

1

简介:桌面隐喻和 技术的新用途

Victor Kapteinin 和 Mary Czerwinski

本书旨在介绍和探讨设计下一代数字化工作环境的新方法。目前最普及的计算机系统,例如 Microsoft Windows 和 Mac OS,都是基于桌面隐喻的。对于许多用户和设计师来说,这些是他们所了解的唯一数字化工作环境。人们很容易假设桌面隐喻将永远决定我们对计算机系统的体验。本书挑战了这一假设。

本书的出发点在于,我们熟知的桌面系统很可能只是交互环境发展的一个暂时阶段,即便它取得了巨大的成功。未来的系统可能会进一步发展、修改,甚至放弃这一比喻。本书试图系统地探讨一系列与未来交互环境设计相关的问题,并特别关注可在“后桌面”系统中采用的全新设计解决方案、概念和方法。

基于桌面隐喻的系统在 20 世纪 80 年代初大规模涌现,成为首个“面向所有人”的通用工作环境 (Smith 等人,1982)。设计者的意图是支持独立计算机的个人用户(通常在传统的办公环境中)启动应用程序以及存储和检索文档。桌面系统为这些类型的活动提供了一致、实用的环境,并被证明是一种巨大的……

巨大的成功。

然而,如今普通计算机用户的生活已大不相同。为了完成日常任务,人们经常使用各种技术,例如台式电脑、笔记本电脑、PDA(个人数字助理)和智能手机,并使用各种类型的信息对象,例如

2 Victor Kaptelinin 和 Mary Czerwinski

诸如文件、电子邮件、URL 和联系人等信息，以便彼此协作和沟通。信息工作需要整合来自各种来源、跨越多个应用程序和同事的信息。实证研究和逻辑分析表明，传统的桌面系统无法为信息工作者在以协作、多任务、多角色和多样化技术为特征的现实环境中提供足够的支持（例如，Plaisant 和 Shneiderman 1995;Kaptelinin 1996;Dourish 等人 1999;Czerwinski、Horvitz 和 Wilhite 2004）。

桌面隐喻的麻烦

现有桌面系统最初成功的一个关键因素是一套直观清晰的底层原则，这些原则为整个数字化工作空间构建了一致的思维模型。桌面提供了一个空间，用于在重叠的窗口中显示当前活动文档的内容，而分层文件系统则方便访问存储的文档和工具。用户可以凭借对典型办公环境的了解，对如何使用桌面系统中的各个对象和功能来执行手头的任务做出合理的猜测。

基于桌面隐喻的工作区并非单一的。用户可以通过创建新文件夹并重新排列现有文件夹来创建任意数量的子空间。这些子空间使得文件组（以及必要时的子文件夹）彼此独立成为可能。尽管这种组织方式较为僵化，无法反映文档重要性随时间的变化（与成堆的纸张不同；参见 Malone 1983）以及用户认知过程的动态变化（Lansdale 1988），但它能够在整个文件系统中进行简单、直观的导航。因此，桌面系统为个人计算机用户提供了相对集成的文件处理环境。

然而，即使在这一应用范围内，桌面隐喻也明显存在固有的局限性。桌面系统设计人员一直以来面临的难题之一是如何将信息访问与信息显示结合起来 即同时支持 (a) 信息对象的访问和 (b) 信息对象内容的可视化呈现。

与其源头 物理办公室 相反，“桌面”的隐喻基于使用同一表面 屏幕 来显示和访问信息。物理桌面上可能堆满了个人文档和成堆的纸张,但我们无需清理这些桌面就能找到文件柜、抽屉或书架。人们通常无需在桌面和文件柜之间做出选择;他们可以同时看到并独立使用它们。而现代信息技术的用户则必须使用同一屏幕空间来查找信息对象并查看其内容。

在磁盘上定位文档和编辑文档都使用计算机屏幕的同一物理表面。

现代桌面系统与早期桌面系统最显著的区别或许在于,多年来设计师们精心设计了一系列工具,帮助用户将信息访问和信息显示结合起来。早期系统的用户必须通过移动、调整大小或关闭打开的窗口来清理屏幕区域,才能查看和访问桌面上的对象。而现代系统,例如 Microsoft Windows XP 或 Mac OS X,允许用户无需选择桌面上的对象即可访问计算机内容。例如,用户可以随时从“开始”菜单 (Microsoft Windows XP)或 Finder 窗口 (Mac OS X) 打开“文档”文件夹。

个人电脑桌面隐喻的一项重大进步是 Microsoft Windows 中任务栏的引入（有关 X Windows 类似工具的讨论,请参阅 LaStrange 1989）。打开的窗口不仅会遮挡桌面上的对象,还会相互隐藏 就像放在物理桌面上的纸张会遮挡其他纸张一样。与物理桌面不同,虚拟桌面允许用户查看所有打开文档的列表,并通过从列表中选择其中任何一个文档来使其可见。如果在现实世界中也能实现这一点,那么这项功能肯定会受到许多使用物理桌面的用户的青睐！

尽管过去几十年桌面系统的设计取得了显著进步,但如何将信息访问和信息显示结合起来仍然是一个难题。可以说,阻碍更有效解决方案的一个因素是桌面隐喻的固有局限性。“桌面”这个名称本身就意味着一个单一的、有限的物理表面,用作虚拟环境中所有资源的窗口。然而,当前的技术

4 Victor Kaptelinin 和 Mary Czerwinski

技术发展提供了更加广泛的可能性。多屏显示、大屏幕显示、环境显示以及新的输入和传感技术,为信息访问和信息显示的结合开辟了新的可能性。本书将介绍其中一些新的可能性。

此外,桌面隐喻本身似乎有些不一致。桌面系统可以从两个不同的视角来看待,这两个视角分别对应于:(a) 虚拟环境的逻辑结构;以及 (b) 用户如何感知该环境。

从逻辑角度来看,虚拟空间的顶层,即访问其他部分的入口点,是用户可用的存储设备集合(例如,Microsoft Windows 中“我的电脑”文件夹中显示的内容)。从这个角度来看,桌面只是这些设备中的一个文件夹。然而,从用户的主观视角来看,系统的入口点是桌面。通常,环境的其他组件,包括“我的电脑”文件夹,都可以从桌面访问。这种不一致可能会让一些用户感到困惑(参见 Ravasio 和 Tscherter,本书)。

桌面系统的另一个常见问题是多任务处理。一方面,桌面系统支持管理多个任务上下文的一些关键活动。如前所述,桌面系统用户可以方便地将文件组织成与其任务结构相匹配的层级结构。此外,Apple Macintosh System 7 和 Microsoft Windows 95 使个人电脑用户能够享受以前只有工作站用户才能享受的功能 同时使用多个应用程序。然而,这些在支持多任务处理方面的成功突显了桌面系统在选择性显示与任务相关的信息对象集方面的固有局限性。这可以通过以下示例来说明。假设与两个不同任务(任务 A 和任务 B)相关的文件被正确地组织到两个不同的文件夹中,并且用户可以方便地访问这些文件夹。即使在这个简单的例子中,从一个任务切换到另一个任务也可能存在问题。用户应该 (a) 关闭所有与任务 A 相关的文档,并打开与任务 B 相关的文档,这可能会很繁琐,尤其是当用户必须在任务之间频繁切换时;或者 (b) 保持所有文档打开,这可能会造成持续的干扰。如果

用户需要处理许多打开的文档来完成这些任务,因此这两种策略都不能被认为是最佳的。

解决选择性呈现任务相关信息问题的一个显而易见的解决方案是采用多个与任务相关的桌面,就像 Rooms 系统 (Henderson 和 Card,1986) 的开发者所提出的方案一样。然而,实现多桌面理念的唯一成功案例是基于 UNIX 系统的窗口管理器,它支持在预定数量的桌面之间切换。

至于更为普及的个人电脑操作系统,例如 MS Windows 和 Mac OS,多 (虚拟)桌面尚未受到用户的广泛欢迎。

因此,即使在狭义的应用范围内 个人用户在独立计算机上处理文件 桌面隐喻也表现出明显的局限性。电子邮件和互联网作为沟通和信息共享工具的使用进一步加剧了桌面隐喻的问题。这些发展导致单一虚拟环境中出现了多个信息层级结构 (Dourish 等人,1999) 。此外,桌面系统的功能并不同样支持这些层级结构。结果,文件成为了“一等公民”,而单个电子邮件消息或即时消息则并非如此。例如,选择一组不同的文件进行复制、备份、传输到其他设备或移动相对容易,但这在电子邮件中并不一定那么容易处理。以电子邮件附件为例 它们必须先被明确保存或放置在文件夹中,然后才能永久保存或以组的方式进行轻松操作。

此外,万维网和搜索引擎的大量使用使得用户可以轻松地将URL或网站内容作为电子邮件内容发送,但访问和永久存储这些内容并不像文件那样容易。大多数情况下,用户必须在长长的电子邮件列表中寻找包含附件或网站信息的邮件。新的个人文件和网页搜索引擎使得在搜索过程中利用电子邮件、文件和网站内容的元数据变得更加容易,但这些系统可能并不适合取代计算机桌面。

最后,随着劳动力流动性的增强,用户能够随时访问自己的信息变得越来越重要。

6 Victor Kapteinin 和 Mary Czerwinski

在任何设备上都能同步。桌面隐喻在这里再次失效,因为每台设备都有自己的外观和感觉,但必须以某种方式与备用环境同步。

总而言之,过去二十年桌面系统的发展已经暴露出桌面隐喻的局限性。具体而言,桌面隐喻未能充分支持信息对象的访问及其内容的显示、多任务处理、多信息层次结构处理、通信与协作以及多种技术的协同使用。

迈向集成数字化工作环境

人机交互 (HCI) 研究人员很早就认识到基于桌面隐喻的系统的局限性,大约在系统广泛应用之时。这一认识引发了人机交互界的争论,并促进了新方法的发展。如今,二十年过去了,我们可以清晰地看到,源于早期人机交互研究并不断发展的一系列研究和开发。20世纪 80 年代,Malone (1983)、Henderson 和 Card (1986) 以及 Lansdale (1988) 等人做出了具有影响力的研究。

Rooms 系统 (Henderson 和 Card,1986)可能是探索传统桌面系统替代方案的第一个重要里程碑。接下来十年发展中的一些亮点包括:Pad++ 系统 (Bederson 和 Hollan, 1994)、个人角色管理 (Plaisant 和 Shneiderman,1995)、Norman 的基于活动的计算 (Norman,1998)、《ACM 通讯》上的“反 Mac”界面辩论 (Gentner 和 Nielsen, 1996)、Lifestreams 系统 (Fertig,Freeman 和 Gelertner,1996) ,以及《SIGCHI 通讯》上关于用户查找和归档计算机文件实践的讨论 (Barreau 和 Nardi,1995;Fertig、Freeman 和 Gelertner,1996) 。

世纪之交,挑战桌面隐喻的新方法、原型和工作系统涓涓细流,最终汇聚成一股持续增长的稳定之流。普适计算作为一个独特领域的出现,进一步激发了人们对交互式环境新设计方案的探索 (例如,Arnstein 等人,2002;Voida 等人,2002;Judd 和 Steenkiste,2003) 。新的组织原则被提出,作为单一工作界面与嵌入式容器的层级化、空间逻辑结构相结合的替代方案,而传统桌面设计正是基于此。

系统。新方法基于不同的基础概念,包括(但不限于):时间(Fertig、Freeman 和 Gelernter,1996 年)、逻辑属性(Dourish 等人,1999 年)、人员(Nardi 等人,2001 年)、任务和项目(Robertson 等人,2000 年;Voida 等人,2002 年;Arnstein 等人,2002 年;Kaptelinin,2003 年)以及集体活动(Christensen 和 Bardram,2002 年)。

目前,这些方法之间的联系相当松散。尽管它们之间的交叉引用越来越多,但每种方法都主要将自身定位于桌面隐喻,而非桌面的其他替代方案。然而,最近的发展表明,加强各个研究工作之间的协调既有理论原因,也有实践原因。

每一个挑战或扩展桌面隐喻的创新系统和方法,都有助于探索一组更普遍的研究问题。在本书中,我们暂时将这些问题定义为“集成数字工作环境的设计”。

我们所说的“集成”环境是指基于一套连贯的底层原则,支持在各种任务和情境中协调使用工具和资源的环境。“数字化”和“工作”这两个词在这里是广义的。“数字化”主要指重塑虚拟世界,但本书中也包含对数字环境和物理环境的考量。“工作”涵盖任何更高层次的活动,这些活动是定义信息工作者主要角色或职能的一部分,但也包括学习、休闲等等。

从实践角度来看,新方法对日常技术使用的影响有限,这可能意味着没有任何单一的概念或系统可以被视为“灵丹妙药”。开发激励性和支持性的集成数字工作环境需要来自不同学科和视角的协调努力。因此,识别当前集成数字工作环境研究之间的共性和相互关系,不仅对这些环境的设计具有概念意义,也具有实践意义。

本书的主题和结构

本书旨在将集成数字工作环境的分析和设计视为人机交互研究的一个独特领域,并支持这一新兴领域的整合。本书提供了

8 Victor Kaptelinin 和 Mary Czerwinski

全面概述最相关研究的最新成果，并尝试促进这项研究的协调。

本书的各个章节在多个方面相互关联。所有贡献中最明显的共同点或许在于，它们都探讨了桌面隐喻的问题。上文题为“桌面隐喻的麻烦”的部分最后列出了传统桌面系统的局限性。列表中的每一项都可以映射到至少一章中的研究问题或设计解决方案：

信息访问与信息显示本书介绍了两个系统：Scalable Fabric (Robertson 等人) 和 Kimura (Voida 等人)，它们将一个显示任务相关信息对象的工作空间与一个方便访问其他对象和其他任务的外围设备相结合。这两种方法的区别在于，外围设备是显示在同一屏幕上的缩小图像 (Scalable Fabric)，还是显示在其他表面上的动态生成的蒙太奇图像 (Kimura)。

多任务处理本书中的一些章节 (Robertson 等人、Plaisant 等人、Voida 等人、Bardram 以及 Kaptelinin 和 Board-man 等人的著作) 都着重强调了信息工作者的任务、角色、项目或活动。由于多任务处理在我们的日常生活中变得如此普遍，并且所有这些作者都观察到桌面隐喻中对多任务处理的支持非常有限，因此这些章节中描述的系统旨在更好地帮助用户找到与特定任务相关的信息，快速启动任务，并在中断后重新获取任务。

多重信息层级结构：支持不同信息层级结构的集成是大多数章节的核心议题。本书提出了四种通用的设计方案：在不同类型的信息对象中使用相同的逻辑属性 (Freeman 和 Gelernter, Karger)；将信息对象与角色、联系人或项目关联 (Plaisant 等人, Robertson 等人, Fisher 和 Nardi, Voida 等人, Kaptelinin 和 Boardman)；将窗口组织成组 (Robertson 等人)；以及在现有层级结构中维护统一的结构 (Kaptelinin 和 Boardman)。

沟通与协作本书描述了几个考试-

如何设计个人工作环境以促进协作的例子。提出的解决方案包括围绕以下方面组织数字资源：(a) 用户角色（Plaisant 等人）,(b) 用户社交互动结构（Fisher 和 Nardi）,以及 (c) 用户参与的分布式活动（Bardram）。在最后一种情况下,工作环境是个性化的,因为它们支持个人整体地完成其应承担的任务。然而,它们也可以被视为集体活动的中介。Bardram 提出的系统的目标之一是使不同的用户能够在必要时参与并开展集体活动。

多种技术的协同使用本书的几章描述了支持在不同计算设备上协调工作的系统（Voida 等人，Bardram）。更多贡献者提到,此类支持是其方法未来发展的方向。

因此,本书的一个共同特点是处理现有桌面系统的局限性。本书的另一个共同特点是

它的设计方向。

本书汇集了众多颇具影响力的方法和系统,并由其开发者亲自撰写了十几篇描述:ABC、ContactMap、GroupBar、Haystack、Kimura、Lifestreams、Personal Role Management、Scalable Fabric、Soylent、Task Gallery、UMEA 和 WorkSpaceMirror。此外,本书还对其他一些系统进行了深入探讨,尽管这些系统的开发者并未亲自提出。其中一章（Ravasio 和 Tscherter 的著作)并未介绍其作者开发的全新系统,但其分析却基于现有桌面系统的设计。

本书的设计导向并不意味着以牺牲概念分析为代价而专注于具体的技术成果。秉承人机交互研究的精神,本书将“旨在理解的活动与旨在设计的活动”融为一体（Carroll and Rosson 1992）。

本书以三种不同但又相互关联的方式,将技术和概念探索结合在一起。首先,本书运用新颖的系统及其分析来阐明一种设计解决方案,即一种可以在一系列系统中实施的策略。其次,本书通过具体的设计,证明了一种新的

需要概念来正确理解和支持技术用户。

特别是,本书有几章论证了将“活动”作为设计的基础概念。第三,也是最后一点,系统与设计为关于桌面隐喻及其在数字工作环境中的整合的命运的总体思考提供了素材。因此,本书分为三个部分。

第一部分和第二部分共五章,阐述了设计数字化工作环境的创新方法。这些章节提出了一系列组织原则,工作环境可以依据这些原则进行组织:按时间(Freeman 和 Gelernter 著),按关系和属性(Karger 著),按任务(Robertson 等人著),按人员(Fisher 和 Nardi 著),以及按角色(Plaisant 等人著)。

第一部分“打破常规的设计”探讨了工作环境如何方便访问传统桌面系统中存储在不透明容器(即文件夹)中的信息。文中提出的解决方案以各种系统为例进行了说明:Lifestreams(Freeman 和 Gelern-ter 开发)、Haystack(Karger 开发)、Task Gallery、GroupBar 和 Scalable Fabric(Robertson 等人开发)。

Plaisant 等人以及 Fisher 和 Nardi 在第二部分“个人环境的社会维度”中提出的设计方法的基本思想是,个人工作环境的组织应该反映工作的社会背景并支持个人参与协作活动。

第三部分“从任务到活动”的两章探讨了活动的概念,并将其作为设计集成数字环境的概念工具。文中以具体系统的例子进行了说明:Kimura(Voida等人)和基于活动的计算(Bardram)。

第四部分“对桌面隐喻和集成的思考”包括两章,讨论一般性问题:分析和讨论用户如何理解桌面隐喻(Ravasio 和 Tscherter),以及比较以应用程序为中心的集成和以工作空间为中心的集成(Kaptelinin 和 Boardman)。

书中提出的主要问题和未来研究的方向是
Moran 和 Zhai 在结论章节“超越七个维度的桌面隐喻”中对此进行了讨论。

这本书是首批系统概述基于设计的集成数字工作环境人机交互研究的书籍之一。但它

不仅仅是现有系统的选集。其类型可以更恰当地定义为对新一代数字工作环境设计空间的集体探索。

本书各章节汇报了作者的最新研究成果,包括技术进步、实证评估以及其方法的创新应用。作者详细阐述了其系统背后的原理、方法的优势与局限性、用户对系统的体验、系统如何解决现有数字环境中的问题、与其他新方法的比较,以及如何利用其背后的理念帮助信息工作者设计未来的数字化工作环境。

总体而言,本书让我们得以一窥在不久的将来,信息技术如何日常应用于支持信息工作者。我们希望这些系统能够启发下一代集成数字化工作环境,为该领域的用户需求提供真正的解决方案。

参考

- Arnstein, L., Hung C.-Y., Franzia, R., Zhou Q.-H., Borriello, G., Consolvo, S., 和 Su, J. (2002). Labscape:细胞生物学实验室的智能环境。
普适计算1 (3): 13-21。
- Barreau, D. 和 Nardi, B. (1995).查找与提醒:桌面文件整理。ACM SIGCHI 通讯27: 39–43。
- Bederson, B. 和 Hollan, J. (1994).Pad++:用于探索替代界面物理的缩放图形界面。载于第七届 ACM 用户界面软件与技术研讨会 (UIST 94) 论文集,第 17-26 页。加利福尼亚州,11 月 2 日至 4 日。
- Carroll, JM 和 Rosson, MB (1992).绕过任务-工件循环:如何提出主张并按场景设计。ACM 信息系统学报10(2): 181–212。
- Christensen, H. 和 Bardram, J. (2002).支持人类活动 探索以活动为中心的计算。载于 Borriello, G. 和 Holmquist, LE (编),第四届国际会议论文集。UbiComp 2002,第 107-116 页。
- 计算机科学 2498 讲义。柏林:Springer。
- Czerwinski, M., Horvitz, E. 和 Wilhite, S. (2004).任务切换与中断的日记研究。2004 年 ACM 计算机系统人为因素会议 (CHI 04) 论文集,第 175-182 页。奥地利维也纳,4 月 24 日至 19 日。
- Dourish, P., Edwards, W., LaMarca, A. 和 Salisbury, M. (1999).Presto:一种用于流畅交互文档空间的实验架构。ACM 计算机人机交互学报,第 6 卷:133–161 页。

Fertig, S., Freeman, E. 和 Gelernter, D. (1996)。重新考虑“发现和提醒”。ACM SIGCHI 公告28:66-69。

Gentner, D. 和 Nielsen, J. (1996)。反 Mac 界面。ACM 通讯39:70-82。

Henderson, A. 和 Card, S. (1986)。Rooms: 使用虚拟工作空间减少基于窗口的图形用户界面中的空间争用。ACM 图形学报5: 211-243。

Judd, G. 和 Steenkiste, P. (2003)。为普适计算应用提供上下文信息。载于 IEEE 普适计算国际会议 (PERCOM) 论文集, 第 133-142 页。德克萨斯州达拉斯, 3 月 23-25 日。

Kaptelinin, V. (1996)。创建基于计算机的工作环境:一项针对 Macintosh 用户的实证研究。载于 1996 年 ACM SIGCPR/SIGMIS 会议论文集, 第 360-366 页。科罗拉多州丹佛市, 4 月 11 日至 13 日。

Kaptelinin, V. (2003)。UMEA: 将交互历史转化为项目情境。载于 2003 年 ACM 计算机系统人为因素会议 (CHI '03) 论文集, 第 353-360 页。佛罗里达州劳德代尔堡, 4 月 5 日至 10 日。

Lansdale, M. (1988)。个人信息管理心理学。

应用人体工程学19:55-66。

LaStrange, T. (1989)。twm (Tom's Window Manager) 概述。<http://www.lastrange.com/work/twm.pdf>。

Malone, T. (1983)。人们如何整理办公桌? 对办公信息系统设计的影响。ACM 办公信息系统汇刊1: 99-112。

Nardi, B., Whittaker, S., Isaacs, E., Creech, M., Johnson, J. 和 Hainsworth, J. (2002)。通过 ContactMap 整合沟通与信息。ACM 通讯45:89-95。

Norman, D. (1998)。《隐形计算机: 为何好产品也会失败, 个人计算机如此复杂, 信息设备才是解决方案》。

马萨诸塞州剑桥: 麻省理工学院出版社。

Plaisant, C. 和 Shneiderman, B. (1995)。组织概览与角色管理: 未来桌面环境的启示。载于第四届 IEEE 使能技术研讨论文集: 协作企业基础设施 (WET-ICE '95), 第 14-22 页。西弗吉尼亚州伯克利斯普林斯, 4 月 20-22 日。

Robertson, G., van Dantzich, M., Robbins, D., Czerwinski, M., Hinckly, K., Risden, K., Thiel, D. 和 Gorokhovsky, V. (2000)。任务库: 一个 3D 窗口管理器。载于 2000 年 ACM 计算机系统人为因素会议 (CHI 2000) 论文集, 第 494-501 页。荷兰海牙, 4 月 1 日至 6 日。

Smith, D., Irby, C., Kimball, R., Verplank, W. 和 Harslem, E. (1982)。《设计 Star 用户界面》。Byte 7:242-282。

Voida, S., Mynatt, E., MacIntyre, B. 和 Corso G. (2002)。整合虚拟与物理环境以支持知识工作者。IEEE 普适计算 1:73-79。

X 窗口管理器: 基础知识 (nd)。<http://xwinman.org/basics.php/>

我

开箱即用的设计

第一部分简介

在实体办公室中,文件和工具存放在不透明的容器中并不罕见,例如抽屉、文件柜、活页夹或纸箱。人们通常先找到合适的容器,然后检查里面的东西,然后开始工作。

可以说,在不透明的容器中翻找东西是一种低效且令人沮丧的工作环境管理方式(例如,参见 Freeman 和 Gelernter 的著作)。在桌面系统中,没有书架、墙壁或大桌子;唯一可直接看到的表面就是桌面,因此人们不得不在不透明的“盒子”中做更多搜索。

第一部分中的章节提出了一些方法,以克服传统桌面系统的这一问题,并促进对潜在有用信息的访问(见表I.1)。这些方法包括创建按时间顺序组织的信息对象子集(Freeman和Gelernter),从而能够开发灵活、动态的个人信息环境。

表 I.1
第一部分中提出的设计方法概述

章	作者	系统	组织原则
第二章	弗里曼和 熟练	生命流	时间和搜索
第三章	卡格尔	草垛	关系和属性
第四章	罗伯逊·史密斯 迈耶斯、鲍迪施 切尔温斯基,霍维茨, 罗宾斯和谭	任务库 分组栏 可扩展结构	任务(明确) 任务(隐式) “焦点加背景”

16 第一部分简介

通过定义和使用关系和属性（Karger）,围绕明确或隐含定义的任务组织信息对象,并将潜在有用的对象放置在工作区域的视觉边缘（Robertson 等人）。

第一部分以 Freeman 和 Gelernter 撰写的关于 Lifestreams 项目的章节开篇,清晰地阐述了透明信息存储的需求。本章是对 Lifestreams 系统创建者的个人及历史的记述。该系统的概念 沿着时间轴组织用户的信息并强调搜索 早于许多如今看来具有开创性的搜索引擎理念。Freeman 和 Gelernter 对该系统在网络搜索出现之前的设计思路的探讨,不仅为 Lifestreams 系统,也为之后的许多系统提供了一个引人入胜的历史框架。

Karger 在一章中介绍了 Haystack 系统,其目标是赋予用户最大程度的控制权,让他们能够创建自己的个人信息环境。该系统允许用户选择各种感兴趣的信息对象,记录其属性,并根据自身需求组织这些对象。该系统被设想为一个强大的工具,支持开发一个高度个性化的信息空间,用户可以在其中保存、描述、构建、查看或搜索自己的所有信息。

Robertson,Smith,Meyers,Baudisch,Czerwinski,Horvitz,Robbins 和 Tan 撰写的章节介绍了三个系统 GroupBar、Task Gallery 和 Scalable Fabric 它们可以优雅地补充现有桌面系统的用户界面。他们的设计通过一些表示形式来实现这一点,这些表示形式可以帮助用户围绕更高级别的任务组织资源集合,以便能够在这些集合之间便捷地切换。

这些系统中的每一个都探索了不同的用户支持方式,扩展常规 Microsoft Windows 任务栏的功能,将窗口组织成组,这些组是隐式定义的任务（Group-Bar）;提供准 3D 表示,方便访问一组 2D 工作区（任务库）;并采用“焦点加上下文”可视化（Scalable Fabric）。¹

第一部分概述了数字化工作环境设计中可以采取的各种途径。所提出方法的多样性也对未来的研究和开发提出了挑战。这些提议中的任何一种是否能够

解决方案是否可以作为设计的唯一基础?如果不是,不同的方法如何相互结合,又如何与桌面隐喻相结合?

