式状态空间分析工具，这个方法使用了一种低级模型(inframodel)，它只表达由工具构件表示的能被剪裁来反映每种形式的语义的基本信息，每种形式的(操作)语义以及在不同形式描述的构件之间的交互都是由转换的管理、匹配和点火规则分别描述的。这样就有了比纯的转换方法更加自然，紧凑的内部表示和更加高效的分析。

8. 度量

意大利 M. Pighin 和 R. Zamolo 在"基于判别式静态分析的预测度量"一文中提出一种基于判别式静态分析的方法，该方法通过评估程序的一系列结构化参数来预测它的风险度，即它包含错误的可能性。度量建立在一实验性背景环境中，该环境(大约 350000 行代码)允许将数据分成两部分，一部分用来进行模型的有效创建，另一部分作为客观的统计手段对所建议的方法学进行评估。得出的结论表明作者的基本假设成立，这一独特的分析类型可以在固定的环境中作为对危险程序早期标识的预测度量在软件测试和提交阶段使用。

美国卡内基-梅隆大学 B. Bruegge 和 A. H. Dutoit 在"软件开发的通信度量"一文中得出一经验性结论：对由群件工具生成的人工通信的度量有助于更好地理解其开发过程。文章介绍了一个用来开发和测试通信度量的测试台。这样一个测试环境是经验性软件工程的理想环境，它非常现实地提供了对

重要项目参数的控制观察。文中通过三个概念证明实验阐述了通信度量的价值，并为通信度量的确认提出一种基于结构公式的统计框架。

9. 过程

日本大板大学 Shinji Kusumoto 等人在“基于一般化随机 Petri 网的新软件项目模拟”一文中提出一种可反映质量、花费和交付数据的软件项目及评估模型。该模型分为项目模型和过程模型两部分。前者瞄准活动、产品和开发者三种关键成分；后者包括一组活动模型，分别说明设计、编码、复审、测试、调试等活动，这种方法通过引入“工作负担”的概念而把人的因素之影响加到模型中，作者开发了一个模拟器，以支持目标过程的描述，执行由活动模型描述的过程并分析模拟结果。他们还把这个模拟器应用于一些实际的软件项目，并把评估值与实现数据进行了比较。结果表明，这种模拟器将可以应用于实际软件项目的管理。

10. 逆向工程和程序理解

美国 Mellon 大学的 R. O'Callahan 和 D. Jackson 在“Lackwrt：一个基于类型推理的程序理解工具”一文中提出，通过静态地决定线程结构和何处需要变量集合分享公共表示，可以识别抽象数据类型，找出对抽象原则的违犯、发现数据结构中未使用的变量、函数和域，发现抽象数据类型中简单的操作错误，并定位可能引用一个统一位置。通过计算与分享类型推理的表示，可以用类型对这些表示进行译码，此方法是有效的和完全自动的。

德国 C. Lindig 和 G. Sneltong 在“基于数学概念分析评估遗留代码的模块结构”一文中介绍，通过分析过程和全局变量之间的关系，可构造一个所谓的“概念格”(concept lattice)。文章讨论了如何在概念格上展示出模块结构以及如何评价候选模块之间的聚合和内聚。作者已将该方法运用于 100K 以上的 Modula-2、Fortran 和 Cobol 的程序分析。

美国亚特兰大佐治亚技术学院的 D. J. Jerding 等人在“预见程序执行中的交互”一文中指出，实现、修改和再工程一个 OO 系统，需要对程序执行时发生的交互进行理解，为此要寻找识别、预见和分析 OO 程序执行时交互的方法以考察和理解动态行为，作者发现程序执行时的交互脚本可用以对理解过程进行抽象。文中提出了一种识别这些交互模式的方法。该方法可用于支持设计恢复、验证、再工程，既可用于 OO 程序也可用于过程式程序。

11. 过程实现

美国 NASA T. Hammer 等人在题为“度量需求测试”的实践报告中指出，软件系统常常是逐步发布的，每次发布增加一些新功能。为此该报告介绍了一种新的软件需求工具，使质量保证工程可开发和使用一些新的规则来帮助评价这些需求与测试关系，以保证新系统中所需的功能。

12. 对象技术

美国南加州大学 D. S. Wile 在“从具体语法得到抽象语法”一文中指出，现代的软件工程实践促进了领域专用的规格说明语言的发展，以刻画特殊问题领域的语言习惯和行话。对于那些留下深刻理解的领域，在一种具体的语法开发之前，最好的办法是构造一种抽象的语法来刻画其领域概念和其中事物的抽象。该文介绍了从低级的具体语法的规格说明产生好的抽象表示的转换过程，并指出这种抽象表示也正是面向对象的开发模型所追求的目标。

加拿大阿伯塔大学 G. Froehhich 等人发表了题为“在面向对象的应用构架中引人挂钩”的论文，指出面向对象的应用构架对给定的问题域提供了一般的设计和对此设计的可复用的实现。它有很大的复用潜力，但这种潜力只有该构架能被开发人员很好地理解和有效地运用时才能得以发挥。构架在设计和实现上的复杂性可能使之难于理解，因此，需要对构架的使用者提供良好的文档和指南。针对目前几种方法未能对构架的用途和如何使用建立良好的文档，本文作者引人了“挂钩”(hook)的概念作为对构架建档和提供使用指南的手段。每个挂钩是对构架上下文中一个需求或一个问题的简单明了的理解，较为复杂的需求可用一组挂钩来解决。文章详细介绍了挂钩的概念、描述形式及使用方法，并指出了它与设计模式和范例的不同。

在会议期间还表彰了一些杰出人员，Peter G. Neumann 获得 ACM SIGSOFT 奖。著名 Boehm 教授获得第一个软件工程杰出研究成就 ACM SIGSOFT 奖，表彰他在软件经济学、风险管理和过程等方面的突出研究成就。

十年前，第九届国际软件工程会议最有影响的论文奖授予著名美国马萨诸塞大学 L. J. Osterweil 教授的论文，名为“软件过程也是软件”。另一有影响的论文奖授予著名英国帝国技术学院 M. M. Leham 教授的论文，名为“过程建模”。

1996 年度计算机企业家奖授予 D. S. Bricklin 先生，他首先设计实现电子表格、字处理系统等等，他的产品促进了计算机工业的发展。