可以说,有些方法可以有效地互补。Lifestreams 的一些优势 (例如提供历史背景) 和 Haystack (例如赋予用户定义、记录和管理属性的控制权) 似乎可以整合到一个系统中。任务库、GroupBar 和 Scalable Fabric 可以轻松集成到现有的桌面系统中。然而,还需要开展更多工作来确定这些方法的整合如何 (以及是否) 能够促进数字化工作环境设计的进一步发展。

笔记

1. 使用外围可视化系统的另一个示例是第 7 章中描述的 Kimura。

2

超越生活流:桌面隐喻的必然消亡

埃里克·弗里曼和大卫·盖伦特

1994年,我们采取了一种我们认为激进的方法来改变我们的电子生活,即创建一种桌面隐喻的替代方案。

我们的方法 Lifestreams (Gelernter 1994;Freeman 1997)是一种用于管理个人电子信息的软件架构。这种方法的革新之处在于,Lifestreams 摒弃了文件名、文件夹和静态文件,代之以一个简单的数据结构:一个按时间顺序排列的文档流,并结合了一些用于定位、组织、汇总和监控信息的强大操作符。我们在耶鲁大学的原型实现实现了该系统的许多定义特性,并使我们能够对该模型的关键思想进行实验。

后来,商业版本实现了 Lifestreams 的一组较窄的功能,但却满足了现实世界的需求,并实现了有限但成功的生产用途,这种用途在某些站点仍在继续 - 尽管我们的商业努力已经走到了尽头。

十多年后,只需回顾苹果的“iApps”(以及其他项目和产品),就能看到 Lifestreams 的许多功能实际发挥作用。我们并不声称我们的工作直接或间接地影响了那些在 Lifestreams 框架下引入、描述和推广其功能的供应商。我们只是声称 Lifestreams 系统非常准确地预测了未来的发展。我们一直声称 Lifestreams 的定义功能是自然而然的,最终会成为标准的商业信息管理系统。如今,这种情况正在发生;因此,人们对早期的工作重新燃起了兴趣。在本章中,我们将回顾我们对替代桌面隐喻的研究。我们将描述 Lifestreams 项目:我们最初的动机,以及这些动机的基础。

在工业心理学和人机交互领域,研究系统,20世纪90年代中期对该系统的反应,以及在当今语境下的生命流。最后,我们将指出在当今的应用程序和操作系统中类似生命流的功能。

动机:回顾1994年

我们俩都缺乏整理东西的天赋（或者说耐心）。1994年,我们实体桌面的混乱程度,只有电脑桌面和文件系统能与之媲美。

我们都意识到,除非雇佣一个完整的秘书团队,否则我们的实体办公空间将毫无希望。但我们知道,必须有更好的方法来管理我们的软件桌面。虽然我们推崇桌面隐喻（并且都是Macintosh的狂热用户）,但我们确信桌面无法扩展以应对即将到来的网络世界信息洪流；事实上,它已经让我们俩失望了。文件（20世纪50年代的发明）、分层存储（20世纪60年代的发明）和桌面隐喻（20世纪70年代的发明）都是杰出的发明,但它们都是在个人电脑普及、电子邮件普及以及万维网诞生之前诞生的。桌面隐喻试图简化常见的文件操作,将其呈现为纸质世界中人们熟悉的语言（纸质文档变成文件,文件夹变成目录,通过垃圾桶图标进行删除）。这种隐喻具有重要的优势,尤其是对新用户而言（尽管仍然需要向新用户解释电子桌面与真实桌面的相似之处,为什么以及如何命名每张“纸”,如何弹出CD等等）。但桌面隐喻也限制了我们未来的软件设计选择。

我们愿意承认我们不是典型的计算机用户。但我们知道,一些,实际上很多,甚至可能是大多数电脑用户,都和我们一样感到沮丧。为了支持（或推翻）我们的猜想,我们求助于人机交互(HCI)和人为因素社区,在那里我们遇到了马克·兰斯代尔(Mark Lansdale)、托马斯·马龙(Thomas Malone)等人的研究成果。我们在我们认为桌面存在问题的领域寻找证据,尤其是在命名、归档和查找、存档、提醒和汇总方面。

归档和查找

Lansdale (1998)的研究旨在研究回忆、识别和分类的过程，并试图提出基于心理学理论的软件框架。他的研究建立在Malone关于人们使用信息方式的开创性研究的基础之上：《人们如何整理办公桌？对在线信息系统设计的启示》。

(1983)。马龙在这项研究中的目标是“系统地了解人们实际上是如何使用办公桌的，以及他们如何组织个人信息环境”，以期改善电子

系统。

两位研究人员都对分类进行了研究，兰斯代尔将其描述为“问题在于决定使用哪种分类，以及事后记住分配给每个分类的确切标签”。

这个主题对我们尤为重要，因为它与创建目录和归档文档直接相关。而这正是桌面隐喻的支柱。马龙的研究表明，信息分类可能是人们遇到的最困难的信息管理任务。兰斯代尔发现，“很简单，信息无法归入系统上可以通过使用简单标签实现的整齐分类。”杜迈斯和兰道尔 (1983)的研究指出了造成这种情况的两个具体原因：(1) 信息存在重叠和模糊的类别；(2) 用户不可能生成随时间推移而保持明确的类别。兰斯代尔进一步基于实证证据得出结论，人们“不擅长对信息进行分类”，强迫用户这样做是一种“有缺陷的心理过程”。

信息分类的困难，以及为之付出却得不到任何回报，通常会导致用户（马龙发现）根本不归档信息，以此来克服“在众多弊端之间做出决定的困难，并避免做出决定的后果”。这一结论反过来表明，两项用户任务（归档和查找）之所以困难且繁琐，是有原因的：大多数人并不擅长这些活动。

还要注意的是，用户被迫以另一种微妙的方式对信息进行分类：通过文件名。Lansdale 的研究表明，名称是一种无效的信息分类方式。尽管名称可以作为助记手段，但随着时间的推移，其价值会逐渐衰减。Carroll (1982) 发现

在短时间内,用户的命名模式变得不一致,导致检索困难。

归档

旧信息通常没有新信息有价值 但它往往是必不可少的。

我们都能回忆起需要记住的信息的场合

上个月才删除的。遗憾的是,如今的软件系统并不容易存档个人信息,也没有提供便捷的方法来检索已存档的内容。Whittaker 和 Sidner (1996) 引用了一位用户描述其困难的言论:“我不愿意存档垃圾……”

我知道存档垃圾的后果是让我们更难找到好东西。

…… 结果是:用户只能自行设计方

案或使用第三方应用程序。这两种方法都无法在检索任务中取得令人满意的效果。

更糟糕的是,用户经常会删除旧信息,而不是被迫处理存储信息或发明归档方案所带来的后果 (Erickson 1991;Barreau 和 Nardi 1995)。这既令人遗憾,又极具讽刺意味。计算机应该让信息归档变得更容易、更便宜。如今,几乎任何想要 TB 存储空间的人都能拥有。硬件履行了它的义务,但软件 (一如既往地)却让用户失望。

“提醒”

Malone (1983) 指出了“提醒”在我们纸质系统中的重要性,并建议将提醒功能纳入软件中。然而,桌面系统对提醒功能的支持仍然很少。尽管市场上出现了许多时间管理、日程安排和待办事项列表应用程序,但它们并未提供针对提醒问题的通用解决方案。

在最近的研究中,Barreau 和 Nardi (1995) 观察到,台式电脑用户经常将文件在桌面上的位置作为一种重要的提醒功能。例如,在一天结束时,一位 Macintosh 用户可能会将文件留在桌面上,以提醒第二天早上要完成的工作。其他人出于同样的原因,也会在收件箱中留下电子邮件 (Whittaker 和 Sidner 1996)。Lansdale 发现这种行为很大程度上是特殊的。我们注意到,这种基于位置的提醒方法很容易被破坏。无论如何,由于桌面隐喻本身没有提醒的语义概念,因此将电子

那些散落在“战略要地”的文件只是在匆忙应对而已。想必我们的软件应该能做得更好。

总结

总结是一项至关重要的信息处理任务。总结可以简化文档或文档集，从而减少信息量。

用户必须处理的信息（Klark and Manber 1995）。它们最终还允许用户“访问和控制信息洪流”；“摘要节省时间”（Hutchins 1995）。

信息摘要显然并非新鲜事物 然而，如今支持自动摘要的电子系统却寥寥无几。目前的桌面系统并未提供通用的摘要支持；它们将这项工作留给了专用应用程序。

我们认为，这种支持不足的部分原因是，当前桌面计算的视角狭隘，以应用程序为中心 工作重点是开发应用程序内部的工具，而不是从系统层面全面改善用户的信息访问。用户可以通过专用产品（例如 Intuit 的 Quick-en，它允许创建财务信息概览）获取摘要。但用户也需要摘要来实现更日常的用途。

超越桌面隐喻

我们更倾向于用隐喻而不是Nelson（1990）的虚拟性概念来进行软件设计。隐喻是一种基于将软件与现实世界中的物体或机器（例如，与办公家具世界中的物理桌面）进行比较来构建软件的方法。

隐喻在某些情况下很有用，但也可能限制设计：一旦选定了隐喻，系统的每个部分都必须在隐喻中扮演适当的角色。当设计师被迫在隐喻中添加意想不到的功能（例如弹出CD）时，解决方案可能会令人困惑，甚至荒谬。（为什么将CD图标拖到垃圾箱会导致CD弹出？用户不想扔掉CD。）尼尔森认为，“墨守成规会阻碍真正新事物的出现。”

另一方面，虚拟性是统一思想的建构，可以用丰富的图形表达来体现，而不仅仅是隐喻

对于一个现有的物理系统来说,这些想法反而(正如尼尔森所言)能够引领新的组织策略的发明。这正是我们设计Lifestreams的目标:提供一个简单、统一的系统,让用户轻松掌握,并且不受任何现实世界隐喻的束缚。

为了创建统一的模型,我们首先遵循以下指导原则,首先由我们自己的设计意识驱动,其次由我们在HCI和人为因素社区中发现的结果驱动。

存储应该是透明的,在创建文件时命名文件并将其塞入文件夹中是无意义开销的两个主要例子。

只有当用户想命名时才应该命名。在现实世界中,“正式文档”(章节、论文、拨款提案、书籍、诗歌)通常有名称,而“非正式文档”(草稿、信函、列表、计算、提醒事项)通常没有名称。随着计算变得越来越普遍,我们电子文档中越来越大的比例是“非正式的”,而要求我们为每一份文档命名变得越来越荒谬。当你拿起一张纸时,你没有必要在开始书写之前给它命名或决定将它存放在哪里。在电子桌面上,许多(可以说是大多数)文件名不仅仅是毫无意义的(例如,“draft1.

文件夹名称通常被称为“draft1.doc”、“draft2.doc”),但对于检索而言毫无用处。文件夹名称只有在用户能够记住它们的情况下才能够作为有效的检索线索,而用户通常不会记住太长。

文件夹和目录是不够完善的组织技术。我们的电子桌面过于忠实于纸质世界:它们强制每个文档只能存储在一个文件夹中。(至少对于新手来说是这样;只有专家才熟悉文件别名之类的概念)

而且即使对于专家来说,使用它们也很笨拙。)在电子世界中,文档可以而且应该被允许同时存储在多个位置。例如,在Lifestreams系统中,Lifestreams上的PowerPoint演示文稿可能同时存储在“Lifestreams”文件夹、“演示文稿”文件夹和“当前任务”文件夹中。传统的软件系统强制用户将信息存储在静态类别(即目录)中。但我们往往直到需要信息时才知道它属于哪里。(史密斯、皮菲尔和施瓦茨出席了一场讨论Mac版Zowie应用程序的会议,会议记录可能……

存储在传统系统中名为“Zowie”或“Mac 应用程序”的文件夹中。回想起来,你可能需要查阅 Piffel 参加的所有会议的记录。但假设你当时并不知道这一点。由于这个显而易见的原因,大脑对记忆的分类是动态的,而不是静态的。我可以让你回忆“在 300 房间举行的所有会议”,即使你从未有意识地按房间号对会议进行分类。)简而言之,信息应该根据需要进行组织,而不是在创建时预先确定。目录应该按需创建,文档应该属于尽可能多的适当目录。

归档应自动化。当前的系统在数据归档方面表现糟糕(尤其与纸质系统相比)。以桌面系统为例,用户有责任创建归档方案并遵循该方案。面对这项任务,许多用户干脆丢弃旧数据,而不是将其归档(归档后还要费力寻找)。软件应该允许文档优雅地老化,并在不常用时进行归档,但同时允许用户快速检索任何已归档的项目。

计算机应该将“提醒”功能融入桌面体验。提醒功能是计算机系统的关键功能(Malone 1983),这一点早已为人所知;然而,该功能目前仅存在于日历和任务管理器等第三方应用程序中,尚未融入我们集成的电子环境。用户研究指出,用户会运用多种应对策略来实现某种提醒功能(即使依赖第三方应用程序的用户也是如此)。提醒功能应该成为任何电子信息系统的一项基本功能。

个人数据应该随时随地、任何设备都能访问,并且兼容性应该是自动的。1994年,我们意识到用户需要通过多种联网设备访问、查看和管理他们的信息。当时,这包括新兴的平板电脑,它引领了诸如Palm-Pilot和微软PocketPC等PDA(个人数字助理)的发展。如今,我们拥有一整套全新的联网设备,其中以手机为主导。
个人电子设备

无论我们选择哪款联网设备,信息都应该能够访问。因此,我们的信息管理模型不仅要适应高分辨率的PC显示屏,还要适应小型、低分辨率的设备。

系统应该提供一种手段,将一组文档汇总成一个简明的概述。管理信息的一个重要方面是能够构建该信息的“全景”视图。例如:共同基金收盘价的时间序列可以用历史图表来概括。一组歌曲可以汇总成一个播放列表,并可打印到CD光盘盒中。一组工资单和付款可以汇总成一份部分完成的纳税申报表。等等。如果我们富有想象力地定义“汇总”功能,它可以(也应该)成为一种非常强大的功能。我们的软件应该包括一个复杂的汇总例程,并支持使用汇总数据作为输入的高阶操作(如数据挖掘和分析)。

掌握这些指导原则后,让我们看看Lifestreams模型。

生命流模型

让我们从Lifestreams模型的基本数据结构开始:

生活流是一个按时间顺序排列的文档流,就像你的电子生活日记一样。(“文档”指的是电子文档,其定义非常广泛:照片、视频、音频或应用程序都可以称为“文档”。)

你创建或接收的每一份文档都存储在你的生活流中(见图2.1)。生活流的尾部包含过去的文档(可能从你的电子出生证明开始)。从尾部向现在,你的生活流包含更新的文档。例如,正在处理的论文或新的电子邮件。每份文档都存储在你首次创建或首次接收时。因此,所有文档(图片、信件、账单、电影、语音邮件和软件)都存储在沿途的相应位置。从现在到未来,生活流包含你需要的文档:提醒事项、约会、日历事项、待办事项列表。

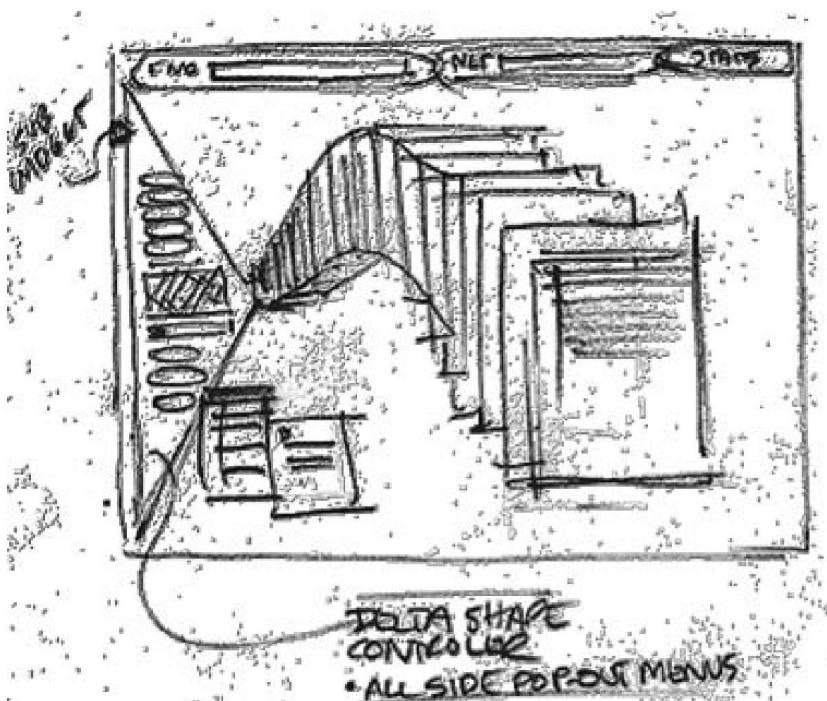


图2.1
生命流的早期概念图。

在这个模型中,我们添加了少量的操作,结合时间排序流,以自然的方式实现透明存储、按需通过目录组织、归档、提醒和摘要。

要创建文档,用户使用以下两个操作之一:新建和复制。
新建操作会创建一个新的空文档,并将其添加到流的头部。(每个流都有一个当前行,以当前时间标记。随着流的老化,当前行会逐渐远离尾部 即远离流中的第一个文档。新建操作会创建一个空文档,并将其添加到流的当前行。)复制操作会获取一个现有文档,创建一个副本,并将其添加到流的当前行。复制操作中的源文档可以位于不同的流中;换句话说,复制操作可用于传输文档的副本。

在流之间。创建过程始终是“透明的”，因为默认情况下，文档始终添加到流的末尾（在当前行），并且不必命名（除非用户选择命名）或塞入文件夹中。

生活流可以通过查找操作即时组织。查找

提示输入搜索查询，例如“来自 Piffel 的电子邮件”或“Leigh Nash 的 mp3”，并创建一个子流。子流呈现的是整个流的“视图”包含所有相关文档的视图

到搜索查询。子流与传统的目录系统不同：用户不会将文档放入静态子流中（尽管在某些情况下，他们可以手动将文档添加到子流中）。

相反，相关文档的虚拟组是根据需要动态创建的。

文档实际上并不存储在子流中。子流是主流中已存在的文档的临时集合。

子流可能会重叠；它们可以在不影响主流或其他子流的情况下动态创建和销毁。

子流不仅仅是搜索操作的结果。查找

find不仅仅是返回结果列表；它会创建一个新的数据结构，即子流。如果你允许子流持久化，它会在你创建或从外部获取符合搜索条件的新文档时收集它们。因此，子流是一种监控信息的自然方式 它不仅仅是一个组织工具

而是作为传入信息的过滤器。例如：使用查询“查找 Piffel 创建的所有文档”创建的子流将在 Piffel 收到新邮件时进行收集。

最后一个操作，汇总 (summarize)，将子流压缩成概览文档。概览文档的形式和内容取决于子流中文档的类型。例如，如果某个子流包含你投资组合中所有股票和共同基金的每日收盘价，则概览文档可能包含一个图表，显示你持有的证券的历史表现以及你的净资产。如果子流包含你必须完成的任务列表，则概览文档可能是一个按优先级排序的“待办事项”列表。如果子流包含“Leigh Nash”的所有 mp3，则概览文档可能是一个类似 iTunes 的可打印播放列表。

时间顺序作为存储模型

关于Lifestreams,最常见的问题（和批评）之一一直是：“为什么要用创建时间作为存储的基础？其他指标不是更有用吗？创建时间会不会太过局限？”

Lifestreams模型的一些实现允许你根据其他键值对数据流进行重新排序 标题、大小，任何你想要的键值。但这无关紧要。真正的重点在于：人类生活在时间中；他们的经历存在于时间中；Lifestreams本身就是一本电子日记。

它不仅仅是一个存放信息的文件柜；它记录着你每日经历的点点滴滴。这样的记录本身就很有用 这就是人们记录日记和日志的原因。2数据流会为其包含的所有内容添加历史背景；就像日记一样，数据流记录了你的工作、业务交易和思维的流程 而这一功能在当今的操作系统中基本上缺失了。我们通常需要的不仅仅是孤立的过去文档；我们需要了解这些文档的创建背景。

通常，我们无法理解一份一年前的文献（一行电子邮件、一份简短的说明、不完整的会议记录），除非我们能够了解它出现的背景。

事实上，历史背景在组织环境中至关重要（Cook 1995）。但目前大多数系统很少追踪文档的创建和删除时间、地点和原因。

请注意，基于时间的流也为我们提供了三种自然的文档分类：过去、现在和未来。流的“现在”部分是指当前线及其后（即比当前线更早）的区域。它保存着“工作文档”；这也是新文档通常创建和存放传入文档的地方。随着文档老化和新文档的添加，旧文档会逐渐从用户的视野中消失，并在此过程中被“归档”。（这里我们指的是概念意义上的归档；用户不必担心旧信息会弄乱桌面或妨碍工作。如果将来某个时间点他们需要归档的信息，可以使用“查找”功能找到。）流的“未来”部分允许在未来创建文档。“未来创建”是发布提醒和日程安排信息的一种自然方法。系统允许用户拨入未来，并使用“新建”操作将文档存放在那里 比如会议提醒。“未来”是一种特别方便的处理方式。

在你太忙而无法回复时收到的邮件。你可以将这样的邮件复制到未来,例如“明天早上”。明天早上,邮件副本就会出现在“现在”栏;邮件本身就起到了提醒的作用。(雅虎也添加了一个类似于 Lifestreams 操作的功能。)

统一

正如我们之前提到的,我们的目标是提供一个简单、统一的系统,方便用户掌握,并解决归档、查找、提醒、存档和汇总等问题。感兴趣的读者可以参考之前的著作,尤其是 Lifestreams 的论文 (Freeman 1997),以了解更多详情。但重要的是要思考该模型是如何实现我们的目标的。

首先,命名。在 Lifestreams 中,没有命名的概念。Lifestreams 以透明的方式存储信息:每当创建或接收文档时,它都会自动添加到流中。这一过程减少了创建信息的开销(正如我们提到的,创建信息是用户最困难且最终效率最低的任务之一)。用户可以专注于手头的任务,而无需纠结于特定文档或数据项的名称、文件夹、磁盘、计算机或网络位置。

第二,归档。Lifestreams 将信息存储的关键字锁定在信息创建或到达的时间,并根据信息到达或创建时的上下文进行组织。如何组织?为了组织信息,用户需要执行“查找”,这实际上会创建一个虚拟目录(子流)。

与目录或文件夹不同,子流不会将信息归类到特定位置。文档可以同时存在于多个子流中。通过消除命名和归档,Lifestreams 模型减少了创建信息的开销,提高了召回率,并使检索更加便捷。Lifestreams 的文档组织方法还有第二个优势:一旦创建了子流,您就可以让它持久化并(自动)成为过滤器或监视器,累积任何符合其搜索条件的新文档。

提醒功能是 Lifestreams 最新颖的功能之一。通过将信息流延伸到未来,系统允许在未来创建、复制或保存文档。当它们的时间到了,这些文档就会自然而然地变成提醒器。(我们稍后会在界面中看到它是如何运作的。)我们相信,Lifestreams 是第一个通用信息

模型将提醒者视为第一类实体，并提供自然适应提醒的隐喻。

Lifestreams 还通过其按时间组织的流解决了概念上的“归档问题”。随着文档的老化，它们会从用户的视野中移出（我们稍后会看到界面如何处理这个问题），并退回到流的过去。结果是一种自然的方式，可以将不再需要的数据移出视野，同时保留其可供将来检索。

Lifestreams 还通过提供用于创建执行摘要和概述的架构框架，为用户提供了一个新的机会来利用文档集合中存在的关系和全局模式。

Lifestreams 本身并不关心数据挖掘或各种用于摘要生成的算法。相反，它提供了一个支持此类分析的数据结构。我们的耶鲁原型提供了几种生成摘要的方法，但这一领域在很大程度上仍未得到深入研究，因此并未纳入我们的商业系统。

另一种理解生活流统一性的方式是，思考如何通过几个简单的操作来管理你的整个电子生活。创建子流（查找）的操作也会创建你的邮箱、你的网页书签、你的所有 PowerPoint 演示文稿以及其中的一切。设想一个包含所有包含“生活流”一词的文档的子流。它将

包括您创建的文档、您收到的文档、Lifestreams 页面的网络书签、所有提及 Lifestreams 的电子邮件等等。在每种情况下，如果您可以创建子流，就可以创建一个持续收集新文档的持久过滤器。我们很快还会介绍如何一目了然地查看任何子流中收集了多少新信息。就像您在查看电子邮件一样。您可以随时预览未来内容，或者只需单击鼠标即可回顾过去；单击任何流上的“汇总”按钮，即可获得上下文相关的概览。

生命流界面

Lifestreams 界面的开发是该项目最重要且最具挑战性的环节之一。创建新界面需要探索一个广阔的设计空间，而我们对此才刚刚开始探索。

我们的耶鲁研究原型由一个运行在互联网上的客户端/服务器架构组成。服务器是Lifestreams的主力。

系统。它管理一个或多个流,存储所有流文档和子流。每个接口都充当客户端,并提供流的视图。

正如我们将要解释的,我们的界面呈现出了流的明确外观和感觉。但我们实际上对界面的外观并无定论;我们设想了许多不同的可能性。事实上,我们认为界面的外观和感觉在从机顶盒到高端工作站等各种计算平台上都会有显著差异。但每个界面都支持基本操作。(再次强调,该模型的目标之一是支持能够处理各种不同设备且功能差异巨大的界面。)

我们在耶鲁大学的工作探索了四种接口实现:X Windows 客户端、纯文本命令行客户端、PDA 实现和 Web 浏览器实现。X Windows 接口提供了丰富的图形界面(在当时);它实现了所有操作功能。ASCII 接口也实现了完整的 Lifestreams 模型,但采用了基于文本的类似邮件的界面。PDA 版本是在 Apple Newton 上实现的;由于 Newton 内存不足且带宽较低,它提供了基本的流访问。Lifestreams 的后期商业版本主要侧重于 Web 浏览器,但也包含了对手机和更现代的 PDA 的大量支持。

详细描述这些接口及其功能远远超出了本章的范围。但我们将研究 X Windows 接口,并简要介绍其他接口。与当今的 GUI 相比,X Windows 接口可能显得有些粗糙,但它的功能仍然无与伦比 即使是商业化的 Lifestreams 实现也无可比拟!

X Windows 界面

我们的 X Windows 界面如图 2.2 所示。该界面基于流隐喻的视觉呈现(让人联想到早期的 Lifestreams 草图,例如图 2.1 中的草图)。用户可以将鼠标指针滑过文档呈现,以“浏览”每个文档内容的缩略图,或者使用左下角的滚动条按时间向前或向后滚动。所有界面反馈

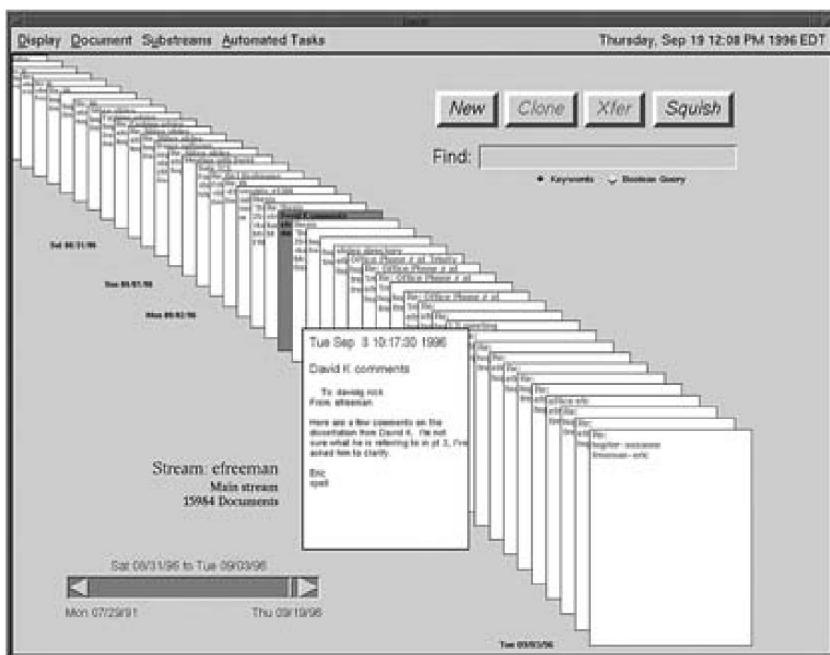


图2.2
Lifestreams 的 X Windows 界面。

返回（包括滚动时间）是即时的并且接近实时。

我们为重要的文档功能添加了颜色和动画效果。红色边框表示“未显示”，粗体边框表示“可写”。打开的文档会向一侧偏移，以显示正在编辑。我们还使用了外部辅助应用程序来查看和编辑文档，这大大加快了 Lifestreams 用户的学习速度。他们可以使用熟悉的应用程序（例如 emacs.xv 和 ghost-view）来创建和查看文档，同时使用 Lifestreams 来组织和交流文档。Lifestreams（文档组织模型）与文档创建和文档查看应用程序相互独立。

传入的文档通过动画从左侧滑入，并伴有“嗖”的一声。新创建的文档从顶部弹出，并将文档流向后推一个文档。要查看（或编辑）文档，用户只需点击其对应的文档即可。

界面突出显示了主要的系统操作

新建、克隆（我们最初对“复制”的称呼）、Xfer（即传输 复制到另一个流）、查找、压缩（即汇总）以及一些实用的辅助操作,例如按钮和菜单。“新建”按钮创建一个新文档并将其添加到流中。“克隆”按钮复制现有文档并将副本添加到流中。“Xfer”按钮会提示用户输入一个或多个地址,然后转发文档（根据需要转发到另一个流或电子邮件）。

“查找”功能通过文本输入框支持,用户可以在其中输入布尔搜索查询（或关键字搜索）。“查找”会创建并显示一个新的子流,如图 2.3 所示。图中我们搜索了“david and meme”这两个词,与查询匹配的文档构成了子流。如果在此期间有新的文档与该查询匹配,它会直接滑入子流就像滑入主流一样。

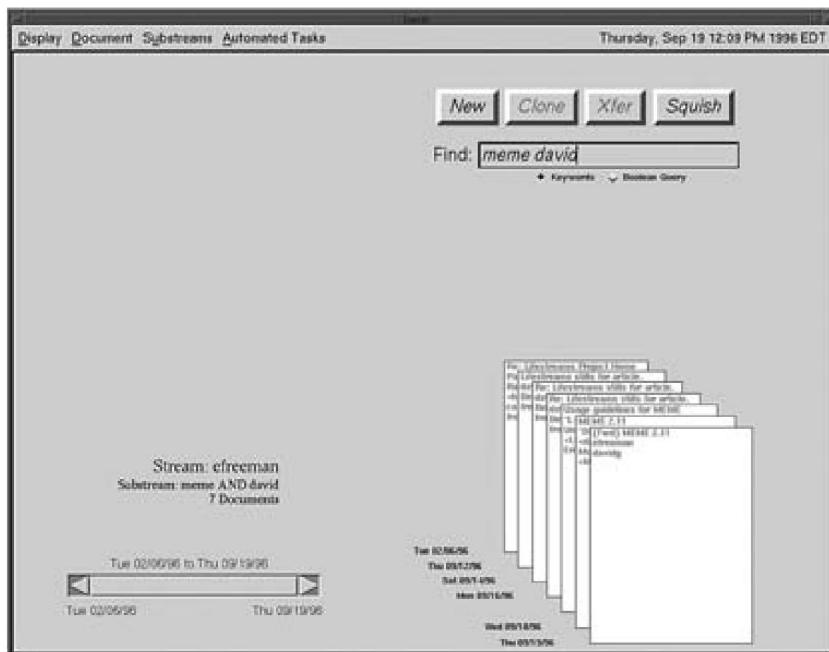


图2.3

创建并显示新的子流。

菜单操作用于选择持久子流、创建摘要和穿越时间（我们将很快解释此操作）。图2.4显示了子流菜单，该菜单分为三个部分。第一部分包含可对子流执行的操作列表（例如“删除”）。第二部分包含一个名为“您的生活流”的条目，并重点显示您的整个生活流（即您的所有文档）。最后一部分列出了您的所有子流。请注意，子流可以增量创建 - 这将生成一组嵌套的菜单。在此示例中，嵌套菜单的创建方式是：首先从主流创建一个子流“lifestreams and david”，然后从该子流增量创建两个子流：“scenarios”和“ben”。最后，从“scenarios”子流创建了子流“pda”。从语义上讲，这种增量子流化相当于将每个新查询与前一个子流的查询进行布尔与运算。

虽然这看起来像是一个典型的信息层级结构，但请注意，同一篇文档可能出现在多个流中。您可以随时使用“移除”菜单项移除子流，但如果子流保留下

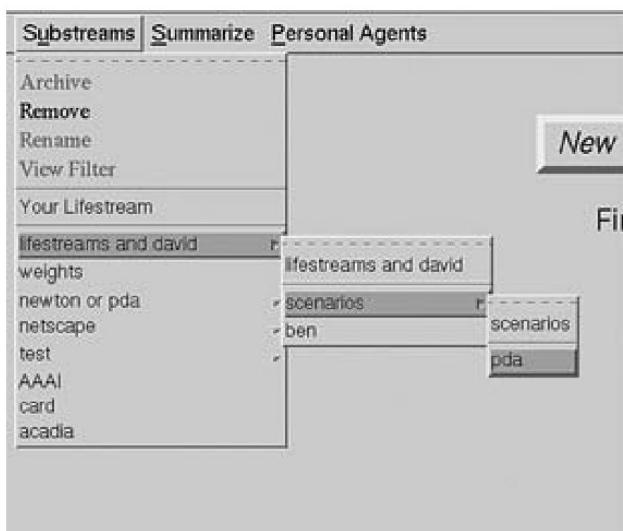


图2.4
子流菜单。

它将继续积累符合其搜索条件的新文件

它们被添加到主流中。

Lifestreams 在界面右上角显示时间。此时间显示也充当菜单（图 2.5），允许用户通过基于日历的对话框将界面时间设置为未来或过去。想象一下，一个光标始终指向流中的某个位置，使得该点之后朝向头部的所有文档（参见图 2.6）都具有未来时间戳，而该点之前朝向尾部的所有文档都具有过去时间戳。随着时间的推移，此光标向前移动到头部；当它越过“未来”文档时，它们会被添加到流的可见部分，就像新文档到达一样。

将时间设置为未来或过去，其效果是将时间光标暂时重置到用户指定的固定位置。通常情况下，用户界面会显示从过去到时间光标的所有文档。

将时间光标设置为未来，用户可以查看流中“未来”部分的文档。以此模式（即“未来”）创建文档将生成带有未来时间戳的文档。用户完成时间跳转后，可以通过选择时间菜单中的“将时间设置为现在”菜单选项重置为现在。

图 2.7 演示了摘要操作（在此版本中称为Squish）；该图显示了包含每日



图2.5
设置接口时间。

早期互联网投资组合服务中的股票收盘价。摘要图表将这些值随时间的变化绘制成图表。

摘要与上下文相关,因此,虽然一个生活流可能包含多个摘要,但只有与特定子流相关的摘要才会作为可能的操作呈现给用户。在此实现中,“挤压”按钮实际上已更改为“挤压股票”,以指示特定摘要可用。如果有多个摘要合适,则会向用户显示一个选项列表。最后,如果没有合适的摘要,则“挤压”按钮将保持灰色。

常见任务

Lifestreams 可用于处理常见的计算机任务,例如通信、创建提醒、管理日程安排、追踪联系人以及管理个人财务(仅举几例)。例如,在 Lifestreams 中使用电子邮件与用户已经习惯的邮件使用方式并无太大区别。要发送消息,用户只需创建

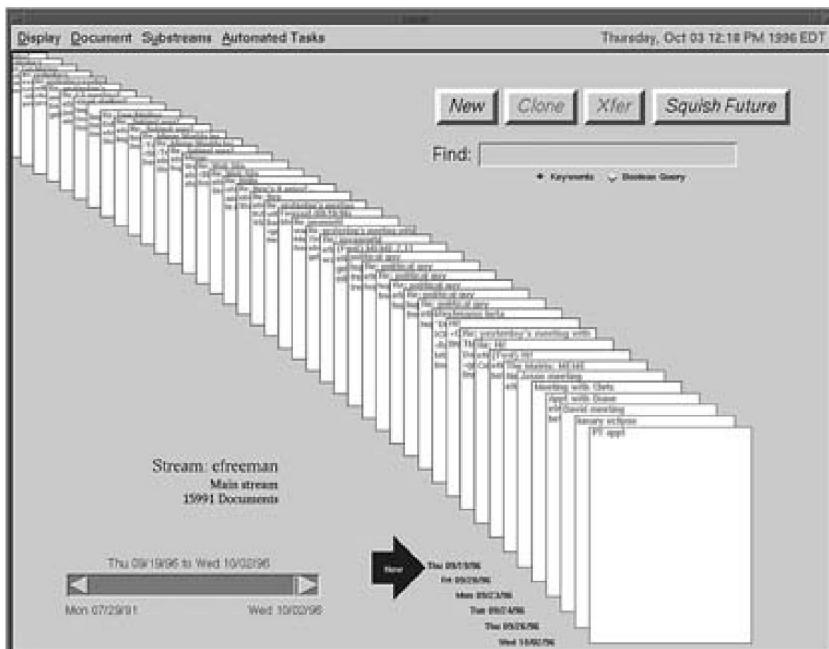


图2.6
显示具有未来时间戳的文档。



图2.7
概要操作。

创建一个新文档（点击“新建”按钮），然后使用常用编辑器撰写邮件。之后，只需按下“Xfer”按钮即可发送邮件。同样，现有文档也可以轻松转发给其他用户，或者克隆文档并进行回复。

虽然所有电子邮件消息（传入和传出）都与流中的其他文档混合在一起，但用户可以通过对其他用户创建的文档进行子流化来轻松创建邮箱；或者，用户可以更进一步，创建包含邮箱子流子集的子流，例如“来自 Bob 的所有邮件”或“我尚未回复的所有邮件”。

再举一个例子，通过拨号到未来并存放作为提醒的文档，可以轻松创建提醒器（我们在原型中将此步骤自动化为一个步骤）。用户还可以发送未来到达的邮件。如果他在撰写消息之前“拨号”到未来，那么当消息被传输时，它不会出现在收件人的信息流中，直到到达未来时间或收件人碰巧将界面拨号到

设置创建日期。目前,文档将位于流数据结构中,但界面不会显示。我们利用此功能发送

邮件提醒功能,用于向他人发送重要会议、部门会谈等提醒。由于这些提醒“即时生效”,无需用户切换到其他应用程序,因此比单独包含在日历或日程安排实用程序中的提醒更有效。

论文中涵盖了许多其他常见任务的示例,我们请感兴趣的读者参阅,因为详细的描述本身就足以写满整整一章。

替代接口

我们之前提到过,Lifestreams 模型的目标之一是扩展到台式电脑以外的其他设备的功能。相比之下,如何在手机上运用桌面隐喻呢?

此外,我们希望提供一种通用的数据结构,以便探索多种类型的界面。除了基于文本和PDA的实现之外,我们还在实际实现和纸面上对其他界面进行了大量的探索。例如,图2.8展示了一个功能齐全的日历界面,它是耶鲁大学的一个高级项目(Larratt-Smith 1996)实现的,它提供了“流视图”界面的替代方案。这个界面对于提醒和安排任务特别方便。

我们还设想了更具雄心的界面,这些界面在当时超越了当前的技术(以及我们自己的图形编码技能)。

如图 2.9 所示,其中一个例子是由 1997 年与我们合作的图形设计师 Jim Dustin 创建的,它看起来非常像现代的 Apple OS X 应用程序。

总而言之,虽然我们的界面体验相当多样化,
用户界面设计领域仍有很大一部分尚未开发。

分析

虽然 Lifestreams 背后的许多动机和想法在今天都是常识,但在 20 世纪 90 年代初期到中期,它们被认为有点边缘。

在接受《技术评论》采访时,戴维·格勒恩特(David Gelernter)这样描述埃里克·弗里曼(Eric Freeman)的作品:“这是一次冒险的、彻底的背离,而不是

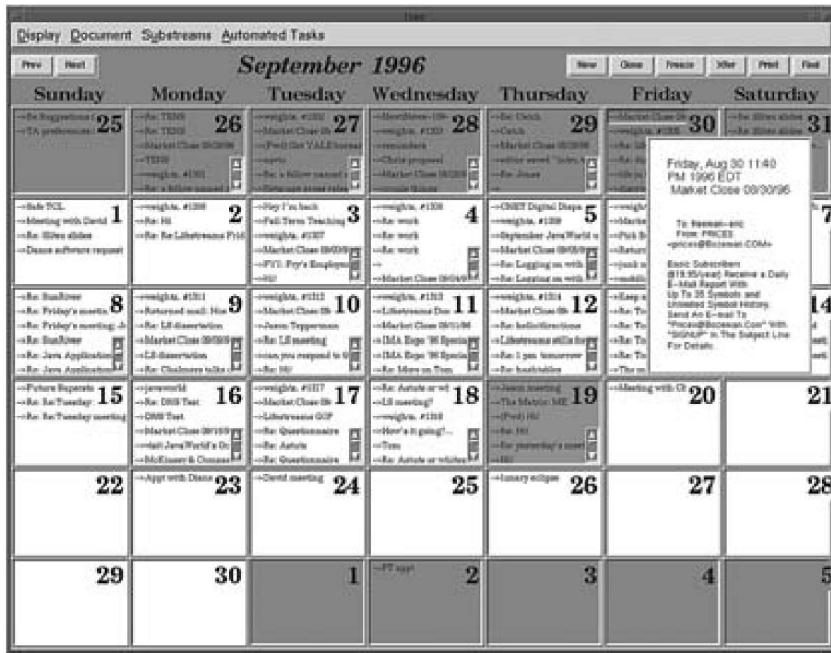


图2.8日历界
面。

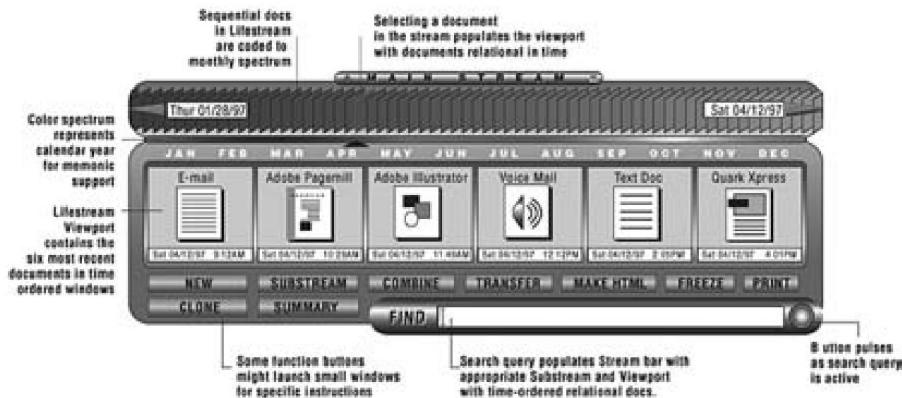


图 2.9 另一个 Lifestreams 界面。

“渐进式改进”（《技术评论》1999）。尽管大卫擅长赞美，但在一个主要从事理论研究的计算机科学系（没有HCI领域的教职员）开展生命流项目无疑感觉很危险。

3

虽然我们关心教职员对这项工作的接受程度，但总的来说，我们更关心日常用户的反应。我们对Lifestreams的反馈主要来自三个群体：计算机科学系本身的一小部分用户（我们不仅招募了技术娴熟的用户，还招募了系里的行政人员）；在大众媒体上听说过Lifestreams但从未使用过该系统的人群；以及相当多的计算机和金融行业人士。我们曾联系他们，并向他们演示了系统，以筹集博士后创业资金。⁴我们的大部分分析来自耶鲁大学的行政人员群体，他们长期使用我们的原型系统。

为了衡量用户满意度，我们采用了用户满意度问卷(QUIS)，这是马里兰大学授权使用的标准化可用性测试工具。用户对系统的总体主观评价较高。以下用户评论反映了较高的满意度，并且与我们收到的其他用户群体的反馈一致：

(生命流)这个概念立刻在两个层面吸引了我。首先，因为我知道自己会根据时间线索来自然地整理和回忆生活中的事件，这些“记忆”对我的日常活动的重要性会随着时间推移而降低（但由于与当前事件的相似性，它们在回忆时仍然具有冲击力和适用性）。而且，如果因为“索引”丢失而无法回忆起一些可能有用的内容，或者因为相关文件被扔掉了（几周前为了清理“杂物”或节省空间而扔掉），我会感到非常恼火。

用户也很快就“掌握”了该系统，并在 QUIS 评分中对系统的快速学习能力给出了最高分。一位耶鲁大学行政人员用户直接谈到了这一点（原文中有下划线）：

我开始使用 Lifestreams 是在学期初，也就是我最忙的时候。。。

考虑到所有这些，我仍然对它在很短的时间内能够并且确实使我的工作变得更轻松的所有方式感到震惊。_____

我们还发现有证据表明该系统确实对用户的观点产生了影响
管理电子信息：

我现在花在这个系统上的时间真的变多了。我讨厌在这棵“树”里翻来覆去地找。往好了说是繁琐，往好了说是烦人。我已经找到了更好的办法。我之前都没意识到自己花了多少时间搜索文档。

除了用户研究之外，我们还发现该系统本身就“打动”了那些从未使用过该系统但在大众媒体上读到过相关信息的人。类似以下的电子邮件相当常见：

我的“收件箱”里每天都塞满东西，无论是收到的电子邮件、普通邮件、电话留言、文章，还是其他什么，我都没时间整理。正如你一针见血地指出的那样，我宁愿把所有东西都存起来（因为存储很便宜！），只在需要的时候才访问。

除了 QUIS 调查之外，我们还对 Lifestreams 进行了测量，以便定量衡量系统的有效性。

总体而言，数据表明，子流是一种有效且高效的信息定位机制（尽管某些改进可以通过更先进的索引技术来实现）。您会在论文中详细探讨这些问题以及其他问题，例如我们在观察系统用户时发现的多样化用户风格。

总而言之，Lifestreams 似乎不仅在实际用户中引发了高度积极的主观反应，在那些仅仅设想使用它的用户中也同样如此。进一步的定量分析进一步表明，Lifestreams 作为一种信息管理工具确实非常有效。尽管这些结果令人鼓舞，但在研究 Lifestreams 在用户中的实用性方面仍有许多工作要做，尤其是与传统桌面系统进行比较。

不同意意见

关于 Lifestreams（或者更准确地说，是其基础）的最有趣的讨论之一是 SIGCHI Bulletin 上的一场简短的公开辩论

与 Deborah Barreau 和 Bonnie Nardi 合作，基于她们的研究（该研究考察了桌面隐喻的用户，并根据他们的工作习惯得出结论）。Barreau 和 Nardi (1995) 的研究尤其让我们感兴趣，因为他们观察到了许多与我们相同的用户行为。更具体地说，她们对 22 名受试者进行的研究发现，桌面用户之间存在以下相似之处：

- 倾向于使用基于位置的搜索来查找文件（与基于逻辑的文本搜索相反）；

- 使用文件放置作为关键提醒功能；以及

- 归档文件的“缺乏重要性”。

我们也对此很感兴趣，因为从表面上看，它显然与我们最初的猜想（以及Lansdale和Malone之前的研究）相矛盾。虽然我们相信他们的发现是有效的，但我们发现从用户行为推断用户偏好的做法具有误导性。Steven Steinberg（1997）在《连线》杂志上评论道，从我们的角度来看，这项研究“就像研究那些听收音机的人，然后决定他们不想要图片一样”。

这正是我们所看到的。当然，1995年的用户更喜欢基于位置的搜索（即通过文件所在的文件夹查找文件），而不是逻辑搜索；他们还有什么选择呢？虽然当时桌面上确实有一些基于文本的搜索工具，但它们充其量也只是粗糙的。如今，随着桌面的“谷歌化”（以及近十年来通过使用WebCrawler、Infoseek、Altavista以及现在的谷歌对用户进行的教育），这种偏好似乎正在改变。

就基于位置的提醒而言，这是一个非常出色的结果，也体现了用户对提醒作为核心功能的需求。但是，就此断定位置是实现这一目标的最佳方式还为时过早。正如我们之前所说，肯定有更好的方法。

最后，声称归档文件不重要，似乎又一次是因为当时现有系统缺乏支持。归档文件显然很有价值，尤其是如果我们能找到方法使流程透明化，并改进查找已归档资料的方法。

最后，Barreau 和 Nardi（1995）观察了研究对象的另一个方面，指出他们的

- 使用三种类型的信息：临时信息、工作信息和存档信息。

我们仍然认为这是一个有趣且重要的结论；我们相信短暂的、工作中的和存档的对于桌面信息来说是至关重要的分类，并且这些分类为我们的软件系统需要支持哪些能力来管理这些不同类型的信息提供了线索。

商业努力

Lifestreams 虽然迄今为止在商业上取得的成就有限,但销量不俗,并拥有一批忠实的用户,其中许多人至今仍在使用该系统。这些商业系统主要面向企业用户(尽管它们仍然支持个人信息),并在 Web 浏览器中呈现 Lifestreams 界面。

自主作者于 2000 年离开公司以来,对该客户群的接触非常有限;然而,用户对该商业系统的反应大多是积极的,这与耶鲁用户研究的结果一致。对限制 Lifestreams 接受度的因素(超出任何营销或业务发展因素)的事后分析发现了与现有企业系统(如 Outlook/Exchange 和 Lotus Notes)以及底层操作系统集成的问题。最后一点不容小觑:Lifestreams 是桌面信息环境不可或缺的一部分,任何试图在应用程序或实用程序中附加此功能的尝试都不会像将类似 Lifestreams 的模型作为操作系统不可或缺的一部分实现那样强大(或被用户接受)(微软和苹果似乎都在朝着这个方向发展,在操作系统级别集成许多类似 Lifestreams 的功能)。

今日生活

90年代末,当我们忙于创办自己的Lifestreams创业公司时,我们曾宣称,即使我们的努力失败了,我们的想法也会成为主流,因为它是一个如此自然而然的想法。放眼如今,我们的公司已然消亡,但许多想法却蓬勃发展 例如Haystack、Chandler、MyLife-Bits、微软Longhorn的部分功能、苹果的Spotlight和Apps,以及谷歌桌面(仅举几例)。其中一些项目直接受到了Lifestreams的影响。但还有一股更强大的力量在起作用:信息管理需要一个比桌面隐喻更强大的底层模型。我们生活在时间之中。

Lifestreams 过去是、现在是、并且将继续是不可避免的。

我们的贡献(其次)是认识和识别桌面隐喻中固有的问题(基于 Malone 等研究人员的开创性工作以及我们自己的观察),主要是我们的 pro-

提出一种摆脱该隐喻束缚的新模式。如今,事情正朝着正确的方向发展。谷歌桌面表明,用户开始内化新的信息模型。

管理。

搜索已成为桌面领域的圣杯。领先的搜索公司纷纷进军桌面搜索领域,以扩大其影响力,操作系统制造商也正积极地朝这个方向发展。将搜索集成到桌面环境中并非新鲜事 微软断断续续地推进 WinFS 已有十多年 但这一次,情况有所不同:每个人都认为它很重要。

为什么人们对桌面搜索的兴趣日益增长?许多人将这种兴趣归因于桌面的“谷歌化”。他们的想法是:如果我的桌面变得像网页一样复杂且难以导航,为什么不把网页上的原则应用到我的桌面上呢?这并非坏主意,而且它的早期成功清楚地表明,我们需要新技术(或许还有新的隐喻)来帮助我们管理电子生活。

但“桌面搜索”真的是我们真正想要的吗?它能让我们最终管理涌入桌面的海量信息吗?我们在 90 年代初并不这么认为,现在我们也不这么认为。我们的主张是:要彻底解决这个问题,我们需要摆脱桌面隐喻,转向一种新的模式,消除桌面带来的开销和设计限制。桌面搜索是朝着正确方向迈出的一步;通过朝着这个方向发展,我们获得了一些有价值的东西:一种用户如何重新掌控自己电子生活的替代模式。在过去十年中,网络搜索实际上已经为用户提供了一种管理电子生活的方式。然而,正如我们所见,我们还没有到达那一步;搜索是这种模式的必要组成部分;但它本身还不够。只有当你知道你在寻找什么时,搜索才是一种解决方案。我们猜测,人们大约有一半的时间知道自己在寻找什么。其余时间,他们不需要一个好的搜索引擎;他们需要一个好的“浏览引擎”,一个好的显示界面。Lifestreams 仍然是我们遇到过的最好的“浏览引擎”。

正如我们在本章中详细介绍的那样,除了单纯的捕获和检索之外,还有很多需要改进的地方,因为桌面系统让我们失望了,

将在分类、提醒、归档、汇总等方面持续发力。

摆脱桌面隐喻将是一个缓慢的过程 用户在当前界面上投入了大量精力;然而,随着信息洪流的持续加速,摆脱它的压力也将随之增大。2006年,我们仍然处于1994年的境地:桌面一片混乱。但我们希望,我们以及其他人的努力已经引领我们走上了一条超越桌面隐喻的道路。而且(正如我们想指出的),你可能还没有听到Lifestreams的最后消息。

致谢

我们最诚挚地感谢 Elisabeth Freeman、Frank Weil 和 Tom Carulli 对本章的贡献。

笔记

1. 为了说明我们并非为了达到戏剧效果而夸大其词,1995年末的《连线》杂志《纽约时报》派了两位纽约摄影师去耶鲁大学为即将发表的专题文章拍摄照片。我们预料到了最坏的情况:他们会让我们摆出一些滑稽尴尬的姿势,让我们永远忘不了。摄影师准时到达,开始在计算机科学系寻找合适的拍摄地点。他们走进大卫的办公室,架起一个几乎到天花板的三脚架,拍下了大卫的办公桌。然后他们一言不发地离开了。

我们只能猜测,他们觉得桌面照片比我们自己的照片更能传达我们想要表达的意思。果然,你可以在1996年2月的《连线》杂志上找到这张照片(而且是整版)。

2. 值得一提的是,我们在1995年就曾说过“这种记录本身就很有用,这就是人们写日记或日志的原因”,但后来觉得需要补充一句:“至少人们过去是写日记的。”过去几年里,出现了一个有趣的现象:网络日志。如今,按时间顺序记录我们的生活日志再次变得普遍,这次是在网络上。我们不禁注意到,我们早期的一些基于浏览器的实现与当前的网络日志系统非常相似(或许是时候让我们将它们转向Lifestreams,让你的整个电子生活都被记录在一个大概是私密的日志中)。

3. Eric Freeman 非常荣幸能有 Ben Bederson 作为外部顾问,这项工作因此受益匪浅。此外,Ben 的参与也为耶鲁大学的教职员提供了必要的可信度,使他们相信这项工作在人机交互方面具有价值。

4. 请注意,在攻读博士学位期间筹集风险投资可以提供一种有效的方法,让你的工作成果呈现在更多人面前。
5. 显然,《连线》杂志想通过暗示这场辩论充满争议来使其在《生活流》的文章更有趣。这完全不符合事实。事实上,我们非常享受这场辩论,尽管我们不同意他们的结论,但我们从巴罗和纳尔迪的著作中受益匪浅。

参考

- Barreau, D. 和 Nardi, B. (1995).查找与提醒:桌面文件整理。ACM SIGCHI 通讯27: 39–43。
- Carroll, JM(1982)。学习、使用和设计文件名及命令范例。行为与信息技术1(4): 327–346。
- Cook, T. (1995). 你知道你的数据在哪里吗? 《科技评论》 (1月)。
- Dumais, ST, and Landauer, TK (1983).使用示例进行描述分类。
1983年ACM SIGCHI计算机系统人为因素会议 (CHI 83)论文集,第112-115页。马萨诸塞州波士顿,12月12-15日。
- Erickson, T. (1991). 设计桌面信息系统:观察与问题。载于1991年ACM SIGCHI计算机系统人为因素会议(CHI 91)论文集,第49-54页。路易斯安那州新奥尔良,4月27日至5月2日。
- Freeman, ET (1997). Lifestreams 软件架构。耶鲁大学计算机科学系博士论文,1997 年 5 月。
可访问 <http://www.cs.yale.edu/homes/freeman/lifestreams.html/>。
- Gelernter, D. (1994). 未选择的网络之路。《华盛顿邮报》 (4月3日) :C1版。
- Hutchins, J. (1995). 文本摘要简介。达格斯图尔研讨会报告,IBFI,达格斯图尔。
- Jones, WP 和 Dumais, ST (1983).空间隐喻:位置指称与名称指称的实验测试。ACM办公信息系统学报4 (1): 43–63。
- Klark, P. 和 Manber, U. (1995)。《开发个人互联网助理》。载于ED-Media 95 世界多媒体与超媒体大会论文集,
第 372-377 页。奥地利格拉茨,6月 18 日至 21 日。
- Lansdale, M. (1988).个人信息管理心理学。
应用人体工程学19(1):55–66。
- Malone, T. (1983). 人们如何整理办公桌?对办公信息系统设计的启示。ACM办公系统汇刊1(1): 99–112。
- Nelson, T. (1990). 思考软件设计的正确方法。载于 Laurel, B. 主编的《人机界面设计艺术》,第 235–243 页。波士顿:Addison-Wesley 出版社。

Larratt-Smith, A. (1996). Lifestreams 的日历界面。耶鲁大学计算机科学系高级项目技术报告,1996 年 5 月。

Steinberg, SG (1997). 生命流。《连线》 5: 148–151, 204–209。

《技术评论》TR100。(1999)。《技术评论》(11月/12月)。

Whittaker, S. 和 Sidner, C. (1996)。电子邮件超载:探索电子邮件中的个人信息管理。载于1996 年 ACM SIGCHI 计算机系统人为因素会议 (CHI 96) 论文集,第 276-283 页。加拿大不列颠哥伦比亚省温哥华,4 月 13 日至 18 日。

3

Haystack:每个用户的信息 基于半结构化数据的环境

戴维·R·卡格

介绍

每个人都有自己的信息处理方式。具体来说，不同的用户在以下方面有不同的需求和偏好：

- 哪些信息对象需要存储、检索和查看；
 - 哪些关系和属性值得存储和记录以帮助以后查找信息；
 - 这些关系和属性应该如何呈现
- 检查物体并导航信息空间；
- 应该采取哪些行动来执行所呈现的信息；以及
 - 如何将信息收集到连贯的工作空间中
- 完成给定的任务。

目前，开发人员会做出这样的决定，并将其硬编码到应用程序中：选择应用程序要管理的特定对象类别，决定这些对象遵循的模式，开发这些信息对象的特定显示方式，并将它们与相关操作一起收集到特定的工作区中。Hay-stack 项目采用了不同的方法。我们认为，没有开发人员能够预测用户想要记录、注释和操作信息的所有方式，因此，应用程序硬编码的信息设计会干扰用户最有效地利用信息的能力。

我们的 Haystack 系统旨在让最终用户对上述四个方面拥有强大的控制权。
Haystack 存储（即参考

50 大卫·R·卡格

它能够将用户感兴趣的任意对象（例如，文本、图片、视频等）转换为用户感兴趣的任意对象。它记录存储信息的任意属性，以及与其他任意对象的关系。其用户界面灵活，能够以有意义的方式呈现存储的任何属性和关系。

为了让用户更灵活地存储和检索内容，Haystack 创造了一个统一资源标识符 (URI)，用于命名用户感兴趣的任何内容：数字文档、实体文档、人物、任务、命令或菜单操作、信息视图或创意。命名后，对象即可被注释、关联其他对象、查看和检索。

为了支持信息管理和检索，Haystack 用户可以记录任意（预定义或用户定义）属性，以捕获用户认为重要的信息片段的任何属性或关系。对象的属性通常是用户访问该对象时所寻找的信息。反过来，它们也可以帮助用户找到他们想要的对象：这些属性可以用作有用的查询参数，作为基于元数据的浏览的方面（Yee 等人，2003），或作为关系链接来支持万维网中典型的关联浏览。

Haystack 的用户界面旨在灵活适应信息空间：Haystack 不使用预定义的硬编码信息布局，而是解释视图处方，这些处方描述了不同类型信息的呈现方式。例如，哪些属性重要以及它们的值应如何（递归地）呈现。视图处方本身是系统中可定制的数据，因此用户可以导入或修改它们，以处理新类型的信息、信息的新属性或查看旧信息的新方式。添加新的关系甚至新的信息类型不需要以编程方式修改应用程序或创建新的应用程序；相反，可以添加易于编写的视图处方来描述如何将新信息无缝地融入现有信息视图中。

除了允许用户自定义他们使用的信息之外，Haystack 还允许用户自定义他们的信息管理活动。通过对部分完成的对话框进行“快照”，用户可以创建专门的操作，以常用的方式处理他们的数据。在更高层次上，使用视图处方方法的变体来定义特定用户任务的工作区，描述哪些信息

所涉及的对象、应如何布局以及可对它们执行哪些操作。借助 Haystack 的统一信息模型，任何异构对象集都可以整合到适合特定任务的一致可视化中。

灵活地整合新数据类型、呈现方式和聚合的需求并不局限于个人用户。正如本书所展示的，研究人员不断提出有益的新属性、关系和数据类型。Plaisant 等人（本书）建议用“角色”属性标记所有信息对象，该属性可用于确定给定信息对象在何种情况下相关，以及哪些操作应该可用。Fisher 和 Nardi（本书）提出，通过记录和显示信息对象与相关人员之间的联系，可以改进信息管理。Freeman 和 Gelernter（本书）主张根据用户所有信息对象的访问时间来记录和呈现信息。这些文章都提供了很好的案例，表明每种方法在某些时候都是正确的。Haystack 系统演示了一种基础设施，可以更轻松地将这些新想法整合到一个系统中，并在适当的时候调用它们，而不是为每个新想法制作新的、不同的应用程序（并说服用户迁移到这些应用程序）。

原则

Haystack 的设计遵循一系列原则和概念。其中许多原则和概念看似显而易见，甚至略显生硬。但所有这些原则和概念都可能存在争议，因此我们尝试在下面的动机部分中对其进行论证。

普遍性用户应该能够记录任何他们认为重要或有意义的信息对象，并且能够在以后寻找、查找和查看它。

元数据和关系的核心地位很多对象的检索都基于回忆对象的特定属性及其与其他对象的关系。因此，系统必须能够记录用户关注的所有属性和关系，并显示它们，并支持在搜索和导航中使用它们。

52 大卫·R·卡格

单一信息空间用户信息不应按“类型”或应用预先划分。相反，所有信息应逻辑地存在于单一空间中。用户应该能够对所选的任何信息对象进行分组和关联。

个性化没有开发人员能够预测用户想要存储哪些类型的信息对象，或者哪些属性和关系对他们的检索有意义。因此，系统必须允许最终用户定义新的信息类型和属性，并根据这些类型和属性进行调整，以便能够存储、呈现和搜索。

语义身份:数据模型中，特定信息对象应该只有一个表示（而不是由不同的应用程序存储不同的表示）。该对象的任何可见表现形式都应是“活的”，能够提供对该对象的访问（而不是简单地充当必须位于其他地方的对象的死文本标签）。

数据与呈现分离应鼓励开发同一信息对象的多种视图，以便根据特定用途选择合适的视图。每个视图都应该能够在任何需要的场景下使用，而不是将每个视图限制在某些特定的应用中。

重用表示形式。许多类型（例如“电子邮件”）是更通用的类型（“消息”）的实例，这些类型具有其他形式（新闻组发布、即时消息、电话呼叫），并且许多属性（发件人、收件人、主题）和操作（回复、转发）统一应用于这些类型。我们应该尽可能将视图设计得通用，以便用户可以忽略与其信息管理需求无关的差异。

本章解释了这些原则的动机并描述了我们为应用这些原则而构建的系统。

干草堆之旅

为了开始探索 Haystack 的设计，我们先从最终用户的角度来简要介绍一下 Haystack。图 3.1 是 Haystack 的屏幕截图。

管理个人收件箱。与典型的电子邮件应用程序一样,Hay-stack 在主浏览窗格中显示用户的收件箱。布局采用表格形式,其中各列列出了发件人、主题、正文等信息。第四列“推荐类别”则不太常见,用户可以将其添加到显示中,方法是将右下角邮件中的“推荐类别”视图拖到收件箱列标题中。

与往常一样,该集合包含一个用于查看选定项目的“预览”窗格,该窗格当前处于折叠状态。

虽然 Haystack 收件箱看起来很像典型的电子邮件显示,但它包含更多内容。收件箱中的某些项目并非电子邮件信息,圣贤。这里有来自简易信息聚合 (RSS) 源的故事,甚至还有人物画像 或许是为了提醒用户需要与他见面而放置的。RSS 消息有发件人和日期,但人物没有。这正是 Haystack 的特色:

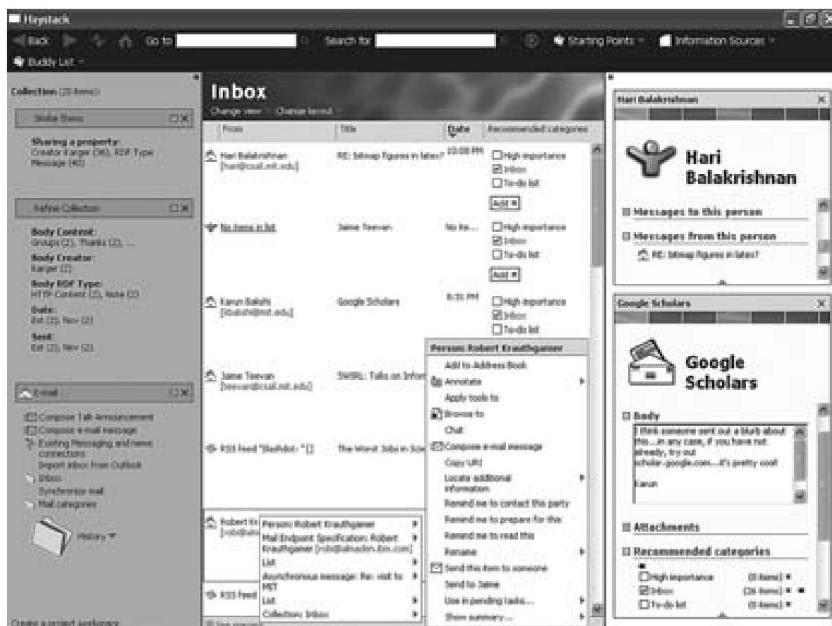


图3.1

Haystack 正在查看用户的收件箱集合。右侧显示一个人和一封电子邮件。用户右键单击名称“Robert Krauthgamer”,打开与该人相关的操作的上下文菜单。

54 大卫·R·卡格

收件箱并非密不可分地绑定到“电子邮件阅读器应用程序”，而是像所有其他 Haystack 集合一样的集合，唯一区别在于用户指定将收到的电子邮件（和新闻）放入哪个集合。它使用与所有其他集合相同的集合视图显示。任何项目都可以成为收件箱集合的一部分，并在查看收件箱时正确显示。这意味着收件箱可以用作通用的“待办事项列表”。Bellotti 等人（2003 年）观察到，许多用户强制使用电子邮件作为这一角色，但结果不得不应对只有电子邮件可以出现在待办事项列表中的限制；Haystack 取消了这一限制。

值得注意的是，RSS 在 Haystack 中“姗姗来迟”。收件箱视图是在 RSS 开发之前创建的。当我们决定将 RSS 故事作为一种新的信息类型纳入系统处理时，我们并没有对用户界面进行任何更改。

相反，我们只是添加了一个视图处方——一个小注释，解释 RSS 对象的哪些属性值得查看。Haystack 就能够立即合并这些故事，如图所示。

这是 Haystack 的标准：新类型的信息和新的信息属性不会强制修改可视化工具。

相反，轻量级注释为 Haystack 提供了足够的信息，以便在现有数据中无缝地合并新类型和属性。

屏幕右侧是一个类似剪贴板的“暂存区”，用于存放任意内容；目前包含一封电子邮件（关于谷歌学术）和一个人（Hari Balakrishnan）。邮件的各个方面都会显示出来，包括正文、附件（目前已折叠）和推荐类别。显示的联系人信息包括收发邮件；其他信息，例如地址和电话号码，则会滚动到屏幕之外。

图 3.1 左侧面板底部显示“电子邮件”任务当前处于活动状态，并列出了用户在执行此任务时可能希望调用或访问的各种相关活动（撰写邮件）和项目（收件箱），以及用户在执行此任务时先前访问过的项目历史记录（在图 3.2 中展开）。点击这些任务即可调用它们并访问它们。

事实上，用户可以点击屏幕上的任何项目来浏览该项目的视图。单个消息、指定为发件人或收件人的各个个人，或任何“推荐类别”。



图3.2

在 Haystack 中调用“将此项目发送给某人”。收件箱集合显示在日历视图中。我们显示了三个不同的打开菜单 特定于任务的历史记录、搜索“Quan”的结果以及操作的上下文菜单 但实际使用时一次只会打开一个。

类似地,用户可以右键单击任何可见项目,以调用可应用于该对象的操作的上下文菜单。用户右键单击列出的某个消息发件人,即可打开一个菜单(及其子菜单),列出可针对该人调用的操作,例如向其发送电子邮件、发起聊天或将其添加到地址簿。这些操作不仅限于电子邮件应用程序的典型操作;而是适用于正在查看的对象的操作。其中一个操作“发送给 Jaime”是由用户创建的,因为他经常执行该操作。他保存了一个尚未完成的“将此发送给某人”操作;Haystack 自动识别出这个新操作适用,并将其添加到上下文菜单上。

最后,用户可以将一个项目拖到另一个项目上,以便以特定于项目的方式“连接”这两个项目 - 例如,将一个项目拖到另一个项目上

56 大卫·R·卡格

拖拽到集合上会将项目放入集合中,而将项目拖拽到对话框参数字段 (见图 3.2)会将该参数绑定到被拖拽的项目上。这三个操作 单击浏览、右键单击显示上下文菜单以及拖放 非常通用。它们可以随时以统一的方式在任何可见对象上调用。

左侧窗格中的“浏览顾问”会建议与集合中的项目类似的各种“项目” - 例如由卡格尔创建的项目或消息类型 - 以及“优化集合”的方式 - 例如,限制为正文包含某些词语或在特定时间发送的电子邮件。

动机:个性化信息管理

在开始详细讨论 Haystack 系统的设计之前,我们先尝试阐述一下我们试图解决的问题,以此来阐明其设计动机。我们观察到,当前的应用程序限制用户以他们可能不熟悉的方式管理信息,因此我们认为,优秀的信息管理工具必须赋予用户更多自主权,让他们能够自主决定存储哪些类型的信息,以及如何可视化和管理这些信息。

非个人应用

Haystack 项目的动机是观察到不同的人有独特的、往往是特殊的方式来管理他们的信息。

例如,我对我的书籍采用传统的主题分类法 (除了一些使用频率高的书籍)。我妻子会按作者的出生日期按时间顺序排列她的书籍。一位朋友会按颜色对她的书籍进行分类。这三种分类方法对拥有不同书籍的人来说效果很好,反映了他或她对相关书籍的记忆,但对其他许多人来说可能行不通。

这种物理组织方式的多样性在数字信息管理中被压缩成一种一致性。与我们的书架不同,当前的应用程序是千篇一律的:其他人已经决定了正确的“组织原则”,而我们无论喜欢与否都必须使用它们。应用程序的选择对于某些假设的普通用户来说可能是“最优的”,但似乎总是忽略了特定用户可能认为有用的某些属性。在一个典型的电子邮件程序中,我们

只能按照应用程序的预定义字段（如主题、发件人、日期和重要性）进行排序，而不能按照更特殊的特征（如发件人的年龄、“我读到这封邮件时在哪里”、收件人数量、发件人的重要性或“需要在某个日期前处理”）进行排序。

有时，所需信息实际上已记录在应用程序中，但没有提供合适的界面来使用这些信息。例如，尽管所有现代电子邮件客户端都会跟踪电子邮件的线程/回复结构，但微软的 Outlook Express 6 不允许用户根据给定线程中的邮件数量进行显示或排序。

所以，如果我记得某封电子邮件引发过很多讨论，我很难利用这些信息找到这封邮件。Mozilla Thunderbird 却有这个功能，因为一位应用程序开发者认为值得整合。虽然有这些信息可以解答这个问题，但这两款工具都不允许我根据发件人上次回复的时间长短来显示或排序邮件。

在其他情况下，问题甚至更早出现，用户关心的信息甚至无法记录在应用程序中。电子邮件程序没有电子邮件的“截止日期”字段。虽然 MP3 音乐文件带有用于记录各种元数据的 ID3 标签，但却没有字段或表示形式供用户记录与特定音乐片段相关的舞步。虽然照片带有元数据字段，但它们都无法保存指向用于向我发送（并告知我）照片的电子邮件消息的指针。微软和 Thunderbird 的通讯录工具都不允许我添加特定联系人的照片。作为一名学者，我甚至可能希望同事的通讯录条目包含我读过的他们的一些论文的描述和链接。但典型的通讯录中没有“出版物”字段，因为应用程序开发人员认为它不值得添加。

很多情况下，人们会尝试使用通用的“注释”或“评论”字段作为最后的手段，但这会剥夺应用程序以有意义的方式使用这些字段的所有机会。虽然我可以记下某个联系人照片的文件名，但在查看他们的通讯录条目时却看不到那张照片 我必须手动查找并打开指定的文件。即使我在通讯录的“自定义”字段中记录了同事的发布内容，我也无法使用通讯录来选择那些发布过文章的同事。

58 大卫·R·卡格

在给定的会议中。如果我在电子邮件的注释字段中以文本形式记录“截止日期”，那么按该字段排序时，很可能无法获得我想要的截止日期结果，因为此类字段通常按字典顺序排序。为了解决这个问题，我必须发明并记住一种特定的“反向”日期表示法（年-月-日，并注意始终使用两位数字表示日期和月份）。

统一

应用程序不记录特定类型信息的一个常见原因是，该信息被认为是“其他应用程序的工作”。开发人员可能会争辩说，记录出版物似乎是书目工具的工作，而不是地址簿的工作，或者坚持认为在特定日期前回复电子邮件应该由待办事项列表或日历而不是电子邮件程序来处理。在所有这些推卸责任之后，我们可能会承认用户需要的信息存储在某个应用程序中，但不可能将所有信息集中到一个地方并一起使用，甚至不可能轻松地从一个应用程序的一个部分导航到另一个应用程序的另一个部分。如果我的日历显示我必须处理某封电子邮件，我必须先在电子邮件程序中找到该邮件才能处理。这种数据碎片化还可能导致用户在多个应用程序中记录重复的信息，当用户只更改了该信息的一个副本而不是所有副本时，就会导致不一致。如果收到一封无需回复的后续邮件，我可能会忘记将其从待办事项列表中删除（尤其是从邮件信息到相应的待办事项需要更多工作）。我们最近关于数据统一的文章（Karger 和 Jones, 2006）详细讨论了这个问题。

应用程序越多，困难就越多。许多人会同时使用电子邮件阅读客户端、音乐管理工具、相册、日历和通讯录。电子邮件客户端和通讯录可能在某种程度上相互关联，但其他应用程序各自独立管理数据。现在，让我们来想象一下一位关注音乐行业的娱乐记者的困境。她与音乐家互发电子邮件，安排采访，参加他们演唱特定歌曲的预定音乐会，并撰写评论和采访。这样的用户似乎很可能会想要

- 将有关某次采访的电子邮件与她撰写的采访文章联系起来

正在写作；

- 将音乐家与他们的消息联系起来（如图 3.1 剪贴板中的“人物”对象所示）、他们演奏的音乐会、他们演唱的歌曲以及他们拍摄的照片；

- 为音乐家的表演照片加上“标题”，并配上歌曲

在照片中表演；

- 根据发行日期将歌曲或专辑放在日历中；

在。

目前，虽然每个感兴趣的项目都由某个应用程序管理，但没有任何应用程序能够感知其他项目类型或应用程序。应用程序的内部模型缺乏足够的表达能力，无法引用其他项目类型（除非通过英文“评论”），并且它们的用户界面无法显示用户感兴趣的链接，也无法将相关对象整合成一个统一的、连贯的表示。系统无法识别歌曲ID3标签中记录的艺术家与通讯录中提到的艺术家是否相同。记者所能期望的最好结果就是同时打开所有相关的应用程序来获取这些数据，但此时，她真正关心的信息就会淹没在与特定任务不太相关的杂乱信息中。记者必须努力将针对特定任务的应用程序“重新利用”到新任务中。她可能更喜欢专门针对她特定任务的应用程序。

在一项关于用户桌面环境的研究中，Ravasio、Guttormsen-Schär 和 Krueger (2004)发现，用户自身也意识到了这个问题：“三位用户认为，系统地将文件、电子邮件和书签分开存储对他们的工作造成了不便。在他们看来，所有数据都构成一个单一的信息体，现有的分离只会使数据备份、从旧电脑切换到新电脑，甚至搜索特定信息等操作变得复杂。他们还指出，这种分离会导致各个存储位置出现不必要的冗余。”

不同应用程序之间相关信息的碎片化阻碍了一种重要的信息搜索策略。在最近的一项研究中 (Teevan 等人, 2004)，我们强调了定向越野在……中发挥的重要作用。

60 大卫·R·卡格

信息探索。用户进行定向越野时,通常不会精确描述自己的信息需求,并期望计算机直接将他们“传送”到相关对象,而是从信息空间中“靠近”该对象的某个地方开始,然后通过一系列本地步骤导航到该对象。当相关信息分散在多个互不“关联”的应用程序中时,用户就无法遵循自然的定向越野路径。

新的数据类型

除了向现有类型添加信息外,用户可能还会发现需要全新的信息类型。有时,这可能是对现有应用程序的丰富。例如,日历程序通常允许记录事件的地点。但这条记录仅仅是一个文本字符串。

用户可能需要更丰富的表示形式,例如包含前往该地点的路线、地图、附近的酒店列表或指向该地点网页的链接(如果存在)。有时,用户可能会从头创建一个新类型。

我们的音乐记者可能意识到她想要记录和检索音乐会的信息 音乐会的举办地点、其他观众、演奏者、演出质量如何、有多少人被捕等等。她应该把这些信息放在哪里?

现有应用程序在整合新类型信息方面的表现甚至比扩展现有类型更差。它们没有提供用于显示新类型的控件,没有用于对其进行操作的操作,也没有将其附加到应用程序中其他信息对象的插槽。面对这种情况,用户通常会求助于电子表格,创建一个表格记录,其中每一行都是一个“项目”,每一列对应于新信息的某些属性。这种“穷人的数据库”确实提供了一些处理信息的能力:用户可以对某一列进行排序,以查找具有某个属性的给定值的所有信息。但其呈现方式必然是通用的,没有任何应用程序提供的、用于促进特定领域工作的特定任务或数据的呈现、菜单和操作。

个性化

应用程序开发人员可以轻松解决本节中提到的每个具体问题。应用程序开发人员可以在通讯录中添加联系人照片的位置,或者编写一个通讯录函数

它可以查询电子邮件数据库并列出“来自此人的消息”，或者将照片与电子邮件消息实时链接。甚至可以为音乐记者构建一个专门的数据管理应用程序，并添加新的“音乐会”数据类型。

但应用程序的用户数量远多于应用程序开发者，每个人对应用程序的信息和呈现方式都有不同的需求。即使应用程序开发者能够满足所有这些个性化需求，最终的应用程序也会充斥着大多数人并不关心的数据和呈现方式，显得杂乱无章。

Kaptelinin 和 Boardman（本书）认为，试图在一个应用程序中服务所有用户，会使应用程序臃肿不堪，以至于无法胜任其最初预期的任务。而且，应用程序所需的属性集合肯定是一个不断变化的目标：无论开发人员在应用程序中容纳多少种信息，总有用户会想要添加新的属性。

解决这个问题的一个方法，或许也是唯一的方法，是让最终用户能够自定义其应用程序管理的数据及其呈现方式。这种自定义功能已经在一定程度上实现了：大多数电子邮件程序、音乐管理器以及文件/目录浏览器都允许用户选择在其集合的表格视图中显示哪些列（属性）。但这种自定义功能主要局限于表格/聚合视图；用户对单个信息对象的呈现方式的控制较少。在添加和使用新的数据类型和属性时，提供的支持要少得多。通常，无处不在的通用文本“注释”字段是保存用户新信息类型的唯一位置。

来自网络的经验教训

万维网似乎解决了我们之前概述的当今应用程序的许多问题。在网页上，用户可以记录任何类型对象的任何信息。要添加新的信息属性，用户只需将其输入到现有网页中即可，无需更改任何应用程序。他们可以链接到其他网页，而不管其他网页描述的是哪种类型的对象。无需“按应用程序划分”。因此，网络非常适合定向越野：搜索引擎通常会将用户引导至正确的位置，然后通过一系列有用的链接找到所需的信息。

62 大卫·R·卡格

那么,用户是否应该放弃应用程序,将所有数据转移到网络上?假设用户可以为每封电子邮件、每个目录、每个音乐文件、每个日历约会、通讯录中的每个人等等创建一个单独的网页。通过编辑这些网页,用户可以指定信息对象之间的任意关系。将这些网页输入到像谷歌这样的工具中,可以赋予用户强大的搜索功能;将这些网页与用户创建的链接提供的定向越野机会相结合,无疑会增强用户查找信息的能力。

当然,这种方法永远行不通:它需要用户付出太多努力。目前尚不清楚,更好的检索方式带来的回报是否值得组织投入如此多的努力。考虑到网络内容创建工具的现状,这种投入将是巨大的。消费网络内容的人远远多于创建网络内容的人;他们将其视为只读资源。“只读”应该从字面上理解 用户通常用肉眼浏览网页以提取信息,而不是将其输入任何复杂的自动化工具。相反,当人们处理自己的信息时,他们会以各种方式对其进行操作和修改。这种操作和修改需要简单易用,因此每个应用程序都应该提供自己专门的接口和操作来操作其管理的数据。我们需要让用户管理自己的信息,而不是强迫他们成为网站开发人员。

操作。

类型和结构

应用程序提供结构化数据模型,其中包含精心定义的类型和关系。这些模型使用户能够以比在网络上更自然的方式操作、导航和显示信息。人们使用的许多信息对象、它们的属性以及它们之间的关系都具有有意义的类型。人构成了一种对象类型,它往往具有姓名、地址和电话号码等属性。

邮件消息构成一种对象,通常具有发件人和收件人等属性,而这两个属性通常都是人员类型的实例。虽然发件人和收件人可能都是人,但这两个属性代表了与消息相关的不同角色,不应混淆。

在网络上,通常由读者自行推断对象的类型和关系。所查看对象的类型可能由对象的标题、其展现的属性或其在上下文中的位置暗示。属性的类型通常在链接的前面或锚文本中以英文标明 例如,互联网电影数据库 (Internet Movie Database) 中某部电影的页面标题为“导演”,

“编剧名单”和“类型”,引入与给定电影有关系的对象链接。

此类视觉或英语提示的缺点在于,它们只有人类才能理解。虽然这对于基本的定向输入来说可能没问题,但它会阻碍我们的应用程序利用隐式类型来改进信息的呈现和检索。根据上下文,用户可能并不想查看某个对象的所有信息。例如,为了支持电子邮件浏览,邮件工具通常会提供一个精简的邮件(列表)视图,仅显示发件人、日期和主题。

这些信息很难从任意人为格式化的网页中提取出来,但从结构化数据模型中却很容易获取。从这个意义上讲,网页很像许多应用程序中提供的灵活评论字段 能够记录所有内容,但却无法被支持应用程序有效地利用。

以结构化形式存储信息也增强了搜索能力。

个别应用程序通常以这种方式利用结构 例如,音乐播放应用程序允许用户按作曲家、专辑或表演者排列音乐。更普遍地说,Yee 等人 (2003) 认为,分面元数据浏览 (用户可以根据所选属性的值选择组织语料库) 是一种重要的信息检索工具。

虽然应用程序一直以来都对其数据施加这种结构,但网络也正朝着这个方向发展。为了支持导航和搜索,像IMD (互联网电影数据库)这样的网站以结构化形式存储信息,然后将其渲染成人类可读的形式。

通过使用模板。结构化存储意味着网站可以提供“站内搜索”工具,以比纯全文搜索更丰富的方式利用现有元数据。例如,Epicurious.com 允许用户搜索和浏览各种菜谱成分或烹饪方法 这是一种分面元数据搜索 (Yee 等人,2003)。

64 大卫·R·卡格

然而,网站向数据专用应用程序转型的举动,意味着它们会遭遇与数据专用应用程序相同的一些问题。用户经常发现,他们认为重要的属性并未在导航中显示,甚至根本无法使用。当数据用户跨多个网站访问时,没有哪个网站能够提供信息的聚合视图,或支持跨网站导航。

那么,我们需要一种方法,将网络的信息灵活性与应用程序结构化数据模型支持的丰富交互融合在一起。

用户界面后果

我们已经阐明了让最终用户自主选择信息模型的动机 哪些信息对象、属性和链接对他们重要。接受这样的立场会给用户界面带来一些有趣的挑战。

用户界面最简单的任务可能就是显示信息。

使用传统的应用程序数据模型,这相对简单。

开发人员会针对每种数据类型和显示上下文,考虑需要向用户呈现该对象的哪些功能。通过考虑每个功能的预期类型,开发人员可以确定一些有意义的方式(例如,字符串、数字、颜色或图标)来显示每个功能,并确定所有单个功能呈现的有效聚合方式。

但在我们的个性化数据模型中,对于将要显示的数据,我们几乎无法进行任何假设。用户可能会开始记录或连接到新的信息类型,而这些信息类型目前尚未开发相应的呈现机制。即使是预期的信息类型,用户也可能已经建立了新的关系,或者违反了旧关系的模式,因此开发人员对需要显示内容的假设变得无效。

因此,我们灵活的数据模型的用户界面需要

同样灵活,根据数据模型的实际内容(而不是计划内容)调整其呈现方式。

网络浏览器似乎是朝着这个方向迈出的一大步 它不对所呈现信息的结构做任何假设,而只是简单地渲染格式化的HTML。但这些HTML究竟从何而来?一种可能性是手工生成的 但上文我们已经论证,用HTML记录所有用户信息是不切实际的。

另一种可能性是通过将某些模板引擎应用于底层数据模型来生成 HTML,就像许多面向数据的网站所做的那样。但这只会将问题进一步推向更深的层次:当前的模板需要对数据结构做出同样严格的假设,而这限制了应用程序的运行。

同样成问题的是网络浏览器的“只读”特性。我们需要一个用户界面,让用户能够像在普通应用程序中一样轻松地操作信息。

概括

在本节中,我们概述了我们构建一个能够适应任何个人用户需求的半结构化数据模型的动机,以及一个能够适应数据模型的用户界面的动机,该用户界面能够融入新的信息属性、新的信息关联以及新的信息类型。在本章的剩余部分,我们将描述在 Haystack 系统中为实现这些目标所做的尝试。在解决了核心的数据建模和用户界面问题之后,我们将讨论该系统提供的一些机遇。

语义网络 Haystack数据模型

上文我们讨论了允许用户操作任意信息对象,并允许他们记录和使用这些对象的任意新属性的重要性。在考虑支持这些活动的界面之前,我们需要开发一个足够灵活的数据模型来保存这些信息。

支持灵活信息的有效通用表示是语义网络:一种图,其中节点表示要管理的信息对象,边标有属性名称,以表示我们想要记录的关系。边只能直接表示二元关系,而不能表示两个以上实体之间的关系。然而,我们遇到的大多数关系都是二元的,而更高元数的关系通常可以通过具体化关系来表示(创建一个新的信息对象来表示特定的关系元组,并使用从该元组到参与该关系的实体的二元连接),因此这种二元限制并不构成负担。

66 大卫·R·卡格

除了我们理解的关系之外,语义网络边还可以通过在对象和给定属性值之间创建以属性名称为标签的链接来表示我们理解的对象“属性”、“特性”或“特征”。这凸显了这样一个事实:从形式化角度来看,这些概念等同于关系。虽然用户可能保持直观的区分(例如,属性是对象固有的,而关系将对象与其他不同对象连接起来),但我们将避免在数据模型中做出这种区分,而是将其融入用户界面,旨在以符合用户直觉的方式呈现数据。

资源描述框架 (RDF)

虽然 Haystack 的原始版本 (Adar、Karger 和 Stein 1999) 实现了其自己的语义网络表示,但我们后来采用了万维网联盟 (Manola 和 Miller 2003) 提出的标准资源描述框架 (RDF)。

RDF 满足了我们的表征目标。它使用统一资源标识符 (URI) 来引用任意信息对象。这些 URI 非常类似于 URL,但不必指向存储在特定 Web 服务器上的信息(当然也不必通过 HTTP 进行解析)。在 RDF 中,信息对象被称为资源,关系被称为属性。

两个资源之间关于给定属性的特定断言被称为语句。语句链接的两个资源被称为主体和客体,而所选的属性被称为谓词。属性也通过 URI 命名,这使我们能够对属性进行陈述。例如为其指定一个人类可读的名称,或者断言每个资源应该只有一个该属性的值。语句也可以具体化并赋予 URI,以便记录例如谁断言了给定的语句。

RDF 还支持具有继承性的类型系统。Type 属性被保留用于指定给定资源属于给定类型。某些 Class 类型的资源表示类型;这些是 Type 的典型对象。

语句。有一个(最通用的)类称为 Object;所有资源都是此类的实例。属性被断言为 Property 类型。

RDF 允许用户定义一系列适用于特定用途的类型和属性。这些属性可以全部定义在一个 (RDF) 文件中;如果为该 RDF 文件指定了一个 URL,那么各个类和属性就可以被定义在一个文件中。

其中可以使用标签语法(<http://url/#label>) 来引用。根 URL 指的是定义的类和属性的命名空间。

例如,都柏林核心定义了诸如dc:document和dc: 之类的类型

人员和属性,例如dc:author和dc:title (这里dc:是都柏林核心命名空间的简写,而每个后缀标记命名空间中的特定类或属性) 。

基于 RDF, RDF 模式语言 (RDFS) 和 Web 本体语言 (OWL) (McGuinness 和 van Harmelen 2003) 可用于定义类和属性的模式。RDFS 和 OWL 是属性和类 (定义在 RDFS 和 OWL 命名空间中) 的集合,可用于断言典型的模式规则。例如,RDFS 和 OWL 可用于断言dc:author语句的主体必须是dc:document ,客体必须是 dc:entity,或者dc:document

最多只有一个dc:date。我们不强制 Haystack 使用模式;尽管如此,此类模式可用于建立适当的信息视图,或引导 (但不强制) 用户填写值。

为什么选择 RDF?

有人可能会质疑我们选择 RDF 而不是 XML 或更传统的每个属性一个表的关系数据库表示形式。从很多方面来看,这个问题并不重要。这三种表示形式都具有相同的表达能力。确实,与传统数据库不同,RDF可以在没有任何模式的情况下使用。但是,如果我们愿意,可以使用 RDF 和 OWL 将模式强加于 RDF 模型。RDF 具有 XML 的标准表示形式 (RDF-XML),也可以存储在传统数据库中 (使用一个三元组表或每个命名属性一个二进制表) 。相反,XML 或数据库元组可以用 RDF 表示。当然,表示形式的选择可能会对系统的性能产生巨大影响,因为它会回答各种查询。但是,最终用户可能既不知道也不关心用户界面背后隐藏着哪种表示形式。

尽管如此,RDF 的一些特性还是促使我们选择了它。下面讨论的缺乏 (强制)模式是一个很有吸引力的特性。所有信息对象都使用 URI (统一资源标识符) ,这提供了一种与位置无关的统一命名方案。同样吸引人的是,RDF 将所有信息对象置于一个公平的竞争环境中:每个对象都由 URI 命名,并且可以成为任意对象的主体或客体。

68 大卫·R·卡格

断言。这与 XML 对信息对象的层次化表示形成了（正面的）对比，XML 的根对象是“特殊的”，而相关对象则嵌套在特定的根对象内部。RDF 更符合我们的信念，即信息设计者无法预测哪些信息对象最受特定用户关注。Codd (1970) 的开创性论文中也提出了类似的观点，他认为，信息层次化表示如果本质上不是层次化的，会引入不良的数据依赖性，从而可能使数据库用户感到困惑。对于关系数据库，也可以提出类似的论点。定义包含许多列的数据库表表明应该将这些字段作为整体来考虑，但不同的用户可能只对其中一些字段感兴趣。我们可以建议将数据投影到列的子集上，但 RDF 从一开始就放弃了这样的想法：每个列本身可能都很有趣，应该有自己的表，从而回避了如何投影的整个问题。

我们采用 RDF 的另一个动机是其结构与万维网 (World Wide Web) 的相似性。网络的强大之处在于其链接，用户可以从页面导航到相关页面。同样，语义网络强调对象之间的联系，而不是整体上的关系。这一点很重要，原因有二。首先，它捕捉到了“局部性”的概念。当用户处理特定信息对象时，他们通常希望访问语义网络中“相邻的”相关信息对象。其次，鉴于定向运动在个人信息搜索行为中扮演的重要角色 (Teevan et al. 2004)，强调联系是恰当的。用户通常更喜欢从熟悉的位置或模糊的搜索开始，并通过一系列联想步骤“锁定”所需信息，而不是精心构建一个精确定义所需信息目标的查询。在 RDF 中，连接主语和宾语的语句形成自然的联想链接，用户可以沿着这些链接从主语定向到宾语。如果用户希望编写复杂的查询，例如“连接”和“投影”等可以用 SQL 等数据库语言简洁表达的操作，那么数据库视角可能更合适。然而，典型的用户不具备使用此类数据库查询语言的能力，因此公开这些操作的价值有限。

图 3.1 中显示的每个项目的各种属性通常只是通过某些谓词与显示的内容关联的其他信息对象

对象。Haystack 的用户界面允许用户点击任何信息对象进行浏览，从而支持定向导航。当我们讨论用户界面时，将会清楚地看到，RDF 的“谓词”这一单一概念以多种方式提供给最终用户。有时作为与另一个对象的关系，有时作为当前对象的属性。给定对象的“属性”或“特性”以及对象对之间的“关系”都由数据模型中的谓词表示。

半结构化模型

除了命名关系之外，结构化数据模型通常具有模式

表达关于不同信息对象和类型如何关联的知识。例如，我们可以声明一首交响曲的作曲家必然是个人，或者任何个人在给定时间内最多只能与一位其他人结婚。这种示意性声明非常有用。它们可以防止用户在记录信息时出错，例如，当用户在输入有关新交响曲的信息时交换作曲家和标题时，可以及时发现错误。它们可以简化信息的呈现，让用户推断出只需一行即可在通讯录条目中显示配偶。

但这些保护措施同时也是社群意识施加的限制，可能违背个人意愿。设想一下，一个对计算机音乐感兴趣的人：如果上述方案得到实施，她试图以交响乐作曲家的身份进入某个计算机程序的尝试将被阻止。同样，一位研究一夫多妻制社会的研究人员可能会发现自己无法查看记录中关于个人及其配偶的关键信息。

因此，尽管图式可能具有重要的指导价值，但我们反对强制执行。用户必须始终有办法修改图式，或者在得到合理警告后，认为违反图式是记录相关信息的最佳方式。这种观点是让用户记录任何她认为重要的信息这一理念的自然延伸。如果我们面临违反图式或拒绝让用户记录她关心的信息的选择，我们会选择前者。开发人员可能认为电子邮件的发件人不太可能是动物，因此可能会将发件人图式化为一个人，而用户可能会做出不同的决定。尽管文档通常具有

70 大卫·R·卡格

作者,用户可能并不关心记录它们。语义网络对图式的依赖程度低于数据库:每个命名链接可以存在,也可以不独立于任何全局图式。

像这样一种可以表示数据库类型结构但不强制执行的结构的表示形式称为半结构化数据模型。虽然我们已经论证过,半结构化模型对于支持用户记录信息至关重要,但在实际呈现或操作信息时,它会带来一些问题。但这些问题是在用户界面层面的问题,我们应该在那里解决,而不是试图通过限制数据模型来解决它们。正如我们将在那里看到的,图式可以在半结构化信息管理中发挥重要作用;不同之处在于,图式是可选的和建议性的,而不是强制执行的。因此,半结构化信息最好被视为“图式可选”,而不是“图式无关”。

导入数据

尽管 RDF 颇具吸引力,但目前大多数数据并非以这种形式存在。Haystack 通过将一系列提取器应用于传统格式的数据来生成 RDF 数据。目前,我们可以整合目录层次结构、各种格式的文档、音乐和 ID3 标签、电子邮件 (通过 IMAP 或 POP3 接口)、Bibtex 文件、LDAP 数据、照片、RSS 提要和即时消息。每种数据都由一个相应的解析器进行整合,当给定类型的信息被吸收到系统中时,解析器就会触发该解析器。

半结构化数据的另一个重要来源是网络本身。

许多网站使用模板引擎来生成存储在后端数据库中信息的 HTML 表示。我们研究了机器

学习技术可以自动将网页中的此类信息提取回 RDF 的结构化形式 (Hogue 2004; Hogue 和 Karger 2005)。在我们的方法中,用户通过突出显示单个项目并标记其各个部分来“演示”提取过程;然后,系统尝试归纳出表示页面上该对象的 HTML 标签和数据元素的 (树形) 结构。如果成功,它可以在以后的页面上识别该结构并自动执行相同的提取。当然,Haystack 并不关心其 RDF 的位置。

因此可以轻松结合其他提取方法（Muslea、Minton 和 Knoblock 1999）。

查看信息

鉴于数据模型的表达能力，下一个问题是如何将其呈现给用户，以便他们能够有效地查看和操作存储的信息。简单地修改传统应用程序以使其运行在统一数据模型之上，可以带来一些有限的好处。例如，减少在多个应用程序中重复的信息量，从而减少这些重复信息之间的不一致。但这会让用户像以前一样受到开发人员对在不同情境下应该呈现什么以及如何呈现信息的理解的限制。相反，我们必须让用户界面能够根据用户的偏好和它所需要显示的数据进行简单的演进。我们通过递归渲染架构来实现这一目标，在该架构中，每个对象本质上都被要求渲染自身，并递归地向与其相关的其他对象发出相同的请求（Huynh、Karger 和 Quan 2002；Quan 和 Karger 2003）。

视图

大多数基础信息管理应用程序都会在屏幕上以层级结构显示信息。为了在屏幕的某个区域中显示特定对象，它们会将该对象的区域细分为（通常是矩形的）子区域，并使用这些子区域来显示给定对象的各种属性以及与该对象相关的其他对象。因此，典型的电子邮件应用程序会通过创建一个显示发件人的区域、另一个显示主题的区域、另一个显示正文的区域等等来呈现电子邮件消息。消息本身可能位于子区域中，作为更大显示的一部分，例如一组消息，使用不同的列来显示每条消息（与发件人、主题和日期的关系）。日历视图每天都会显示一个约会列表，而通讯录则具有标准格式，可以通过以某种格式整齐的布局列出姓名、地址、电话号码和备注等属性来显示个人。地址本身可能

72 大卫·R·卡格

是一个复杂的对象,具有需要布局的不同子属性,例如街道、城市和国家。

当应用程序针对特定领域时,它们可以对其子区域中显示的内容做出大量假设。电子邮件地址的发件人是人;他将拥有可在显示屏的发件人区域中显示的姓名和地址。地址簿条目将描述拥有地址的人。在 Haystack 中,我们不想做出这样的假设:我们的收件箱包含 RSS 故事,其发件人类型可能与电子邮件消息不同。但我们仍然可以应用递归显示原则。我们可以通过 (i) 决定需要显示 X 的哪些属性以及与其他对象的关系,(ii) 请求递归渲染 X 所需对象的视图,以及 (iii) 以指示 X 与它们的关系的方式布局这些递归渲染的视图。举一个具体的例子,在渲染邮件消息时,我们可能认为渲染发件人很重要;我们通过递归请求发件人的视图,然后将发件人的视图布局到邮件消息视图的某个位置来实现这一点。在渲染发件人时,递归调用可能会递归请求渲染发件人的地址,以便将其合并到发件人的视图中。

这种递归方法的主要优点是,根视图只需要了解它负责显示的根对象,而无需了解最终显示的任何相关对象。将 RSS 源合并到收件箱中不需要彻底重写邮件应用程序;只需为每条 RSS 消息定义一个视图即可。定义该视图后,显示收件箱的集合视图便会在需要时调用它。

查看处方

正式来说,视图由视图处方 (view prescription) 定义,而视图处方本身就是模型中的数据。视图处方是一组 RDF 语句,用于描述如何划分显示区域,以及每个子区域应显示哪些常量 (例如标签) 和相关对象。它还声明某些标准图形控件 (例如滚动条和文本框) 应环绕显示还是嵌入显示。

当视图处方被调用时,它需要一些上下文才能正确渲染。最明显的是,我们需要知道

渲染对象应占用的空间。为了获得一致的呈现效果,通常需要从父视图向下传递其他状态,例如当前颜色和字体大小。这是通过动态范围控制来实现的。视图可以访问由祖先视图在递归渲染过程中设置的变量环境。它可以检查这些变量,也可以修改它们以适应其子视图。

Haystack 接口层的关键任务是确定应使用哪种视图方案来渲染信息对象。目前,我们采用一种非常简单的方法:根据要显示的对象类型及其显示区域的大小进行选择。每个视图方案都使用更多 RDF 语句指定其适用的类型和大小;当渲染请求被委托时,Haystack 会使用 RDF 查询来确定要应用的合适方案。类型和大小是影响方案选择的最明显属性;一个值得进一步研究的问题是扩展用于讨论哪些视图在哪些情况下适用的词汇表。

在按类型匹配时,Haystack 会在信息对象上使用类型层次结构,并选择与最具体的可能类型最匹配的视图。类型层次结构使我们能够定义相对通用的视图,从而提高用户界面的一致性并减少所需的不同处方数量。例如,RSS 帖子、电子邮件和即时消息都被视为通用“消息”类型的子类型,我们可以预期该消息包含发件人、主题和正文(Quan, Bakshi, and Karger 2003)。因此,一个视图处方适用于所有这三种类型。为了确保所有信息对象都能以某种方式显示,Haystack 包含始终适用的“最后手段”视图。例如,

“小”最后手段视图仅显示信息对象的标题,如果不可用,则显示其 URI;而“大”视图则显示对象所有属性和值的表格列表(以递归方式呈现)。

有人可能会说,我们的视图架构非常贫乏,只提供了基于对象类型和大小的矩形层级分解和委托。虽然我们同意这是一种贫乏的架构,但我们断言,它捕捉到了当前(同样贫乏的)信息管理应用程序的大部分展示能力,并以图 3.1(可以看作一个典型的邮件客户端)作为证据。虽然与现有

74 大卫·R·卡格

应用程序中,我们的委托架构有助于合并新的数据类型和跨域信息链接。

与现有应用程序相比,一个关键的改进是视图可以在任何地方调用。图 3.1 的右侧面板展示了一个类似“剪贴板”的东西,任何信息对象都可以拖拽到其中显示。因此,无需启动整个通讯录应用程序,即可查看“Hari Balakrishnan”这个人的信息;同样,即使我们选择离开收件箱并停止“处理电子邮件”,关于“谷歌学术”的电子邮件仍然可见。这种无需依赖外部应用程序即可获取数据的想法与 Tan、Meyers 和 Czerwinski (2004) 提出的 WinCuts 技术相呼应。

我们的视图架构还能直接为同一信息对象提供多个视图,让用户根据其任务选择合适的视图。图 3.1 的中心窗格提供了一个“更改视图”下拉菜单。用户可以从此菜单中选择任何已标注为适合所显示对象的视图。

同样重要的是要认识到,在视图递归的基础上,复杂数据对象的呈现可以委托给专用的小部件。Haystack 的视图规范不足以描述散点图的呈现以及用户在查看散点图时可能想要调用的交互操作,但规范可以明确指定某个“散点图小部件”是需要显示散点图时应该调用的合适视图。这种方法甚至可以允许将整个应用程序嵌入到 Haystack 中,只要它们能够被告知要关注哪个数据对象。

镜头

虽然显示给定对象的属性列表可能就足够了,但这些属性通常会自然分组,以表征所呈现信息的某些“方面”。Haystack 中的这种分组是通过定义镜头来实现的。镜头为信息呈现增加了另一层。与视图类似,数据模型中描述的镜头适用于特定类型的对象。图 3.1 右侧窗格中的人员和邮件消息就是使用镜头视图显示的。镜头视图适用于所有对象类型。它只是识别给定类型的所有适用镜头,并显示每个镜头。每个镜头都有一个标题来描述它所显示的方面,例如“来自此人的消息”。

与递归渲染的视图不同,这些镜头是“具体化的”,用户可以直观地定位每个镜头,选择展开或折叠它(通过镜头名称旁边的小加号/减号)。这种选择是有状态的:用户对显示哪些镜头的选择每次都会被记住。

镜头视图适用于该类型的信息对象。这提供了一定程度的视图定制。此外,我们的许多镜头都是简单的“属性集镜头”它们由一个列表描述,该列表列出了它们将显示的对象属性,并且这些属性仅显示在列表中。用户可以通过在列表中添加或删除属性来轻松修改这些镜头。因此,如果用户选择在其数据模型中定义一个全新的属性,他可以直接调整用户界面来呈现该属性。

镜头也可以与上下文相关。例如,某些镜头可能仅在执行特定任务时才会出现。Google Scholars 电子邮件消息中显示的“推荐类别”镜头仅在用户执行“组织信息”任务时才会出现。

“帮助”镜头可以聚合有关任何物体的有用信息,但只有当用户真正寻求帮助时才可见。

用户可以通过操作透镜进一步定制信息视图。例如,收件箱视图中的第四列“推荐类别”是通过将“推荐类别”透镜从谷歌学术视图拖放到收件箱集合的标题上创建的。如果用户希望根据标题快速浏览和整理电子邮件,而无需查看每封邮件的详细信息,这将是一个非常实用的操作。此表格集合视图在每一行中布局一个项目,并在每一列中应用一个透镜,以确定在该列中显示有关给定行中对象的哪些信息。任何透镜都可以放置在此集合视图的某一列中,从而允许用户构建一种“信息电子表格”,以显示用户关注的集合对象的任何方面。

收藏

集合是人们使用的最常见的数据类型之一。

几乎每个应用程序都提供了用于管理其原始元素集合的工具:文件的目录/文件夹管理器、网页浏览器的书签管理器、电子邮件的邮件文件夹等等。通常,这些集合仅限于特定应用程序“拥有”的对象类型。

76 大卫·R·卡格

在 Haystack 的统一数据模型下,可以聚合与给定任务相关的任意信息集合。现有桌面系统中与之最接近的类似物或许是文件管理器。目录能够保存任意类型的文件,这意味着用户可以按任务而不是文件类型对文件进行分组。当然,其局限性在于这种管理只能应用于文件级别。因此,那些封装在应用程序数据文件中的项目(例如地址簿中的个人联系人或邮件文件夹中的单封邮件)无法组织成异构集合。Haystack 通过为所有感兴趣的对象提供统一的命名方案,将异构集合的优势扩展到任意对象。我们已经在图 3.1 中指出,非电子邮件对象(例如 RSS 故事和人物)可以无缝地放入收件箱中。

多个视图的可用性对于集合来说尤其重要,集合可能是 Haystack 中的核心非原始数据类型。

由于集合用途广泛,因此存在多种视图。图 3.1 展示了集合的标准行布局,此外还有日历视图(其中集合中的每个项目都按日期显示,此视图应用于图 3.2 中的收件箱)、图形视图(其中的对象显示为小图块,连接图块的箭头用于指示它们之间特定的选择关系)以及“最后手段”视图,显示集合的所有属性及其值。每种视图可能适用于不同的时间。标准视图适用于传统的电子邮件阅读。图形视图可用于检查长对话的线程结构。用户可以应用日历视图,根据截止日期而不是到达时间来重新排列电子邮件。

另一个集合视图是菜单。当集合充当菜单时,单击鼠标左键会下拉该集合的“菜单视图”,以便快速选择集合中的某个成员。以这种方式实现菜单使用户能够自定义界面:通过在菜单中添加或删除操作集合,用户可以修改菜单。用户还可以类似地自定义左侧窗格中固定到位的任务菜单(例如图 3.1 中显示的电子邮件任务菜单),以便提供新的特定于任务的操作和项目。

传统上,下拉菜单用于呈现操作集合。虽然 Haystack 确实将操作放在菜单中(参见

(如下图所示),任何对象都可以包含在以这种方式呈现的集合中。因此,轻量级访问和存放集合的概念与访问操作的问题分离了。例如,如图 3.2 所示,系统顶部搜索框中的搜索结果以下拉菜单的形式呈现(但也可以导航至下拉菜单进行更仔细的检查和操作)。

一个特别值得一提的集合视图是位于屏幕右下角的“复选框视图”。这形成了一种略微颠倒的集合视图,因为它显示了 Google 学术搜索电子邮件所属的给定类别集合中的哪个集合。勾选或取消勾选复选框将在给定集合中添加或移除该项目。

当然,集合本身是动态的。可以通过将项目拖放到集合名称上来添加项目,也可以通过左键单击来浏览集合。但在之前的一项研究中(Quan、Bakshi 和 Karger,2003),我们证明了将集合以可查看的“类别”形式呈现给用户,会对其使用方式产生巨大影响。

许多电子邮件用户不愿将电子邮件分类到文件夹中,担心分类后的电子邮件会从收件箱中丢失或被遗忘。许多邮件工具允许用户将电子邮件复制到文件夹中,并在收件箱中保留一份副本,但用户显然觉得这项操作过于繁琐。尤其是,一旦复制了两份,用户可能难以保持同步。一份副本上的注释不会出现在另一份副本上。另一方面,复选框更像是一种注释邮件的方式,而不是一种收件方式,因此鼓励用户进行多重分类。在我们的研究中,选择使用复选框进行分类的用户充分利用了这一功能,并发现它提高了他们日后检索信息的能力。当然,在底层数据模型中,复选框代表着与所有其他可浏览的集合一样的集合(事实上,收件箱本身就是可勾选的类别之一)。

创建新视图

我们持续探索让用户自定义信息呈现方式的方法。我们创建了一个“视图构建器”工具,允许用户根据给定的信息类型设计新的视图(Bakshi 2004)。用户可以通过菜单和拖动来指定屏幕空间的特定布局,并指定在每个区域中显示所查看对象的哪些属性,以及应该使用哪种视图来显示它们。

78 大卫·R·卡格

将视图处方表示为数据,而不是以任意效果调用的代码,使得这种视图定义变得可行。

它涉及对视图数据的简单操作。这项工作仍处于早期阶段;虽然系统具有视图构建能力。

为了实现我们的愿望,我们持续寻求最直观的界面,为用户提供这种能力。当前的方案需要明确引用属性、类型和视图,这可能超出了许多用户的能力。最终,我们的目标是让用户能够现场编辑视图,通过将合适的视图元素从一个地方拖到另一个地方来操控信息的呈现。这种“以身作则”的设计很可能在更多用户的能力范围内。

即使拥有理想的工具,许多用户可能也懒得设计新的视图。然而,将视图描述为数据意味着,与其他数据一样,视图可以从其他地方获取并整合到系统中。我们设想,各种高级用户会将视图处方以 RDF 格式发布到网站上,以便其他用户可以找到并整合这些处方,就像个人目前为 MP3 播放器等应用程序定义“皮肤”一样。

从长远来看,我们希望探索机器学习的应用,让 Haystack 自动创建和修改视图。通过观察用户查看和操作信息的方式,系统或许能够推测在特定情境下哪些属性对用户真正重要,并构建仅显示这些属性的视图。

在更高层次上,相同的视图构建框架可用于设计整个工作区——以特定方式布局和呈现的信息对象集合,以支持特定任务的执行。我们将在“工作区”部分讨论这个问题。

操纵

除了查看信息之外,用户还需要能够操作信息。Haystack 的大多数视图都提供即时编辑所呈现信息的功能,以便更改对象的特定语句。更概括地说,Haystack 提供了一个通用框架,用于定义可用于以任意方式修改信息对象的操作。大多数操作都通过右键单击上下文菜单来调用。

在对象上。特别常见的操作由自然的拖放隐喻支持。

运营

Haystack 中的基本操作原语是操作。操作是已具体化并向用户公开的任意函数。每个函数都接受一定数量的参数。当操作被调用时，系统会收集其参数。如果操作仅接受一个参数，并且是在上下文菜单中调用的，则该参数将被假定为被点击的对象。如果需要多个参数，则会在右侧窗格中打开一个对话框来收集其他参数。与许多传统应用程序不同，此对话框是非模态的。它不会强制用户在转到其他任务之前填写完毕。具体来说，用户可以使用 Haystack 的所有导航工具在调用操作之前查找并找到他希望赋予操作的参数（通过将它们拖放到对话框中）。

操作是可以像 Haystack 中的其他对象一样进行操作的对象。具体来说，用户可以将操作拖放到（菜单）集合中，以便从用户希望的任何位置访问它们。

调用操作

上下文菜单提供了一种访问与给定对象相关的所有操作的标准方法。数据模型中的语句声明了哪些操作适用于哪些类型的对象；右键单击会引发数据库查询，该查询会创建适用于所单击对象的操作（及其他项目）的集合。

拖放功能允许用户通过将一个信息对象拖到另一个信息对象上来关联两个信息对象。拖放到集合上，其语义显而易见，就是将对象放入集合中。拖放到镜头中显示的特定属性上，可以将拖动对象设置为镜头所显示对象的该属性值。拖放到对话框参数中，可以将拖动项作为参数分配给正在调用的操作。更一般地说，视图可以指定在将特定类型的对象拖放到视图中时应调用的操作。

80 大卫·R·卡格

定制

与其他数据一样,操作可以由用户自定义。具体来说,用户可以填写一些被调用操作的参数,然后“柯里化”结果,将其保存为一个新的、更专业的操作(Quan, Huynh, Karger, and Miller 2003)。例如,用户可以将标准的“通过电子邮件发送对象”操作填写为其老板的电子邮件地址作为目的地,然后将其柯里化为“将此内容发送给我的老板”操作。

由于柯里化操作仅接受一个参数(要发送的对象),因此可以通过右键单击上下文菜单调用它,而无需任何对话框。创建新操作后,可以为其命名,然后将其拖放到各种命令集合(菜单)中,以便在需要时访问。

我们正在努力为用户提供更强大的操作自定义功能。除了柯里化操作之外,我们还希望允许用户通过组合现有操作来定义新操作 将一个操作的结果作为参数传递给下一个操作。我们还在探索类似我们从网页中提取信息的技术(参见“导入数据”部分),这些技术允许用户将网页操作(通过网页表单访问)封装为 Haystack 操作,然后无需访问网站即可通过 Haystack 界面访问(并自定义)这些操作。

与视图类似,操作也为 Haystack 提供了任意、细粒度的扩展机会。操作以 RDF 格式定义,因此高级用户可以创建并提供,供任何认为有用的人下载。有些操作可能只是精心设计的柯里化操作;其他操作则可能包含新编写的数据库查询,甚至是任意代码。

例子

图 3.2 展示了用户对特定对象调用“发送此项目”操作后发生的情况。右侧窗格中的对话框会收集必要的参数,包括要发送的对象(已填写)和收件人。为了填写收件人,我们展示了用户如何在左侧窗格中下拉特定于电子邮件的历史记录,其中列出了用户在处理电子邮件时最近使用的项目。由于所需的收件人不存在,用户可以在顶部导航栏中的搜索框中执行搜索。与此搜索匹配的(单个)结果将显示在

下拉菜单。从那里,可以将其拖放到对话框中,以指示它是预期收件人。如果用户有理由相信他需要将此特定项目发送给其他人,他可以从对话框(如图所示)中拖放一个上下文菜单,并选择“将此另存为操作”来创建一个新的操作,该操作的发送项目是预先指定的,并且只需填写预期收件人。生成的操作仅接受一个参数(预期收件人),它将在右键单击任何人时下拉的上下文菜单中显示。一个互补的操作,其中收件人是预先指定的,但发送的项目不是,在图3.1的上下文菜单中显示为“发送给Jaime”。

任务

我们认为在 Haystack 中建模的另一个关键概念是任务。无需正式定义,我们认识到许多人会花时间从事通常被称为“任务”的事情:处理电子邮件、规划一天的日程、撰写论文、浏览网页、购买商品等等。对于每一项任务,用户可能需要处理一些信息(电子邮件收件箱、日程规划的日历、正在撰写的论文等等),以及用户在执行任务时可能调用的一系列操作(回复电子邮件、安排约会或检查文档的拼写)。如今,人们似乎经常同时执行多项任务;然而,最多只能同时记住几项任务。

任务窗口

在 Haystack 中,我们正在探索两种支持任务的方法。第一种是图左侧所示的任务窗格。任务窗格可以显示对给定任务有用的对象和操作的集合。例如,在图 3.1 中,我们看到一个“电子邮件”任务窗口,其中包含与电子邮件任务相关的对象(例如收件箱)和操作(例如撰写邮件)。用户可以通过在任务窗口中点击来导航到与任务相关的对象,或调用与任务相关的操作。任务窗口只是呈现一个集合,可以像任何集合一样进行操作。特别是,如果用户决定其他对象或操作

82 大卫·R·卡格

如果某些项目对任务非常有用,她可以将它们拖放到任务集合中,以便将来访问。当然,用户也可以创建全新的任务,并在其中填充相关项目。

任务窗口中还可以看到特定任务的历史记录集合,其中包含用户在执行此任务时最近访问过的项目。与 Web 浏览器中的通用历史记录不同,特定任务的历史记录不会因用户执行其他不相关任务时访问的无关项目而变得杂乱无章。

如果用户在执行特定任务时经常访问某些项目,则这些项目往往会出现历史记录中。因此,即使用户不费力地自定义任务窗口以包含执行任务所需的项目,历史记录也能提供一种自动化的自定义功能,实现同样的效果。

任务窗口比普通应用程序轻量得多,但同时比“最小化”应用程序更详细。我们相信,这种折中方案对于多任务用户来说非常有效。用户无需在处理多个应用程序信息时循环展开和折叠各种全屏窗口,而是可以查看每个任务的少量状态。

任务窗口可以看作类似于可停靠的“工具栏”

目前已在许多应用程序中可用。但是,它们作为标准集合建模意味着用户可以自由地合并任何他们认为有用的对象或操作。

任务窗口以两种方式激活。首先,用户可以明确地开始执行任务。例如,用户可以从起始菜单(右上角)选择“电子邮件”来调用该任务。或者,用户可以在搜索框中输入“电子邮件”,然后从结果集中选择“电子邮件”任务。这类似于启动应用程序:用户明确表示希望开始执行该任务。第二种选择是让系统猜测用户正在执行哪些任务。例如,在图 3.2 中,左侧窗格中显示一个灰色的“地址和联系人”任务。

这表明系统认为用户可能正在执行此任务。

如果用户点击灰色标题,任务窗口将展开,显示与该任务相关的项目。目前,此类猜测是系统硬编码的。某些对象和类型通过适当的 RDF 语句与某些任务明确关联。例如,

收件箱与电子邮件明确关联,因此每当用户导航到收件箱时,左侧窗格都会显示电子邮件任务。从长远来看,我们认为发现用户当前正在执行哪些任务这一问题将成为机器学习研究的一个富有成果的目标。

工作区

我们第二种处理任务的方法规模更大。工作区是一个(可能相当大)的区域,其中包含各种信息对象的(相对详细的)视图,可用于处理给定的任务。例如,传统的电子邮件应用程序可能会呈现一个包含当前消息集合的区域、一个包含特定当前消息的区域、一个包含地址簿的区域等等。正在撰写有关特定研究项目的论文的用户可能希望收集和布局相关的研究数据、有用的引文和文档、拼写检查功能以及向其合著者发送邮件的操作(有关自定义操作,请参阅标题为“操作”的部分)。继续我们的主要论点,即开发人员无法预测最终用户想要什么样的工作区,我们希望赋予最终用户创建自己的工作区的能力,决定应该呈现哪些信息(以及以何种方式)来让他们执行给定的任务。

创建工作区与创建新视图非常相似。视图可能用于处理多条信息,而工作区通常只创建一次,用于单个任务。视图通常显示与正在查看的对象相关的信息,而工作区则显示与要执行的任务相关的信息。从某种意义上说,工作区可以看作是该任务的视图。

鉴于它们的相似性,我们可以将类似于视图构建的工具应用于工作区的构建。要构建工作区,用户需要选择要在工作区中显示的项目集合,为每个项目选择一个视图,并确定这些视图在工作区中的布局方式。项目的选择(创建集合)和(预定义)视图的选择已作为 Haystack 的标准组件提供。我们已经设计了一个原型工具来管理项目的布局,以便创建工作区(Bakshi 2004)。

图 3.3 展示了一个用我们的拖放工具构建的论文写作工作区,它通过组装同样用我们的

84 大卫·R·卡格

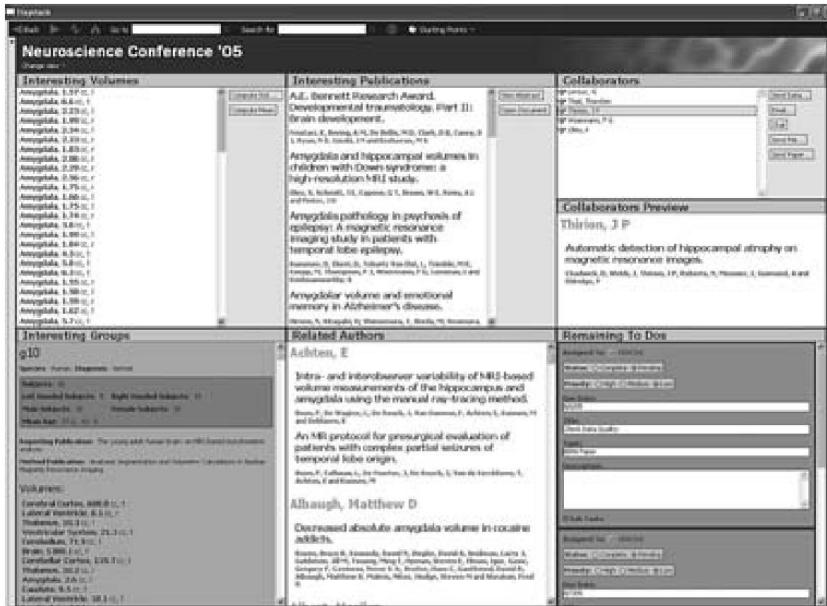


图3.3

通过拖放操作构建的工作区。此工作区专门用于撰写特定的研究论文,展示研究数据、合著者及相关参考文献。出版物视图是使用类似的视图构建工具创建的。

“创建新视图”部分中提到的拖放工具。

搜索

除了阅读和撰写信息之外,搜索或许是信息管理的关键活动。Haystack 提供了许多搜索工具。

我们的目标是使搜索既普及又轻量级 我们希望用户将搜索视为作为常规导航的一部分执行的“无开销”活动,而不是放下他们正在做的事情并启动搜索。

正如我们上文所述,定向越野是一种自然的搜索模式。如果一个合理的起点清晰可见,我们预期用户会通过“超链接”的方式从一个对象导航到另一个对象,最终找到他们想要寻找的对象。通过放置

通过在左侧面板中提供用户可定义的特定于任务的信息集合,我们的目标是最大限度地提高用户找到良好搜索起点的机会。

文本搜索

当然,有时并没有清晰可见的起点。这时可以依靠的一个简单方案是文本搜索。信息对象通常与易记的文本相
关联,例如标题、正文或注释。Haystack 的上方导航栏包含一个搜索框,可在其中输入任意文本查询。此搜索
的结果是一个集合。该集合显示在集合的下拉菜单视图中,这有利于在搜索成功的常见情况下快速选择项
目。但是,也可以“导航至”结果集合,为更复杂的搜索提供起点。

在 Haystack 中,文本与许多项目相关联。不仅仅是传统的数据,还包括操作等其他对象。
因此,用户可以通过描述操作来搜索(然后就地调用)该操作。本质上是动态地指定与给定
描述相关的命令菜单。使用细粒度搜索来定位小项目也变得很自然,例如,查找要作为对话框
参数填写的特定值。

与传统语料库的文本搜索不同,传统语料库中与给定条目关联的文本是明确的(文件中的
文本及其元数据),而将哪些文本与给定的 RDF 资源关联起来则是一个复杂的问题。将任何
通过语句直接连接到资源的文本与资源关联是很自然的,但人们也可以想象将位于语句链上
更远距离的文本关联起来。

模糊浏览

数据库社区对搜索进行了大量的研究。一些人(Bosc、Motro 和 Pasi,2001)甚至在寻找提
供排序或近似匹配的方法,以避免布尔数据库查询令人不快的“全有或全无”效果。然而,正
如我们上文所述,需要注意定向搜索,它在搜索中表现为查询规范、结果检查和查询细化的迭
代过程。沿着语句定向搜索很自然地可以从一个资源中获得

86 大卫·R·卡格

但是当用户开始发出查询时,她会面临一系列更好或更差的匹配,她需要从中进行定向。

Yee 等人 (2003) 探索了分面元数据浏览,以便让用户通过选择限制信息的某些属性来浏览数据集合。

在 Haystack 中,我们正在探索将定向越野工具从文本搜索领域引入数据库领域的方法 (Sinha 2003; Sinha 和 Karger 2005)。我们建议将资源的属性和值 (谓词和宾语) 视为该资源的特征,可用于搜索和相似度评估,就像文档中的单词用于文本搜索一样。换句话说,我们可以将每个项目关联到一个包含“单词”的“虚拟文档”,例如“author:yc1yb87Karger”和“Send-Date:012937”(请注意,URI 保留在术语中是为了区分词典顺序相同但语义不同的值)。我们可以将所有已充分研究的模糊文本搜索技术应用于这些虚拟文档。

例如,对于任何商品,我们可以将“相似商品”定义为那些拥有许多相同属性值的商品。这些商品在我们查看商品时可能值得显示,因为它们很可能有助于用户进行定向越野。文本搜索研究表明,各种术语权重

确定哪些属性在相似性判定中“重要”的方法 例如,极其常见的属性可能应该被忽略。在发起查询、浏览结果和修改查询的常见搜索过程中,文本搜索社区也开发了各种查询细化和相关性反馈技术,可用于建议后续步骤。图 3.1 左侧窗格中显示的正是此类建议。

数据库搜索

我们还提供了一个通用的“查找”接口,允许用户根据 RDF 模型设计数据库查询。目前,它仅限于表达特定谓词必须取特定值的约束。我们在这个接口上投入的精力相对较少,因为我们认为以这种方式表达查询是更轻量级导航工具失败的标志。我们预计,与通用查询接口不同,开发人员可能会将特定有用的查询打包成

使用特定领域对话来捕获制定查询所需信息的操作（如上所述）。

讨论

介绍了 Haystack 系统之后,我们现在来讨论一些设计选择以及我们将继续研究的一些悬而未决的问题。

为什么要有语义网络?

我们对 Haystack 的讨论可能会让人疑惑,我们为什么要使用数据库或结构化模型?用户几乎看不到底层数据库的踪迹;元组从不显示,数据库查询也已弃用。

有人可能会认为,鉴于我们关注的是链接遍历,我们最好简单地将用户信息以 HTML 的形式存储在某种“个人网站”中。

相反,我们认为半结构化数据模型对于个性化信息管理系统的设计至关重要。用户使用的大部分数据显然都是结构化的,主要依赖于属性以及与其他项目的关系。与网络不同,在网络上,每个链接都必须手动标注其角色的文本描述,而结构化模型则提供了一种简洁的机器可读的方式来指示某一类链接所扮演的角色。我们的视图渲染架构可以利用该结构以各种信息丰富的方式渲染信息对象。以机器可读的形式呈现链接意味着,即使复杂的数据库查询超出了最终用户的能力,高级用户可以将复杂的数据库查询(作为操作)和信息呈现(作为视图和镜头)打包,然后由普通用户整合,以增强其系统的功能。更广泛地说,该模型中提供的结构使得编写各种自主代理成为可能,这些代理可以代表最终用户导入和操作数据。

语义网络具有普遍性

我们还认为,语义网络,特别是 RDF,提供了一种自然的“通用数据模型”,应该在

88 大卫·R·卡格

应用程序开发。语义网络足够丰富,能够表示用户需要处理的大量信息。同时,它足够简单,可以轻松融入任何应用程序设计中。所需要的只是一种命名单个对象的机制,以及一种连接这些命名对象的特定关系的表示。

有了这样的表示,应用程序可以轻松使用其他应用程序创建的数据,即使它们不了解其他应用程序的语义。一个对象名称就足以让一个应用程序创建指向另一个应用程序中该对象的实时链接。

应用程序无需深入了解链接的含义即可利用连接两个对象的关系(就像在网络上的情况一样)。应用程序不可避免地会保留一些过于“复杂”而无法在语义网络中公开的信息 例如图像的单个像素,或某些复杂数据文件的模糊分类方案

但这些信息很容易隐藏在网络中命名的单个信息对象中,并由了解这些对象内部结构的应用程序处理。同时,搜索工具可以让用户查询和浏览语义网络所表示的元数据,而无需了解这些关系的语义或它们所关联的信息对象。与文本或文件非常相似,关系具有足够的通用性,值得给出一个标准的表示,因此跨应用程序工具(如剪贴板和桌面搜索引擎)可以帮助减少跨应用程序的数据碎片化问题。

还值得注意的是,每个应用程序所做的大部分工作只是对关系和属性的直接操作。几乎每个应用程序都提供了某种“集合”框架,并使用相同的拖放交互在集合之间移动项目。许多应用程序提供“注释”可以用任意文本填充的自定义字段

同样使用类似的界面。通过应用程序提供这些功能浪费开发人员的精力,也意味着它们不能跨应用程序使用。鉴于预期数据模型操作本质上很简单,我们有充分的理由在系统级公开它,就像在现有系统中公开文件(和 ASCII 文本)操作一样。

图式的作用

虽然我们高度依赖结构化表示,但图式显然并非如此。我们允许用户以违反图式的方式关联任意对象 文档的作者可以是一件家具,交货日期可以是一个人。并且,我们允许用户创建任意的新关系来连接对象,而无需提供任何图式描述。

关于不使用模式总体而言,我们认为这种轻模式方法在个人信息管理系统中是必要的。

给定模式,我们必须选择是否强制执行。就像开发人员设计应用程序一样,我们总会发现用户想要记录违反我们模式的信息。这时,我们必须选择是否强制执行我们的模式,并禁止用户

记录他们认为重要的信息,或者允许违反我们的模式。虽然后者给我们设计系统结构带来了挑战,但前者违背了根本目标:让用户记录他们需要的信息。Mangrove (参见 Halevy 等人,2003) 采取了类似的策略,认为在实践中,模式需要根据现有数据进行设计,而不是相反。

当然,有人可能会说用户并非最了解情况。或许可以将强制执行作为一种教育体验,教会用户如何构建信息。然而,我们怀疑用户过于固执己见,这种方式行不通。即使界面可以引导用户记录信息,

以“正确”的方式记录信息,我们预计用户回来寻找信息时会以他们最初设想的“错误”方式寻找,因此由于信息被记录得“正确”,他们无法找到信息。我们需要按照我们预期用户寻找信息的方式来记录信息,即使我们预期他们会以错误的方式寻找信息。

关于使用图式:虽然我们并不打算强制使用图式,但它们仍然遍布 Haystack。为了保持一致性,我们确实会尝试引导用户使用合理的信息图式。我们预计,Haystack 最初分发的大量图式所建立的“预先存在的条件”将导致用户

90 大卫·R·卡格

类似的大型知识表示,以便标准视图、查询和操作可以与它们一起工作。

模式在视图设计中扮演着尤为重要的角色。具体来说,我们大量使用类型断言来决定合适的视图和操作;使用高度非标准类型系统的用户也需要高度非标准的接口来使用它。选择在给定类型的对象的视图中显示哪些属性也是基于模式的。它表达了一种预期:这些属性通常可用,而其他属性不可用(或不重要)。

当用户修改视图时,他们实际上实际上是在修改与所查看类型关联的模式。然而,一个关键的区别在于,视图所建议的模式约束是“软约束”。虽然视图暗示了某些属性是预期的,但缺少这些属性就会导致不显示任何信息。我们可以在图 3.1 中看到这一点:虽然收件箱显示提示需要与每个对象关联发件人和日期,但实际上可以将一个人包含在集合中,唯一的后果是出现一些空白字段。同样重要的是,多个视图意味着,从某种意义上说,可以根据用户正在执行的任务,在不同时间对同一对象施加不同的模式。

虽然我们不强制执行模式,但我们用户界面的操作原语通常会提供强有力的建议。关于给定属性是单值还是多值的模式注释会影响拖放行为:拖放到单值字段上可能会替换属性的值,而拖放到多值字段上可能会为属性添加一个附加值。同样,这些建议并非严格执行:只要用户付出足够的努力,就可以为模式化的单值属性添加第二个值。此时,假设单值的视图最终可能会不确定地只显示两个赋值中的一个。当然,用户始终有机会修改视图来修复此缺陷。而数据库查询则不受视图约束地处理数据,可以充分利用多值。

我们使用模式的根本原因在于一个普遍的研究问题:如何利用“通常为真”的数据库模式。我们已经讨论过通常为真模式如何促进信息视图的设计。在程序员层面,模式可以帮助开发人员编写更清晰的代码,因为它们可以避免处理复杂的案例分析。

使用数据。举一个简单的例子,知道某个给定的属性始终存在意味着可以跳过处理其缺失所需的代码。

一个有趣的问题是,在多大程度上,真正的模式可以用来维护清晰的代码?目前,Haystack 操作充斥着各种处理模式异常的代码块 例如,按日期排序的操作需要显式检查每个日期是否确实是 date 类型。在其他情况下,操作在遇到意外异常时会默默失败(可以说,这是合理的行为,实际上就是拒绝将操作应用于违反模式的数据)。人们或许希望编写代码,将所有违反模式的情况隐式捕获,并分支到某种异常处理块。但这引出了一个问题,即如何描述异常处理代码,特别是如何清晰地描述违反模式的方式以及违反模式时应采用的默认设置。

Haystack 的局限性

我们对 Haystack 的使用凸显了其设计中的各种局限性和缺陷。其中一个显著的缺陷是“UI 模糊性”。由于屏幕上的每个对象都是动态的,用户界面有时很难猜测用户点击时要访问哪个对象。屏幕上的任何一点通常都包含在多个层级嵌套的对象中,当用户点击时,并不清楚他要访问的是嵌套的哪一层。对于上下文菜单,我们通过允许作者访问点击点处嵌套的所有对象的菜单来解决这个问题。如图 3.1 所示,上下文菜单提供了对电子邮件发件人、包含该发件人的电子邮件消息以及包含该电子邮件消息的收件箱的操作。当用户拖放对象时,我们会做出启发式决策,在点击和放置点处处理“最具体”(层次结构中最低)的对象。这通常是正确的,但有时也会导致问题。例如,为了将一个项目拖放到集合的显示上,必须仔细寻找集合显示中不属于任何递归渲染集合成员的部分。关于

消除 UI 操作歧义的最佳方法。

我们赋予用户数据模型的权力也可能造成损害。

Haystack 并没有为用户提供足够的保护,因为用户执行的操作可能会破坏他们的数据。除了用户自己的数据之外,由于

92 大卫·R·卡格

如果整个界面都被描述为数据,用户可能会以各种方式破坏界面,导致其无法使用。例如,用户可以将视图与其呈现的数据类型分离,然后突然发现自己无法查看信息。

解决这个问题的正确方法是开发有效的数据访问控制(尤其是写控制)方法。我们尚未解决这个关键问题,因此将其作为下文中的未解决问题提出。

其他应用

在本节中,我们推测了我们所创建的架构的一些其他角色:让用户使用语义网工作(Berners-Lee、Hendler 和 Lasilla 2001)产生的半结构化数据,并让个人用户通过共享或发布一些他们自己的半结构化信息为这项工作做出贡献。

语义网

无论人们是否认同每个用户桌面都需要一个语义网络,随着所谓的语义网(Berners-Lee、Hendler 和 Lasilla,2001)的发展,语义网络似乎注定会在信息传播中扮演关键角色。网络是一个极其丰富的信息来源,但其 HTML 文档以“人类可读”的形式呈现信息。也就是说,文档的语义由人类基于其对人类语言的理解进行解码。此类文档不易被试图代表用户提取和利用信息的自动化代理所理解。

因此,人们正在努力以 RDF 和 XML 的形式在网络上呈现信息,因为这些形式更适合自动化使用。

人们或许会认为,语义网相比传统网络提供的更丰富的语义,也能提升人类用户从中检索信息的能力。但目前情况恰恰相反,因为语义网目前尚无完善的接口。在语义网中,数据和服务以语义丰富的机器可读方式公开,但用于检查这些数据的用户界面(即使存在)通常是由集中式数据和服务组合而成的。例如,通过语义门户(例如 SEAL,Stojanovic 等人,2001)或语义搜索(Guha、McCool 和 Miller,2003),数据库管理员可以聚合

将语义分类信息集中到集中式服务器上,分发给网络用户。这有助于用户通过传统的网络浏览器访问语义网。

但 Web 门户界面与传统应用程序有着同样的缺点。一位设计师似乎不太可能设计出一个“恰到好处”的界面,让所有不同的用户都能使用。此外,任何一个门户的设计都基于一个固定的本体;来自语义网其他部分(“其他应用程序”的任意信息无法自动整合到门户生成的视图中)。

如果某个模式被增强,任何门户都将无法呈现增强模式中的信息,除非门户开发人员修改其显示系统。因此,门户将我们带回到我们试图用语义网络模型消除的割据信息结构。

另一方面,如果用户的客户端软件能够基于每个用户执行这种数据聚合和用户界面构建,那么我们就能够恢复用户在语义网中自由浏览信息和服务的能力。我们的视图架构提供了这样一个在客户端集成数据的机会(Quan and Karger 2004)。关于单个资源的独立信息,过去需要通过多个不同的网站进行导航,而现在这些信息可以合并到一个屏幕上,而且这种合并无需专门的门户网站或网站/数据库之间的协调即可完成。此外,适用于某条信息的服务无需打包到包含该信息的网页中,也无需跨网站复制粘贴信息来访问服务;资源与可以使用它们的服务(操作)的语义匹配可以由客户端完成,并以菜单的形式公开。通过设计和分发视图和操作,用户可以创建和发布查看现有信息的新方式,而无需修改原始信息。

来源。

协作与内容创作

到目前为止,我们的讨论主要集中于一位用户与她

信息(然后是语义网)。但我们相信,我们的系统可以增强个人记录知识以供社区使用,以及更广泛的社区对这些知识的搜索和使用。

94 大卫·R·卡格

万维网的一大优势在于它能够吸引

语义网极大地降低了个人与更广泛社群分享知识的门槛。任何个人无需复杂的工具支持,就能记录信息,并供他人查找和访问。如果语义网也能实现同样的功能,那么用户记录的信息将更加丰富,对其他个人(以及自动化代理)来说,其价值将远超纯HTML。

不幸的是,用于创作语义网信息的最先进工具是图形编辑器,它直接将信息对象显示为节点,将属性显示为连接这些节点的弧(Eriksson等人)。

1999;Pietriga,nd)。与简单的HTML编辑器相比,此类工具需要更老练的用户,因为简单的HTML编辑器可以让新手用户将他们的知识发布到万维网上。

Haystack使用户能够轻松编写结构化信息,这些信息已经以语义网的原生RDF格式表示。

这降低了用户决定向外界公开部分“内部使用”信息的门槛。传统上,阅读文档并进行注释以供自己使用的人必须做大量的

将这些注释(可能还有文档)转换为HTML格式,以便在网络上发布。有了语义网络表示,文档和注释就已经具备了在语义网上发布的正确格式,用户只需决定谁有权访问它们。

当然,访问控制问题本身就很棘手,而数据模型的精细粒度更是雪上加霜。我们需要一个简单的界面,让用户指定哪些对象的哪些属性和关系应该对哪些人可见。

另一方面,当信息从众多来源收集时,个人必须开始做出信任决策。同样,必须开发界面,让用户指定他们希望将哪些语义网断言作为“真理”纳入自己的语义网络中。

用户协作时必须解决的另一个重要问题是模式分歧问题。如果允许每个用户修改其信息表示,那么在数据交换时,这些表示不太可能保持一致。我们希望通过共享视图规范和操作以及数据来改善这个问题。

我们应该在这里提到的一项相关工作是 REVERE 系统,特别是 MANGROVE 项目 (Halevy 等人,2003 年)。

REVERE 与 Haystack 的许多目标和方法相通。与 Haystack 一样,REVERE 的目标是在结构化和非结构化信息之间找到一个有用的节点。Haystack 专注于帮助每个人更好地管理自身信息。而 REVERE 则将协作视为首要目标。因此,对于 Haystack 来说,可以推迟到未来的模式对齐问题,成为了 REVERE 设计的主要驱动力。

相关工作

近期许多研究都强调了以应用程序为中心的数据管理所带来的问题,并提出了扩展或协调应用程序以解决该问题的方法。Bellotti 等人 (2003) 观察到电子邮件应用程序被用于任务管理,并展示了如何增强电子邮件应用程序的“视图”以支持这项额外任务。Ravasio、Guttormsen-Schär 和 Krueger (2004) 提供了用户在尝试执行数据跨多个应用程序的任务时遇到的问题的证据。在本书中,Kaptelinin 和 Boardman 认为,必须采取“以工作区为中心的方法”,将任务所需的数据汇集在一起,而不是仅将数据集中在单个应用程序管理的范围内。

过去曾有多项努力将信息管理的核心置于关系的理念之上。Presto 项目 (Dourish 等人,2000) 提议废除以静态目录作为关键的组织框架文档,转而基于针对每个文件记录的元数据的查询来确定位置。本书讨论的 Lifestreams (Freeman 和 Gelernter) 则主要关注一种元数据:上次使用时间。然而,这两个系统仍然将文件作为信息的基本单位,并强调整查询,而不是链接、显示和浏览。

ObjectLens 系统 (Lai 和 Malone,1988) 显然是我们在 Haystack 中探索的许多理念的先驱。ObjectLens 强调了从类型层次结构中提取的任意信息对象的概念,这些对象具有属性以及与其他对象的链接。OVAL (Malone,Lai 和 Fry,1995)

96 大卫·R·卡格

是一种使用“对象、视图、代理和链接”快速创建应用程序的工具,类似于 Haystack 中使用的工作区设计。

WinCuts 工具 (Tan、Meyers 和 Czerwinski,2004) 展示了一种从应用程序中释放数据的替代方法:它将小窗口切分成多个应用程序,以便可以查看这些应用程序中的单个数据(靠近其他应用程序的数据),而不会影响应用程序的其余部分。由于 WinCuts 在像素级别运行,因此它非常通用 它可以从任何应用程序附近获取信息。但这同时也是它的弱点。由于统一的只是不同应用程序的像素,而不是数据,因此 WinCuts 不会在多个应用程序中的数据之间建立额外的语义链接。只有当两个底层应用程序已设置为共享数据时,才能将数据从一个 WinCut 拖到另一个 WinCut。

本卷中的几章提出了一些值得记录信息对象之间有趣的新关系,或对现有或新关系提出了有趣的新可视化方法(例如 Fisher 和 Nardi、Freeman 和 Gelertner、Plaisant 等人)。在当前的应用程序开发方法下,这些工具都必须从头开始开发,并且必须投入大量精力来连接和远程控制现有应用程序以处理给定的数据对象。对于每种新方法,都必须重复进行此类集成工作。如果有人雄心勃勃地尝试构建一个包含所有不同方法的应用程序,那么工作量将进一步增加。这似乎很浪费,因为每种方法的核心思想都只是跟踪一些额外的关系,并创建一些利用这些关系的视图。像 Haystack 这样的基础设施可以更轻松地将这些新的关系和视图合并到一个系统中,并在适当的时候调用它们,而不是为每个新想法构建新的、不同的应用程序(并说服用户迁移到这些应用程序)。

结论

Haystack 框架展示了在半结构化数据模型中统一管理用户信息的一些好处。它将数据与呈现分离,打破了当前应用程序模型对信息操作的限制。它

将允许用户精确收集解决特定任务所需的信息，并使用最能传达所需信息的视图将其可视化。

笔记

1. Nicolson Baker (1994)在著作中直接观察到了这一现象,他感叹在将纸质卡片目录转移到电子数据库的过程中,顾客用铅笔写在卡片上的有趣信息丢失了。
2. 尽管我们对回报持乐观态度,但快速浏览一下同事的办公室和办公桌就会发现,我们中的许多人都过于短视,现在不投入组织的努力,而这些努力将在以后带来更好的回报。

参考

Adar, E., Karger, D. 和 Stein, AL (1999). Haystack: 基于用户的信息环境。载于《第八届信息与知识管理国际会议论文集》, 第413-422页。密苏里州堪萨斯城, 11月2日至6日。

Baker, N. (1994). 丢弃。《纽约客》 68 (4月): 81-83。

Bakshi, K. (2004). 面向最终用户的信息管理任务界面创建和定制工具。硕士论文, 麻省理工学院。

Bellotti, V., Ducheneaut, N., Howard, M. 和 Smith, I. (2003). 将电子邮件转化为任务: 以任务管理为中心的电子邮件工具的设计与评估。载于《2003 年计算机系统人为因素大会论文集: 计算系统中的人为因素》, 第 345-352 页。佛罗里达州劳德代尔堡, 4 月 5 日至 10 日。

Berners-Lee, T., Hendler, J. 和 Lasilla, O. (2001). 语义网。《科学美国人》 284 (5): 34-43。

Bosc, P., Motro, A. 和 Pasi, G. (2001). 第四届灵活查询应答系统国际会议报告。SIGMOD记录 30 (1): 66-69。

Codd, EF(1970). 大型共享数据库的关系数据模型。ACM通讯 13(6): 377-387。

Dourish, P., Edwards, WK, LaMarca, A., Lampert, J., Petersen, K., Salisbury, M., Terry, DB, 以及 Thornton, J. (2000). 使用用户特定的活动属性扩展文档管理系统。ACM信息系统学报 18(2): 140-170。

Eriksson, H., Fergerson, R., Shahar, Y. 和 Musen, M. (1999). 本体编辑器的自动生成。载于1999年第12届班夫知识获取研讨会论文集。加拿大班夫, 10月16日至22日。

Guha, R., McCool, R. 和 Miller, E. (2003). 语义搜索。载于2003年万维网大会论文集, 第700-709页。匈牙利布达佩斯, 5月20-24日。

98 大卫·R·卡格

Halevy, A., Etzioni, O., Doan, A., Ives, Z., Madhavan, J., McDowell, L. 和 Tata-rinov, I. (2003)。跨越结构鸿沟。载于首届创新数据系统研究双年会 (CIDR)论文集。加州阿西洛马,1月5日至8日。

Hogue, A. (2004)。万维网上包装器归纳的树形模式推理与匹配。麻省理工学院硕士论文。

Hogue, A. 和 Karger, D. (2005)。Thresher:自动从万维网中解析语义内容。载于第14届国际万维网大会 (WWW)论文集,第86-95页。日本千叶,5月10-14日。

Huynh, D., Karger, D. 和 Quan, D. (2002)。Haystack:一个使用 RDF 创建、组织和可视化信息的平台。WWW2002语义网研讨会。夏威夷檀香山,5月 7 日。

Karger, DR 和 Jones, W. (2006)。个人信息管理中的数据统一。ACM通讯, 2006 年 1 月。49 (1): 77-82。

Lai, K.-Y. 和 Malone, TW (1988)。ObjectLens:用于协同工作的电子表格。ACM办公信息系统学报6 (4): 332-353。

Malone, TW, Lai, K.-Y. 和 Fry, C. (1995)。OVAL 实验:一种可彻底定制的合作工作工具。ACM信息系统学报13(2): 177-205。

马诺拉,M.,和米勒,E. (2003)。 RDF底漆。 <http://www.w3.org/TR/rdf-primer/>。

McGuinness, DL 和 van Harmelen, F. (2003)。Owl 网络本体语言概述。<http://www.w3.org/TR/owl-features/>。

Muslea, I., Minton, S. 和 Knoblock, C. (1999)。一种分层的包装器归纳方法。载于《第三届自主代理国际会议论文集》(Agents 99), 第 190-197 页。华盛顿州西雅图,5月 1 日至 5 日。

彼得里加,E. (nd)。伊萨维兹。 <http://www.w3.org/2001/11/IsaViz/>。

Quan, D., Bakshi, K., Huynh, D. 和 Karger, DR (2003)。支持多分类的用户界面。载于INTERACT:第九届 IFIP 国际人机交互会议论文集,第 228-235 页。瑞士苏黎世,9 月 1 日至 5 日。

Quan, D., Bakshi, K. 和 Karger, DR (2003)。语义网消息传递的统一抽象。载于《第十二届国际万维网大会论文集》,第231页。匈牙利布达佩斯,5月20日至24日。

Quan, D., Huynh, D., Karger, D., 和 Miller, R. (2003)。用户界面延续。载于《用户界面系统与技术》(UIST)论文集,第 145-146 页。

148. 加拿大温哥华,11 月 2 日至 5 日。

Quan, D. 和 Karger, DR (2003)。Haystack:一个用于编写终端用户语义网应用程序的平台。载于2003 年国际语义网会议论文集。佛罗里达州萨尼贝尔岛,10月20日至23日。

Quan, D. 和 Karger, DR (2004)。如何制作语义网浏览器。在第 13 届国际万维网会议论文集,第 255-265 页。

纽约市,5 月 17 日至 22 日。

Ravasio, P.,Guttormsen-Schär, S. 和 Krueger, H. (2004)。追求桌面进化:现代桌面系统的用户问题与实践。ACM人机交互学报 (TOCHI) 11 (2): 156–180。

Sinha, V. (2003). 动态利用可用元数据进行浏览和信息检索。麻省理工学院硕士论文。

Sinha, V. 和 Karger, DR (2005)。Magnet:支持半结构化数据环境中的导航。载于SIGMOD 05: 2005 年 ACM SIGMOD 国际数据管理会议论文集,第 97-106 页。马里兰州巴尔的摩,6 月 13-16 日。

Stojanovic, N.,Maedche, A.,Staab, S.,Studer, R. 和 Sure, Y. (2001)。SEAL:用于开发语义门户的框架。载于K-CAP 01:第一届国际知识捕获会议论文集,第 155-162 页。加拿大维多利亚,10 月 22-23 日。

Tan, DS, Meyers, B. 和 Czerwinski, M. (2004)。WinCuts:操控任意窗口区域,更高效地利用屏幕空间。载于ACM 计算机系统人为因素 CHI 2004 会议论文集扩展摘要,第 1525-1528 页。奥地利维也纳,4 月 24 日至 29 日。

Teevan, J.,Alvarado, C.,Ackerman, M. 和 Karger, DR (2004)。完美的搜索引擎是不够的:定向搜索中的定向越野行为研究。

在2004 年 ACM CHI 计算机系统人为因素会议论文集,第 415-422 页。奥地利维也纳,4 月 24-29 日。

Yee, P.,Swearingen, K.,Li, K. 和 Hearst, M. (2003)。用于图像搜索和浏览的分面元数据。载于ACM CHI 计算人为因素会议论文集,第 401-408 页。佛罗里达州劳德代尔堡,4 月 5 日至 10 日。

4

任务管理探索 桌面

乔治·罗伯逊、格雷格·史密斯、布莱恩·迈耶斯、帕特里克·鲍迪什、玛丽·Czerwinski、Eric Horvitz、Daniel Robbins 和 Desney Tan

介绍

越来越多的任务要求用户协调和操作来自多个来源的信息。每个信息源通常包含在一个窗口中，窗口是用户当前操作信息的基本单位。随着计算和网络功能的不断进步，用户可以打开大量窗口，每个窗口包含不同的信息。通常，用户受益于同时查看存在于不同窗口中的相关信息。此外，这些信息的空间布局对于有效的任务执行可能至关重要，因为它不仅可以帮助用户建立空间关系，还可以直观地比较内容。

本书的后续章节介绍了旨在使用户更高效地管理和执行任务的各种项目。

由于对构成连贯活动的理解存在差异，每个项目对任务的定义也各不相同。我们通过访谈发现，许多最终用户通常将任务定义为一组窗口及其操作。

例如处理财务、撰写论文或管理信件，这些工作都可能涉及一组不断变化的窗口和/或应用程序。如今，用户在管理这些窗口和任务方面面临着越来越大的挑战。在本章中，我们将介绍我们构建的工具，这些工具允许用户有效地操作桌面窗口以完成他们的任务。

任务管理研究

在执行一项任务时,用户通常需要同时查看多个窗口 (Kandogan and Shneiderman 1997)。此外,研究人员观察到,信息工作者经常在多个窗口之间切换,

租赁任务的发生,要么是因为多任务活动 (Bannon 等人,1983) ,要么是因为外部中断 (Cutrell,Czerwinski 和 Horvitz,2001;Czerwinski,Cutrell 和 Horvitz,2000; Czerwinski,Hovitz 和 Wilhite,2004;Gillie 和 Broadbent,1989;Maglio 和 Campbell, 2000) 。因此,我们关注的两个主要问题是：(i)有效地管理多个窗口和任务；(ii)从任务切换和中断中恢复。

大量研究探索了窗口管理系统,该系统允许用户管理屏幕上的多个窗口 (历史和回顾,参见 Myers 1988) 。在传统的桌面隐喻中,管理任务可能涉及数十项操作,包括打开、移动和调整窗口大小,以及滚动内容。这可能极其繁琐,并在用户执行主要任务时显著增加认知负荷。此外,桌面隐喻对保存和任务切换的支持不足,导致用户浪费精力并感到沮丧 (Card,Robertson 和 York 1996;Henderson 和 Card 1987;Kandogan 和 Shneiderman 1997;Robertson,Card 和 Mackinlay 1993;Robertson 等人 1998) 。

Card 和 Henderson (1987) 在他们的 Rooms 系统中提出了解决这些问题的方法。他们观察到,可以通过管理窗口工作集来支持任务,其方式与操作系统管理内存中工作集的方式非常相似。在这项研究中,他们发现了任务管理系统的几个理想特性,包括快速任务切换、快速任务恢复以及在中断后轻松重新获取心理任务上下文。Rooms 系统提供了一种机制,可以将特定窗口与特定任务关联,并方便用户在这些任务之间轻松切换。

作为桌面隐喻的扩展,一些现代操作系统提供了虚拟桌面管理器。这些管理器允许用户将窗口组织到多个虚拟桌面上,并在它们之间轻松切换。目前有许多这样的系统可用,并在 XDesk 2003 中进行了描述。虚拟桌面隐喻将物理

任务管理探索 103

虚拟桌面管理器会以视口的形式显示在一个更大的虚拟空间中。因此,使用虚拟桌面管理器的用户可能需要跟踪窗口和任务在相当大的空间内的布局。尽管用户可以将每个虚拟桌面视为不同的任务,但大多数虚拟桌面并不提供明确的任务管理功能。尽管虚拟桌面管理器取得了一些成功,但关于这些系统的实用性和可用性的文献却很少,尤其是在任务管理方面(参见 Ringel 2003)。

除了虚拟桌面管理器之外,还提出了许多用于管理大量窗口的替代解决方案,包括使用额外的低分辨率屏幕空间扩展用户桌面(Baudisch、Good 和 Stewart 2001)、将桌面扩展到 3D 空间(Wurnig 1998)、扩展到 Pad++ 中的可缩放空间(Beder-son 和 Hollan 1994)以及扩展到时间维度(Rekimoto 1999)。

此外,一些涉及移除其他窗口的系统(Bell 和 Feiner 2000;Hutchings 和 Stasko 2004;Kandogan 和 Shneiderman 1997)和平铺窗口管理器(Bly 和 Rosenberg 1986;Morris 等人 1986;Teitelman 1986)也解决了其中一些问题。弹性窗口采用填充空间的平铺布局,并通过允许用户创建可拖动多个窗口的容器来解决同时显示多个窗口的问题(Kandogan 和 Shneiderman 1997)。

3D Rooms(Robertson、Card 和 Mackinlay,1993)扩展了 Rooms 的理念,使用 3D 虚拟环境来表示信息工作空间。该系统并非严格意义上的窗口管理器,因为抽象的信息可视化取代了窗口。该系统的基本动机是利用人类的空间认知和感知,从而简化任务管理。Web Forager(Card、Robertson 和 York,1996)和 Data Mountain(Robertson 等人,1998)都使用虚拟环境,以便在管理文档时更充分地利用人类的空间认知和记忆。Data Mountain 的研究(Czerwinski 等人,1999)

1999 年;Robertson 等人 1998 年)证明,将文档放置在空间中有助于用户在后续检索时记住文档的位置。我们的工作也采用了这种方法。我们提供的工具旨在将人类空间认知和感知的优势应用于管理当前计算机应用程序中的窗口和任务。

我们的方法

在执行任务时,用户需要轻松访问包含相关信息的特定窗口和应用程序。因此,我们认为,一个有效的任务管理系统应该提供一些机制,使用户能够轻松地对相关的窗口进行分组、组织组内窗口、在组之间切换,以及调整组和窗口在屏幕上的显示布局。

在本章中,我们介绍了三个探索任务管理不同方面的系统: GroupBar、Scalable Fabric 和 Task Gallery。

GroupBar 为现有的 Microsoft Windows 任务栏添加了新的语义,用于组织和管理任务。Scalable Fabric 使用缩放和“焦点在上下文”隐喻来可视化相关窗口组。在此系统中,所有任务都经过缩放并位于外围,以便同时可见。最后,Task Gallery 是一个 3D 环境,其中任务的组织和管理基于画廊的物理世界隐喻。针对每个界面,我们都进行了用户研究,以总结在设计过程中获得的经验教训,从而测试系统的可用性。

分组栏

GroupBar 的设计目标是通过扩展当前的 Windows 任务栏设计来提供任务管理功能。GroupBar 保留了基本的任务栏磁贴功能,为系统中每个打开的窗口显示一个磁贴,并以较暗的按钮状态显示当前“活动”窗口磁贴。点击任何磁贴都可以激活相应的窗口,或者最小化已处于活动状态的窗口。

GroupBar 超越了现有任务栏的功能,提供任务管理支持。它允许用户将代表打开窗口的磁贴拖放到称为“组”的高级任务中,并通过单击鼠标在这些任务之间切换。由于原始任务栏的磁贴上没有定义拖动交互,因此即使用户不使用此分组功能,也可以像使用常规任务栏一样使用 GroupBar。GroupBar 与 Windows 任务栏的相似性不仅可以利用熟悉度来缩短学习时间,也为我们提供了针对性地比较 GroupBar 提供的任务管理功能的基础。

GroupBox 中的任务形成

使用 GroupBar,用户只需将一个窗口磁贴拖到另一个磁贴上,即可将多个窗口分组到不同的任务中。拖动过程中,一个白色插入符号会沿着栏杆移动,以跟踪指针位置并提示拖放操作的结果(图 4.1a)。当一个组形成后,GroupBox 会以灰色背景包围各个磁贴,并在顶部添加一个绿色“标签”来统一各个磁贴的视觉效果。

用户可以通过重复此拖放操作将窗口添加到组中,也可以通过将图块拖出组来取消分组。当组缩小为单个图块时,剩余的图块将自动取消分组,并且“组”选项卡将消失。

GroupBox 中的任务组织

目前 Windows 的任务栏会按照底层应用程序的启动顺序显示窗口图块。然而,我们觉得

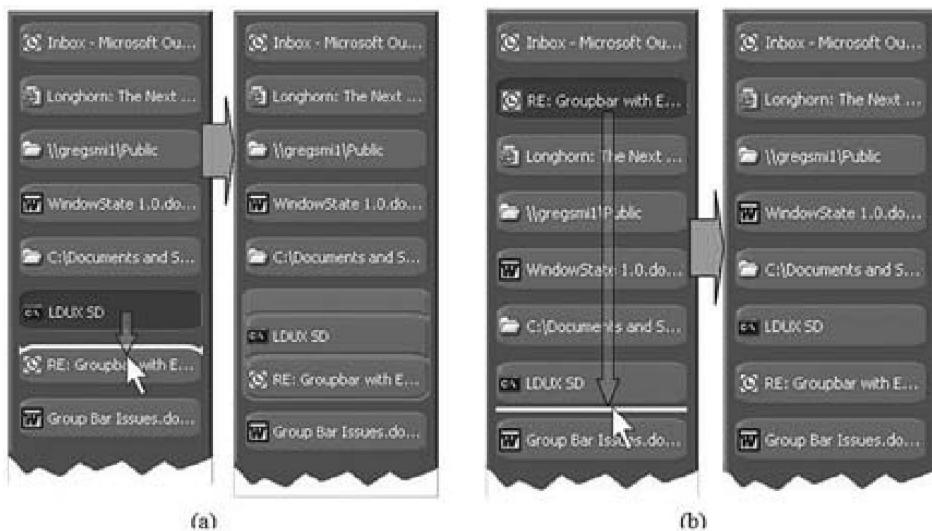


图4.1

GroupBox 的基本外观和功能:在(a)中,将一个窗口拖到另一个图块上方,即可将两个图块合为一个组。分组信息主要通过分组图块形状和周围颜色的细微变化来传达,并使用绿色的“组”选项卡作为高对比度的统一器和控制界面。

在(b)中,在其他图块之间拖动窗口图块或组选项卡会重新排列图块的顺序,以允许用户更好地利用空间记忆。

如果允许用户明确排序磁贴,用户就能更轻松地找到他们的窗口和任务。因此,我们允许用户重新排列任务以及单个磁贴,以便在 GroupBar 中提供更有意义的顺序(图 4.1b)。此功能通过将项目拖放到现有磁贴或组之间的栏上所需位置来实现。通过这项小小的改进,我们进一步巩固了将空间定位作为记忆和任务切换辅助手段的理念。

在拖放操作中同时支持分组和重新排序语义需要精心设计。分组插入符号(见图 4.2)与重新排列插入符号的区别不仅在于位置,还在于曲率,曲率提供了更明确的视觉提示。事实上,正是出于这种考虑,我们选择了弧形窗口磁贴设计,而不是之前原型和任务栏本身使用的纯矩形磁贴设计。

如图 4.2 所示,包含直线插入符号的屏幕空间太小,用户无法有效地获取该空间。Group-Bar 通过将目标表面与插入符号的视觉位置解耦来解决这个问题。将图块中心之间的屏幕空间均匀地分配给三个相邻的目标,而与插入符号的出现位置无关。到目前为止,我们对这种目标表面分配的经验是积极的,更大的最小放置区域的好处似乎超过了绝对位置精度的不足。

从前两张图可以看出,GroupBar 可以随时通过将栏拖动到屏幕的不同边缘来配置为水平或垂直形式,就像标准 Windows 任务栏一样。

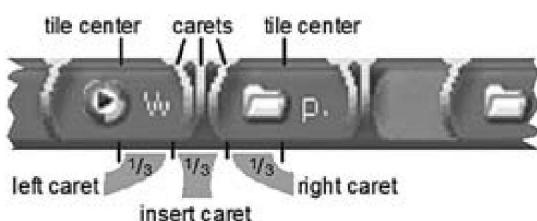


图4.2
栏片段及其上所有可能的插入符号。为了让用户轻松获取插入的放置位置,GroupBar 将激活面均匀分布在各种可能性上。

在 GroupBar 中切换任务

用户可以通过点击任务的绿色“组”选项卡快速切换任务。GroupBar 中的组级操作类似于使用 Windows 任务栏最小化和恢复窗口的现有窗口级操作。点击非活动组选项卡可激活该组。点击活动组选项卡可在最小化组内所有窗口和恢复所有窗口之间切换。借助这个简单的扩展，GroupBar 成为了高效的任务切换工具。

通过单击新任务的 GroupBar 图块可以在多窗口任务之间切换，就像通过单击新窗口的相应图块可以在单个窗口之间切换一样。

虚拟桌面管理器往往会严格区分任务，而 GroupBar 则特意允许用户通过点击各个磁贴，同时显示任意窗口子集，即使它们被分配到不同的任务。为了实现最大的灵活性，我们在 GroupBar 的右键菜单中添加了多个选项，用于控制在不同窗口和任务之间切换的体验。例如，“点击群组时最小化其他窗口”二元选项有助于在切换到新任务时减少不必要的屏幕混乱。

GroupBar 中的任务布局

除了任务切换之外，我们还在“组”选项卡的右键单击（上下文）菜单中添加了许多窗口管理功能（见图 4.3）。

如前所述，在管理一个或多个多窗口任务时，为了显示正确的信息而反复移动和调整多个窗口的大小非常繁琐。我们将熟悉的窗口“最大化”功能扩展到组级别，发现了解决部分窗口布局问题的机会。显而易见的是，最大化组中的每个窗口会导致组内所有窗口相互重叠，这很可能没有用。因此，我们扩展了这个类比，创建了一个“布局组”操作，用于最大化组占用的集体空间，而不是最大化每个单独窗口占用的空间。通过使用用户提供的分组信息，系统可以做出更多



图4.3

点击绿色的“组”选项卡可恢复该组中的所有窗口并将其置于前台。右键单击“组”选项卡可执行其他操作。

更智能地选择如何最有效地利用整个屏幕空间来完成特定任务。GroupBar 的当前实现提供了一个子菜单（如图 4.4 所示）, 用户可以在多个布局模板中进行选择, 这些模板会根据系统的屏幕配置进行调整。

这项实现只是我们朝着实现此类布局辅助的全部潜在价值迈出的一小步。首先, 我们设想未来的组布局选择器会用组中实际窗口的缩略图预览来填充样式化的窗口表示。这将为用户提供最强的控制感和对操作效果的可预测性。此外, 布局应考虑组中窗口的数量及其当前的大小和位置, 以便为给定的组呈现最具体、最定制的选项集。定制化可以进一步

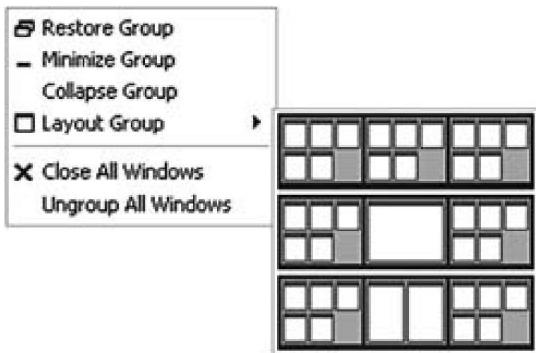


图4.4

GroupBar 上下文菜单允许用户根据布局模板排列该组中的所有窗口。此处,用户使用水平配置的三显示器显示屏,因此每个布局都跨越三个屏幕。

通过允许用户设计自己的布局模板,或使用学习算法开发与个人使用模式随时间变化相符的自动定制模板,可以实现这一目标。更进一步,我们设想能够利用窗口和任务本身的语义和视觉内容,为用户提供更高效、更智能的布局选择。

扩大规模

我们的研究表明,显示屏尺寸的增加会导致打开的窗口数量增加,从而导致未来桌面更加拥挤。因此,我们必须创建能够平滑扩展以适应更多托管窗口的设计。随着显示窗口数量的增加,任何类型的固定栏界面最终都会耗尽空间。GroupBar 扩展了任务栏当前使用的机制,并实现了两种处理不断增加的窗口和任务数量的方法。

在标准 Windows 任务栏上,“分组相似任务栏按钮”选项默认启用。该选项会将同一应用程序的所有窗口磁贴合并为一个任务栏磁贴,点击后会弹出菜单。遗憾的是,任务栏“分组”标准(即应用程序)与用户对当前任务的理解并无必然关联,因此也未必如此。

这与用户自定义组中固有的频率和新近度关系有关。因此,尽管任务栏的分组机制确实有助于回收空间,但它实际上阻碍了窗口和任务的切换,因为它会使空间记忆无效,并将一组相对随意的窗口降级到隐藏位置,而不管它们对当前任务的重要性如何。

我们处理大量窗口的方法利用了 GroupBar 对群组的认知。群组具有“折叠”功能,可以通过群组上下文菜单上的“折叠群组”命令显式触发,也可以在特定情况下自动触发(例如,特定栏上的磁贴空间不足时)。群组折叠后会以更节省空间的形式呈现(如图 4.5b 中垂直方向所示,图 4.5c 中水平方向所示),在群组选项卡内仅显示每个窗口的图标。GroupBar 中的“自动折叠非活动群组”选项会折叠任何当前没有活动窗口的群组。即使仅仅依靠简单的“最近最少使用”群组标准来仅在绝对必要时折叠,我们也相信 GroupBar 的功能是对现有任务栏的改进。

当“分组”策略不足时,任务栏允许用户使用小箭头手柄翻阅多组图块(如图 4.5a 所示)。这会导致大量潜在相关的图块难以访问,而且由于没有关于当前图块“页码”的反馈,查找单个图块的过程可能会非常漫长。或者,用户可以调整任务栏的大小以显示更多图块,但这会直接占用用于显示窗口内容的空间。

GroupBar 无需在栏上翻阅多个图块集或增加单个图块的大小,而是允许在同一桌面的不同边缘同时实例化多个图块。其他图块最初为空,可使用 GroupBar 上下文菜单中的“添加新图块”命令添加。用户可以将新创建的图块放置在任何显示器的任何外部边缘,然后使用与在图块内使用的相同的拖放机制进行填充。这使 GroupBar 能够有效地处理大量窗口和组。遵循我们尝试更好地利用空间内存的原则,这还补充并扩展了简单的图块重新排序功能,允许更广泛的 2D 放置机会,无论是由用户明确移动还是由系统自动移动。

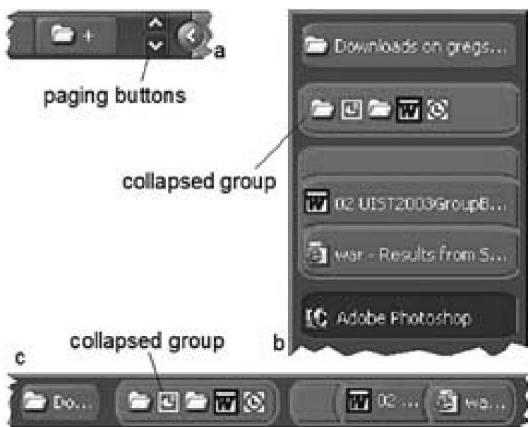


图4.5

(a)任务栏溢出 vs. (b,c)GroupBox 中的折叠组。

初步研究结果

我们开展了两项涉及 GroupBar 的研究。第一项研究是为期 7-10 天的现场研究,参与者为五位用户,他们在自己的多显示器系统上使用 GroupBar 处理自己的工作。在本研究中,我们的目标是确定 GroupBar 的设计是否会与现有任务栏用户的期望相悖,并确定新的分组功能是否易于学习。为此,我们仅向参与者提供了 GroupBar 可执行文件和一封简短的电子邮件教程,指导他们如何使用其分组功能。由于 GroupBar 原型并未与任务栏本身集成(这需要对 Windows 操作系统进行修改),因此用户被指示在运行 GroupBar 时隐藏现有任务栏。所有参与者都履行了承诺,将 GroupBar 作为主要任务栏使用一周,并且他们没有报告任何安装 GroupBar 或理解其使用方面的问题。正如我们所期望的那样,用户能够轻松地将 GroupBar 融入他们现有的工作实践中,他们的评论和分组习惯就是明证。

使用 GroupBar 一周后,四位参与者填写了一份用户满意度问卷,内容涵盖了系统感知到的优势以及需要改进的地方。一位用户没有交回问卷。用户满意度调查结果令人满意,其中两位

参与者表示,尽管 GroupBar 存在一些缺陷且与真实任务栏缺乏集成,但他们仍愿意在研究结束后继续使用它。

GroupBar 在回答以下问题时得分高于平均水平:

- 通过拖动来对 GroupBar 上的图块进行分组非常有用
- 彼此“之上”。
- 能够同时关闭/打开一组窗口很有用。
- 让 GroupBar 记住一组窗口的布局很有用,这样它们打开时的布局与用户关闭窗口时的布局相同。

团体。

- GroupBar 使多台显示器更加有用。

另一方面,用户没有报告发现在组切换时最小化非组窗口或一次运行多个 GroupBar 的实用性。

我们询问用户 GroupBar 的哪些功能最有助于他们管理打开的窗口,以及我们还可以做些什么来为 GroupBar 设计更好的窗口/任务管理支持。在回复中,我们发现了一些共同点。四分之三的用户谈到了他们的窗口分组,这暗示他们认为“分组”概念适用于他们的工作习惯。几乎所有关于改进的建议都归结为两条。第一条建议是 GroupBar 和任务栏需要统一,以便任务栏的其他功能(系统通知托盘、“开始”菜单等)能够在单个栏上以分组语义的方式使用。当然,这代表了我们一直以来的愿景。GroupBar 原型的设计仅针对任务栏的平铺行为,并且肯定需要整合任务栏的其他非平铺功能才能完全发挥作用并成为任务栏的替代品。第二条建议的改进则更为复杂。用户建议 GroupBar 在适当的情况下执行“自动分组”,这样即使不进行简单的拖放操作,也能享受分组的益处。从用户的评论来看,他们似乎并不认为这项功能特别难实现,事实上,我们在设计阶段就已经考虑过这项功能。我们研究了几种策略,从简单的(始终将新窗口添加到最近的组中)到复杂的(使用窗口标题文本相似性进行分组),但最终都被否决了。

每种策略都可能出现错误,而不是正确。我们意识到,要始终保持正确的自动分组策略,必须检测并理解用户的意图,这是一个极其困难的问题。

第二项研究结果

我们决定收集小规模实地研究中的初步反馈,更深入地理解 GroupBar 相对于任务栏的易用性假设。我们进行了一项对比实验室研究,18 名参与者分别使用任务栏和 GroupBar 执行限时任务。

每项任务都要求参与者(均为经验丰富的 Windows 用户)在任务内切换窗口才能完成;例如,复制/粘贴或引用其他文档。此外,实验者会系统性地打断用户正在执行的一项任务,以促使其切换到另一项任务。我们想看看 GroupBar 能否通过减轻任务栏在单窗口级别执行任务管理的限制所带来的内置任务切换开销,从而实现可衡量的生产力提升。

用户评论说,强制切换的任务和中断与他们在现实世界中经历的类似,因此我们认为该实验成功模拟了信息工作者的日常任务切换。

我们发现 GroupBar 在任务完成时间上具有略微显著的优势,如图 4.6 所示。虽然定量结果当然可以更进一步,但 GroupBar 从技术上来说是一款“新”的任务切换工具,而且我们将其与所有参与者都拥有多年日常使用经验的现有工具进行比较,这让我们感到非常鼓舞。更令人鼓舞的是用户满意度问卷的结果。如表 4.1 所示,用户在所有问题上都显著偏向 GroupBar,而非任务栏(由方差分析和 Bonferroni 校正多重检验确定,均在 $p < .05$ 的显著性水平上)。

最后,大家一致认为 GroupBar 比任务栏更受欢迎。尽管如此,许多参与者还是提出了一些改进 GroupBar 的建议。最常见的建议是,为 GroupBar 中组织的不同任务添加颜色编码或标签,以及在组折叠时添加显示文档名称的工具提示。这些功能可以轻松添加到 GroupBar 中。

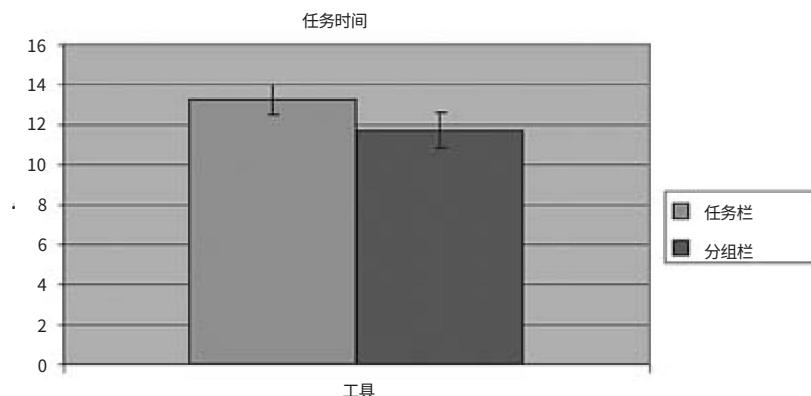


图4.6

平均任务时间 +/- 任务栏和 GroupBar 平均值的一个标准误差。

表 4.1

任务栏和组栏的平均满意度评分

调查问题 (1 = 不同意, 5 = 同意)	任务栏	分组栏
使用...可以轻松执行任务切换。	2.95	4.63
使用...在我的各个窗口和应用程序之间来回切换非常困难。	3.32	1.42
我对...的功能感到满意。	2.68	4.42
[任务栏/组栏]是 Windows 的一个很有吸引力的创新。	3.16	4.42

注意:所有评级在 $p < .05$ 水平上均明显有利于 GroupBar。

一些专家用户希望在 GroupBar 中启用更好的键盘加速器支持 (例如用于在组之间切换的 Alt-Tab)。

概括

我们希望 GroupBar 能够让用户轻松快捷地对窗口进行分组和重新分组,然后让他们能够像操作单个窗口 (或任务)一样对多个窗口组进行操作。我们认为,通过整合更广泛的空间布局偏好设置,为用户提供更高级别的组织结构 (即“组”),并将现有的窗口操作功能扩展到“组”级别,我们可以设计出一种基于现有

任务栏隐喻。我们认为我们已经实现了这些目标：实地研究表明，参与者认为 GroupBar 很有价值，而实验室研究使我们能够在更可控的环境中，结合熟悉的现有技术，更好地验证这些优势。这些研究进一步证明，像 GroupBar 这样的软件工具可以在用户管理多个复杂任务时提供帮助。我们发现，只需解决现有的限制，即窗口管理机制只能在越来越不自然的“低”级别（单个窗口级别）运行，就能真正改善任务管理体验。

在下一节中，我们将介绍可扩展结构，这是一种专为未来计算显示表面设计的任务管理系统，届时大型显示器或墙壁投影将取代当今大多数用户交互的较小、更孤立的显示表面。

可扩展结构

在设计我们的第二个原型“可扩展结构”时，我们超越了当前操作系统中占主导地位的一维任务栏隐喻。

设计这个原型的目标是展望未来，届时用户将拥有更大的屏幕，包含更多的窗口。

在我们公司开展的一项非正式研究（Hutchings 等人，2004）中，我们发现，使用更大屏幕的专业用户会同时运行更多应用程序并打开更多窗口。例如，我们观察到，在 16 位用户中，单显示器用户平均同时打开 1 个或 4 个窗口，而双显示器用户平均同时打开 12 个窗口，三显示器用户平均同时打开 18 个窗口。

Scalable Fabric 是一个基于管理 Windows 桌面上的多窗口“任务”的系统，这次使用焦点加上下文显示来根据用户的注意力分配屏幕空间。可扩展

Fabric 允许用户通过缩小窗口并将其移动到外围来始终保持窗口或窗口组打开并可见。可扩展 Fabric 是一种“焦点加上下文”的显示方式，即当用户将注意力集中在主要任务上时，外围显示的其他工作（即竞争或潜在相关的任务）的上下文信息。这种外围显示方式既利用了用户的空间记忆，也利用了用户的视觉识别记忆。

用于图像,以便于任务识别和定位 (Czerwinski 等人,1999)。该机制的灵感来自于 Flatland 中显示边缘的缩放 (Mynatt 等人,1999) 以及 ZoomScapes 基于位置的缩放机制 (Guimbretiere、Stone 和 Winograd,2001)。

虽然 ZoomScapes 不是一个任务管理系统,但它对工作表和工作表组的管理类似于 Scalable Fabric 对窗口和任务的管理。

为了方便任务切换,Scalable Fabric 允许用户将同时使用的窗口集合分组,类似于 GroupBar 的功能,但这种方式可以更有效地利用空间记忆。我们从 Data Mountain 的用户研究中了解到 (Robertson 等人,1998),空间记忆在虚拟环境中的工作方式与在现实世界中类似,并且当用户执行的任务涉及检索空间放置的物品时,其任务绩效会得到显著提升。

在本节的剩余部分,我们将首先描述 Scalable Fabric 方法论的细节。我们将展示 Scalable Fabric 与 Windows 任务栏的比较用户研究结果,以及 Scalable Fabric 的纵向实地研究。最后,我们将讨论项目方向和未来研究的机会。

可扩展结构中的布局

在可扩展结构 (Scalable Fabric) 中,用户通过将外围边界标记移动到所需位置,在显示表面上定义一个中心焦点区域。在图 4.7 中,这些边界标记可见 (由细蓝色矩形定义),但用户通常会隐藏边界标记,除非他们要更改焦点区域的大小或形状,在这种情况下,这些标记将用作调整大小的手柄。用户对焦点区域位置和大小的选择通常受物理显示器的配置和功能的影响。例如,在三显示器显示器上,用户可能更喜欢将中央显示器定义为焦点区域,不设置上下外围区域,并将侧面显示器作为唯一的外围区域。

在焦点区域内,窗口的行为与 Windows 桌面上的正常行为相同。外围区域包含当前未使用但可随时使用的窗口和窗口簇 (或任务)。外围区域的窗口较小,以便在用户关注其他内容时可以容纳更多任务。基于此比喻,我们相信用户很少需要关闭或最小化窗口。



图4.7

Scalable Fabric 将三个任务显示为窗口集群,其中一个来自“大学”任务的窗口显示在焦点区域。在这种情况下,显示中心焦点区域边框的选项已启用。

传统意义上的。用户可以利用额外的屏幕空间,尤其是在较大的显示器上,以使外围窗口始终可见。

当用户将窗口移至边缘时,窗口会随着与焦点边缘边界距离的单调缩小,越靠近屏幕边缘,窗口就越小。当用户点击边缘的窗口时,窗口会返回到上次焦点位置;这是新增的“恢复”行为,通过一秒钟的动画将窗口从一个位置移动到另一个位置来实现。当用户“最小化”焦点区域内的窗口(例如,点击窗口的“最小化”按钮)时,窗口会返回到上次边缘位置。

可扩展结构中的任务形成

Scalable Fabric 使用自然的隐喻和手势,允许用户定义、访问和切换任务。为了定义任务,外围的窗口被分组到用彩色横幅包围的簇中(参见图 4.8 和 4.9)。要创建新任务,用户只需将外围的窗口移动到不属于任务的另一个窗口附近即可。然后,用户可以命名

隐式创建的任务。在任务被命名之前,它是短暂的。也就是说,如果任务中的项目数量减少到 1,则该任务将被移除(即任务标记将消失)。将窗口移动到任何任务标记附近,使其成为该任务的一部分。此行为使用户可以轻松地通过将窗口拖放到现有窗口或窗口集群上来构建任务。

可扩展结构中的任务组织

当一个窗口移到外围时,其他窗口会暂时移开。这类似于 Data Mountain (Robertson et al. 1998) 中采用的遮挡避免行为,使得一个外围窗口无法被另一个外围窗口遮挡。

当集群移动时,它们会像窗口互相避开一样互相避开,不同之处在于,静止的集群会保持在原位,而移动的集群会绕其移动。为了在 Scalable Fabric 中移动和缩放窗口和集群,我们参考了 ZoomScapes (Guimbretiere, Stone 和 Winograd, 2001) 的研究结果。由于窗口是矩形而非点,因此确定缩放发生的点非常重要。与 ZoomScapes 一样,Scalable Fabric 使用光标位置(即拖动点)作为缩放点。我们尝试了几种替代方案,并同意早期研究的观点:

光标位置是最有用的刻度点。

移动集群时,仅缩放集群中的窗口是不够的。ZoomScapes 会缩放页面中心与光标拖动点之间的距离。在 Scalable Fabric 中,我们通过缩放窗口中心与集群中心之间的距离来实现更令人满意的效果。也就是说,随着集群变小,窗口之间的距离会越来越近。当窗口跨越缩放边界时,缩放比例的突然变化会令人感到不安。ZoomScapes 通过设置一个桥接区域来解决这个问题,在该区域内应用缩放比例的急剧变化。Scalable Fabric 采用了不同的方法,并根据新的缩放比例应用了半秒的过渡动画。

这看起来比斜坡区域方法更为优雅。

可扩展结构中的任务切换

在 Scalable Fabric 中,用户可以使用自然手势访问和切换任务。当用户点击任务标记时,整个任务将被选中,其窗口将恢复到焦点位置。如果用户点击

当所有窗口当前位于焦点区域时,点击任务标记,集群中的每个窗口将返回到其外围位置。如果用户选择了一个任务,并点击了另一个任务标记,则会发生任务切换,即当前任务的所有窗口将移动到其外围位置,而外围所选任务的窗口将移动到其在焦点区域中的先前配置。

迭代设计

我们遵循迭代设计流程,不断完善和测试可扩展架构的各个版本。迄今为止,我们已经创建了该系统的三个实现版本。

Scalable Fabric 的第一个版本是一个原型,它使用窗口图像,这使我们能够改进视觉设计和交互行为。我们进行了非正式研究,收集可用性问题,以推动第二次设计迭代。虽然用户理解了基本概念,但他们在理解任务标记(一个 3D 卡片夹)方面存在很大困难。

第二种设计适用于 Windows 桌面上的真实窗口。一项将 Scalable Fabric 与 Windows XP 任务栏进行比较的用户研究表明,Scalable Fabric 易于学习,并被参与者认为很有价值,但也发现了一些可用性问题。任务标记被重新设计成旗杆的形状,但测试参与者仍然难以识别它是什么。此外,我们发现任务遮挡回避行为会造成混淆。在这个版本中,当一个任务被移动时,其他任务会移开(类似于外围窗口相互避开以防止遮挡窗口的方式)。一项研究表明,两种方法在任务执行时间上没有显著差异。

Scalable Fabric 的第三个版本(如图所示)是在第二个设计的基础上进行改进而开发的。图 4.8 展示了窗口和任务标记外观的特写,光标悬停在一个窗口上时会显示其标题工具提示。大多数情况下,任务标记的显示方式如图 4.9 所示。但是,如果用户将鼠标悬停在标记上或将窗口移动到任务组中,则会显示一个框,如图 4.9 所示。根据前两个设计的反馈,最终设计中的任务标记更加简洁。此外,我们重新设计了任务遮挡避免机制,使移动的任务能够避开其他任务,而不是

120 乔治·罗伯逊等人



图 4.8任务
特写（第三个设计）。



图 4.9悬停
时任务突出显示。

让其他任务移开。测试参与者发现这更加直观。

为了进一步了解用户实际使用虚拟桌面管理器的方式，并更详细地了解 Scalable Fabric 在实际场景中的应用，我们开展了一项纵向研究，研究对象为 13 位参与者，他们使用了各自的真实系统和任务。这项研究揭示了设计迭代的新机遇。具体来说，随着设计中核心问题的解决，系统性能和错误修复对我们的最终用户而言变得更加重要。

Scalable Fabric 的一个版本于 2005 年向公众发布。发布后的四个月内，约有 11,300 人下载并使用了它。总体而言，用户反响良好，大多数用户仍在继续使用。然而，一些特定于实现的性能和行为问题导致一些人在评估期后停止使用原型。其中一些问题可以通过修改当前实现来解决，但其他问题则需要重写 Scalable Fabric 来替代旧版窗口

经理。

后续步骤

Scalable Fabric 采用“焦点+上下文”的空间隐喻，提供基本的任务管理功能。位于中心焦点区域的窗口运行正常，而位于显示边缘的窗口则缩小为“最小化”窗口。由于占用空间较少，边缘窗口可以保持打开状态并保持活动状态。只需单击鼠标即可完成任务切换。两项用户研究为 Scalable Fabric 的迭代设计提供了指导，并表明用户更喜欢这种方式而不是标准的 Windows 任务栏，尤其是在使用多显示器或大型显示器时。这些研究也发现了一些仍需解决的问题。其中许多问题可以归因于决定在现有窗口管理器的基础上构建 Scalable Fabric，而不是在现有窗口管理器中构建或替换现有窗口管理器。

可扩展结构的未来实施将解决这些问题。

然而，现在有了强大的显卡和更好的渲染支持，探索未来的桌面交互方式也很有趣，这使得虚拟 3D 桌面成为一种可行的替代方案

典型桌面用户界面的候选对象。我们将在下一节中通过任务库来解决这个问题。

任务库

与 Scalable Fabric 类似,任务库创建了任务的可视化呈现,并允许用户轻松切换任务。任务库还利用用户的空间记忆进行任务管理。在任务库(图 4.10)中,当前任务显示在虚拟艺术画廊尽头的舞台上。它包含该任务已打开的窗口。

其他任务则放置在画廊的墙壁、地板和天花板上。用户可以通过点击切换到新任务,并将其移动到舞台上。

只需单击按钮即可同时查看多个窗口,并利用 3D 空间中的自动布局和移动功能,提供统一且直观的缩放比例。应用程序和常用文档保存在用户虚拟左手的“开始”面板中。研究



图4.10
任务库。

任务管理探索 123

表明用户对任务库很感兴趣,可以轻松导航空间,并且可以轻松查找任务并在它们之间切换。

任务画廊设计

选择可导航的空间隐喻是出于利用人类空间记忆的愿望。之所以选择美术馆,是因为它为人所熟知。为了方便检索,任务图库除了包含空间位置和标题提示外,还包含空间中文档和任务的图像。经典的助记研究表明

以视觉图像形式呈现的心理线索是增强记忆的绝佳方式 (Patton,1990)。我们之前的研究表明,快照/缩略图线索在网页存储和检索过程中对空间记忆具有强大的辅助作用 (Czerwinski 等人,1999)。

现有的 Windows 桌面隐喻使用菜单 (尤其是“开始”菜单) 和工具栏来方便用户访问常用的工具和文档。为了更好地契合在走廊中移动的隐喻,并借鉴 Glances 和 Toolspaces (Pierce 等人,1999) 的理念,我们设计了 Task Gallery,以便用户携带与虚拟身体相关的工具和文档。Glances 是一种轻量级、短暂的虚拟环境浏览方式,无需移动虚拟身体。Toolspaces 放置在用户周围,用于存放各种工具或对象,并在虚拟身体在虚拟环境中移动时保持与虚拟身体的关联。

任务库在用户的左、右、上、下四个位置都设有工具空间。工具空间中对象的手和脚的显示,使对象的比例更加明显,并暗示这些工具在用户导航过程中始终伴随用户。在任务库中,可以通过图 4.11 所示的控件来启动扫视。扫视效果会一直持续到用户在工具空间中选择某个对象或扫视到其他地方为止。

左侧工具空间包含“起始面板”,它是一个数据山 (Robertson 等人,1998),外观类似于艺术家的调色板 (图 4.12)。最初的数据山是一个倾斜的 3D 平面,用于存放收藏的网页。起始面板上的对象是应用程序、收藏文档或网页的图标和快照。起始面板的行为类似于数据山,包括对象移动和遮挡避免。唯一的区别在于,选择一个对象

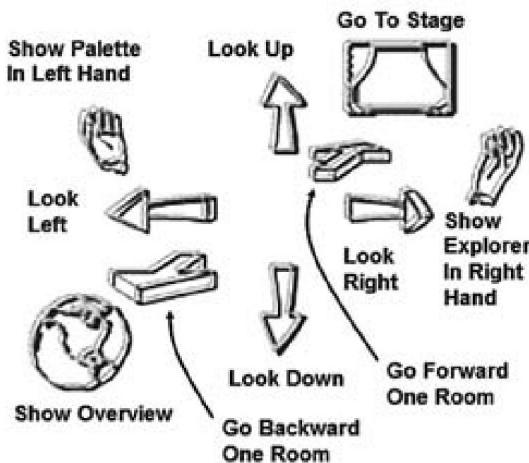


图 4.11 屏幕 3D 导航控件出现在屏幕的左下角。



图 4.12 开始面板 数据山,位于用户左侧的工具空间中。

任务管理探索 125

从“开始面板”启动应用程序时，其窗口会显示在当前任务中。应用程序启动后，Glance 会终止。我们的用户测试表明，参与者学会了如何使用“开始面板”轻松地将应用程序和文档添加到他们的任务中。

早期对数据山的研究（Robertson et al. 1996）表明，用户应该能够比在开始面板中更快地找到图标传统的“开始”菜单。

任务库中的任务形成

可以通过选择菜单或开始面板上的“新任务”项来创建新任务。系统会选择背景图像以区分新任务和现有任务。用户尚不知道新任务的期望位置，因此它被放置在舞台前面的地板上。地板上的其他任务被移离舞台，为新任务腾出空间。这是通过三步动画完成的：将摄像机移回以使动作可见，将地板上的任务移回并将新任务放置在地板上，最后按照前面描述的方式执行任务切换。三步动画是用户测试的结果，它极大地提高了任务创建的可用性。我们假设用户稍后会将任务移动到图库中更合适的位置。

任务库中的任务组织

用户可以通过拖动将任务移动到任何所需的位置。

任务被限制在墙壁、地板或天花板上，但可以以受物体联想（Bukows-ki 和 Sequin 1995）启发的方式在这些表面之间移动。例如，从墙壁到地板的过渡会导致任务在地板上移动到合适的方向。任务框架向外倾斜，以便从远处更清晰地辨认。

墙上的任务画框被安装在支架上，使隐喻更加清晰，并在视觉上更具深度。将画廊划分成不同的房间，将任务窗口分组，并根据作品进行分组，以及使用独特的背景，所有这些都提供了地标和空间线索，有助于记忆。

用户（尤其是非游戏玩家）往往会在许多需要导航的 3D 系统中迷失方向。我们通过保持空间

简单（线性走廊）,通过选择适合上下文的隐喻（在画廊观看艺术品）,并通过限制导航。

因此,我们提供了一些简单的控件,而不是通用的以自我为中心的导航机制。图 4.11 展示了这些屏幕控件,它们允许用户“跳转”到后退、前进、主页（主视图）以及显示任务库中所有任务的鸟瞰视图。每个跳转控件都会启动从当前位置到目标位置的一秒钟摄像机动画。我们的研究表明,用户在使用这些控件时不会在 3D 空间中迷失方向,并且可以轻松找到所需的任务。

在任务库中切换任务

在任务库中,切换任务和同时查看多个窗口都是简单的操作。此外,任务库提供了一个强大的空间框架,用于编码位置信息和前后关系,从而调动用户的空间记忆来检索任务和服务。任务切换只需点击库中的框架即可完成。一秒钟的动画用于强化空间隐喻。当前任务通过创建快照来关闭,快照会移回任务在库中的框架。然后,新选定的任务会从其框架移到舞台上。当它到达舞台时,它会从艺术作品转换为实时窗口。任务的“重影”视图会保留在库中,以标记其来源位置。

初始和主要工作视图是舞台的特写（图 4.14）,显示当前任务及其活动窗口。要查看其他任务,用户需要后退以查看画廊的更多内容,如图 4.10 所示。画廊由一系列房间组成,只有一个封闭的末端;更多房间



图4.13

当用户指向窗口横幅时,窗口操作控件就会出现在其上方。

随着用户向后移动,内容会无限地展现。这提供了一种管理用户注意力的简单方法。随着用户向后移动,注意力会得到拓展。移动到舞台上可以将注意力集中在当前任务上。

任务库中的布局

舞台上的当前任务包含多个组件,包括松散堆栈、有序堆栈和选定窗口集。松散堆栈用于重叠窗口,其使用方式与当前桌面隐喻非常相似。这些窗口安装在支架上,使其在视觉上与舞台融为一体。点击其中一个窗口会将其移动到选定窗口的位置,并替换当前选定的窗口。

图 4.13 所示的窗口操作控件用于移动窗口并将其放置在不同的堆栈中。当用户指向窗口横幅时,这些控件会显示在横幅上方。

松散堆栈中的窗口可以直接移动到舞台上的任何位置。

使用类似于兴趣点物体移动的技术 (Mackinlay,



图4.14
具有有序堆栈和一个选定窗口的舞台。

Card 和 Robertson 1990),鼠标控制在垂直于视线的平面内的移动,而 Shift 和 Control 键控制朝向或远离用户的移动。

有序堆栈显示在舞台左侧,如图 4.14 所示。用户选择将窗口放置在有序堆栈中,以便将当前未使用的窗口(例如,打开的电子邮件消息)保持有序。如果移动了舞台上的某个窗口,堆栈就会被整理,使每个窗口之间的距离保持固定。点击有序堆栈中的某个页面,会将其移动到所选的窗口区域。

选择窗口后,系统会将其移近用户,以提高可读性。可以使用图 4.13 中的“添加到选择”图标选择多个窗口。每次添加窗口时,自动布局都会移动窗口,使它们并排可见(图 4.15)。

与平铺窗口管理器裁剪窗口并可能强制用户滚动不同,此操作不会影响选定窗口中可见的内容。

因此,我们利用 3D 中的距离以直观的方式提供统一的缩放比例。



图4.15
选定多个窗口。

任务库用户研究

我们的前三项研究考察了各种可用性问题解决前后的任务管理情况。第三项研究在前两项研究几个月后进行,评估了为响应前两项研究而添加的功能(例如图标识别)。此外,我们还关注空间认知如何与任务库等3D环境相关,以及现实世界空间位置记忆的某些方面是否能够迁移到电子环境中。

我们特别关注用户创建和修改任务以及安排整体任务空间的能力。此外,我们还收集了关于组织和检索策略的详细信息,以便在未来的设计中支持这些策略。我们想知道组织策略是基于频率、大小、内容类型还是时间。虽然美术馆的隐喻表明要使用墙壁而不是地板和天花板,但先前的研究表明,在现实世界中,某些身体轴线被认为是主要的(Bukowski 和 Sequin 1995;Feiner 等人 1993)。我们想知道参与者的组织策略以及随后的检索表现和空间表征是否与隐喻的属性相关,还是与相对于用户在空间中的方向的上下、前后和左右轴线相关。

用户研究 1 和 2 的结果

方法 11 名年龄在 16 至 65 岁之间的参与者(5 名女性)参与其中 5 名参与者评估了第一个迭代原型,6 名参与者评估了第二个迭代原型。任务库的两个原型版本使用了文档的“快照”,除了应用程序不是实时的之外,它们是完全交互的。在实验期间,用户创建任务,以对他们有意义的方式组织任务,检索 8 个任务及其特定内容项,最后执行各种 Windows 操作。在第一次实验后,我们要求用户画出他们眼中的走廊是什么样子的,以及他们在走廊中的位置和方向。在会话结束时,用户尽可能详细地绘制了他们在走廊中的信息布局。此外,他们还填写了一份用户满意度调查问卷。

在第一次研究和第二次研究之间,我们根据观察到的用户问题对原型进行了一些修改。我们改变了任务的创建、命名和选择时的标签方式。

参与者在画廊左右墙上放置的任务明显多于天花板或地板。这种趋向于遵循现实世界画廊空间通常使用方式的倾向表明,参与者正在使用隐喻来引导互动。在这两项研究中,墙壁、地板和天花板的可读性相同。

参与者的任务组织方式包括对相关任务进行空间分组。“一起完成”的任务会被紧密地放在同一表面上。研究人员观察到各种组织策略,包括按使用频率、使用地点(例如,在家还是在工作)、语义关系和字母顺序排序。此外,大多数参与者使用了不止一种组织策略。

18% 的任务被召回但放置不正确。
对这些错误的分析表明,记住一项任务是放在左墙还是右墙比记住其深度顺序(即,它是距离舞台最近、其次是距离舞台最近,等等)更困难。

92% 的位置错误是由于在错误的墙上进行绘画任务造成的。只有 8% 的错误是由于绘制任务的相对深度顺序错误造成的。这与空间阵列记忆的文献(Franklin and Tversky 1990; Siegel and White 1975)一致,这些文献发现前后关系比左右关系更容易记住。这支持了我们的设计,表明用户会利用 3D 提供的前后关系来表征和回忆任务位置。

总体而言,鉴于这是对原型的首次评估,用户满意度评分为积极。第一次和第二次迭代的平均满意度评分均为4.9分(采用7分制,7分为积极)。

用户研究 3 结果 实时任务库

九位年龄在16岁至52岁之间的参与者(其中三名女性)参与了本次迭代测试,测试系统包含实时Windows应用程序。为了进行这项研究,我们在研究之前确定并创建了八个任务及其内容。这些任务通常包含5到11个文档(例如Word文档、Excel电子表格、网页和电子邮件)。需要注意的是,本次迭代测试包含许多

任务管理探索 131

任务中的文档数量比前两次迭代更多,因为我们感兴趣的是任务库如何扩展到更大的文档数量。因此,我们没有尝试与前两次迭代进行任何定量比较。

平均而言,用户 44% 的时间能够识别 Windows 控制图标 (图 4.13),48% 的时间能够正确匹配图标。

考虑到用户在图标评估时没有见过任务库,也不知道在这个环境中可以做什么,这个结果并不令人意外。使用系统不到十分钟,所有用户就了解了新颖的3D窗口控件的操作方式及其独特功能。

此次迭代的满意度评分更高。总体平均满意度评分为 5.3 (标准差 = 1.4),采用 7 分制,7 分表示满意。平均而言,用户认为 Task Gallery 比他们目前的 Windows 软件更受欢迎 (平均评分 = 5.0,7 分表示更喜欢 Task Gallery)。

在课程结束时,我们询问参与者他们将任务摆放在哪里,以及他们选择这些空间位置的原因。大多数参与者认为将任务放在天花板或地板上会违反“任务画廊”的隐喻。一些参与者只是不喜欢将任务放在地板上。然而,有两位参与者提到,由于放置角度的原因,天花板和地板上的任务更难阅读。但在实验1和实验2中测试的原型中,情况并非如此。由于解决了一些严重的纹理管理问题,系统最终版本中出现了可读性问题。

讨论

任务库是对 3D 虚拟环境在窗口和任务管理中的应用的探索。其动机是希望利用人类的空间认知和感知能力,并充分利用 3D 图形硬件即将普及的潜力,使其不仅仅局限于电脑游戏领域。

用户测试表明,任务库确实有助于任务管理,而且使用起来很愉快。但我们只是触及了皮毛。

在我们的可用性研究中,我们观察到用户展现出许多与现实世界相同的空间认知原则。用户对任务的前后顺序有很强的感知,很少混淆

记忆中的排序。我们将继续探索在未来的 3D 环境中利用用户现实世界知识的隐喻。

构建 3D 窗口管理器的关键技术挑战之一是让现有应用程序在 3D 环境中运行,而无需更改或重新编译它们。这需要操作系统提供输出和输入重定向功能。输出重定向要求在应用程序更新其视觉显示时发出通知,以便系统可以强制应用程序渲染屏幕外的位图,以便在 3D 环境中用作纹理。输入重定向会导致鼠标和键盘事件由应用程序接收,而不是由 3D 环境的主窗口接收,但鼠标坐标会从 3D 转换为 2D。

我们计划对任务库进行几项改进。

我们已经发现,更优质的地标可以显著帮助用户记住他们将任务放在了哪面墙上。此外,Data Mountain 遮挡避免算法可以用来帮助避免移动任务帧时出现遮挡问题。

我们的目标是设计一个 3D 窗口管理器,解决当前桌面隐喻的两个问题:任务管理和多个窗口的比较。任务库是解决这些问题的第一代系统,它基于通用的应用程序重定向技术构建,该技术使我们能够探索适用于应用程序环境的替代用户界面。

任务持久性

任何任务管理系统如果在重启后丢失所有积累的任务信息,最终都会变得非常有限。与当今大多数常见的操作系统一样,Microsoft Windows 并没有提供任何标准化机制来封装特定正在运行窗口的状态,使其能够被持久化或重建。事实上,任何此类机制要想发挥作用,都需要应用程序开发者的配合。因此,目前还没有针对任务持久性问题的理想解决方案。本章中我们描述的三个系统都构建在未经修改的 Windows 系统之上,因此它们提供的任务持久性解决方案的范围有限。

我们明确设计了 GroupBar,采用快速、轻量级的交互机制,以降低组织工作量,并希望由此

任务管理探索 133

鼓励即使是短暂的群组也应如此。然而,有些任务需要长期执行,需要持久化机制。我们引入了“快照”的概念,它是一个带有时间戳的二进制文件,包含一系列窗口标题、窗口位置、窗口缩略图、窗口持久化字符串和群组成员信息。我们使用多种技术尝试从窗口中提取重新创建窗口所需的持久化字符串。我们尝试保存拥有该窗口的应用程序的名称和路径,以及表示窗口当前内容的“文档”字符串。如果我们的技术不足,我们还允许用户编写自定义持久化字符串。为了在界面中显示快照,我们在 GroupBar 中添加了一个永久的顶级按钮,显示一个相机图标,该图标会弹出一个对话框,允许用户选择栏上的单个窗口、单个组或所有窗口和组来添加快照。第二个顶层按钮(标有“列表...”)会启动快照浏览器,其中会显示现有快照及其属性的列表,以及所含窗口的图形预览。用户可以从此浏览器中选择“恢复”命令,重新定位或重新启动任何选定快照的窗口。

在 Scalable Fabric 中,我们并未尝试解决一般的持久化问题,而是简单地保存正在运行的窗口和任务的位置、大小和标题信息。每当创建或移动窗口,或者选择或更改任务时,这些信息都会更新。当 Scalable Fabric 重新启动时,如果发现具有相同标题的打开窗口,它将恢复到在 Scalable Fabric 中渲染的最后一个状态。如果窗口不存在,Scalable Fabric 不会尝试启动应用程序并恢复其运行状态。这种方法意味着 Scalable Fabric 可以被终止并重新启动,并且所有状态都将恢复。但是,如果用户注销或重启计算机,Scalable Fabric 将无法恢复状态,尽管任务标记将保留。显然,GroupBar 使用的快照或窗口持久化字符串在未来版本中也可以用于 Scalable Fabric。

Task Gallery 采用了一种中间方法,记录并存储用于启动应用程序的信息。这与 GroupBar 的方法类似,只是没有提供修改用于重启应用程序的持久性字符串的功能。

这些临时解决方案远非理想。一些 Windows 应用程序允许通过各种 COM 接口检查其打开的文档。也可以通过跟踪文件的打开、关闭和窗口的创建来实现这一点，但如果不能修改现有应用程序，这种方法就很难实现。此外，应用程序通常允许用户更改打开的文件，有些甚至在应用程序内部提供了一种复杂的窗口管理形式，这可能有助于持久化。如果没有某种标准的方法来获取应用程序中打开的文件和子窗口的状态，那么解决这个普遍问题将极其困难。在最初的 Rooms 系统（Henderson 和 Card, 1987）中，操作系统环境（基于 Lisp 的操作系统）提供了必要的应用程序和文档信息，允许系统在重新启动时将 Room 状态恢复到用户上次看到的状态，从而提供了一定程度的系统级持久化支持。然而，在单个窗口或任务级别，持久化问题可能比这更加复杂。理想情况下，一个健壮的任务管理持久化模型需要允许灵活地命名、持久化、修改和调用工作项的子集。它还需要支持不同的持久化粒度：例如，当用户尝试持久化一组打开的 Web 浏览器窗口时，所需的长期信息可能是这些窗口网页的当前内容，也可能仅仅是这些窗口指向的 URL，甚至可能是多个浏览器窗口的物理布局。最终，我们需要对操作系统进行修改，以允许应用程序显示和恢复其状态，并且编写应用程序时，使其状态可供任务管理系统检查和使用。

结论

随着显示器成本下降以及处理器和显卡性能的不断提升，多显示器系统或更大尺寸显示器的使用可能会增加。如果不了解人们通常如何与窗口交互和管理窗口，以及多显示器使用方式与过去有何不同，就很难理解如何（或者实际上是否）为即将到来的变革进行设计。我们在本章中呈现的结果的总体价值在于，它为进一步研究这些实践及其相似之处和差异提供了起点。我们

还提出了一些关于如何开始新颖的用户界面设计的想法,这些设计可以利用用户如何与跨不同显示尺寸的窗口进行交互。

在本章介绍的三种设计中,我们假设用户通过对窗口进行分组(可能还会对分组进行命名)来手动识别任务。许多用户和测试参与者都要求提供能够自动分组或至少提供分组建议的工具。我们已经开始研究监控用户活动并半自动地推导任务分组,例如,通过观察窗口交互随时间变化的聚类情况。然而,要使这项工作有效,仍有许多研究工作要做。

对于每一种设计(GroupBox、Scalable Fabric 和 Task Gallery),我们都讨论了任务切换支持如何相较于现有工具有所改进,以及每个设计中哪些方面可以进行迭代改进。综合来看,我们已经看到充足的证据表明,对快速重复任务切换的软件支持对桌面电脑信息工作者具有重要价值,现在是时候将这些研究理念融入到商业化产品中了。

致谢

我们衷心感谢 Maarten van Dantzich、Ken Hinckley、Kirsten Risdan、David Thiel 和 Vadim Gorokhovsky 对 Task Gallery 开发做出的贡献,以及 Dugald Hutching 对 Scalable Fabric 开发做出的贡献。

参考

Bannon, L., Cypher, A., Greenspan, S. 和 Monty, M. (1983). 用户活动组织的评估与分析。载于《CHI 83 会议纪要》, 第 54-57 页。
纽约: ACM。

Baudisch, P., Good, N. 和 Stewart, P. (2001). 焦点加上下文屏幕: 将显示技术与可视化技术相结合。载于 UIST 01 论文集, 第 31-40 页。纽约: ACM。

Bederson, B. 和 Hollan, J. (1994). Pad++: 用于探索替代界面物理的缩放图形界面。载于 UIST 1994 会议论文集, 第 17-26 页。
纽约: ACM。

Bell, B. 和 Feiner, S. (2000). 用户界面的动态空间管理。载于 UIST 00 会议论文集, 第 238-248 页。
纽约: ACM。

- Bly, S., and Rosenberg, J. (1986).平铺窗口和重叠窗口的比较。在CHI 86会议论文集,第 101-106 页。纽约:ACM。
- Bukowski, R. 和 Sequin, C. (1995)。对象关联:一种简单实用的虚拟 3D 操作方法。载于1995 年交互式 3D 图形研讨会论文集,第 131-138 页。纽约:ACM。
- Card, SK, 和 Henderson, AH, Jr. (1987).一种支持用户任务切换的多虚拟工作空间界面。载于1987 年 CHI+GI 会议论文集,第 53-59 页。纽约:ACM。
- Card, S.,Robertson, G. 和 York, W. (1996)。《WebBook 与 Web For-ager:万维网的信息工作空间》。载于CHI 96 会议论文集,第 111-117 页。纽约:ACM。
- Cutrell, E.,Czerwinski, M. 和 Horvitz, E. (2001)。通知、中断与记忆:消息中断对记忆和绩效的影响。载于《人机交互 Interact 01》,第 263-269 页。IOS 出版社。
- Czerwinski, M.,Cutrell, E. 和 Horvitz, E. (2000)。即时通讯与干扰:任务类型对绩效的影响。载于《OZCHI 2000 会议论文集》, Paris, C.,Ozkan, N.,Howard, S. 和 Lu, S. 编,第 356-361 页。澳大利亚悉尼,12 月 4-8 日。
- Czerwinski, M., and Horvitz, E. (2002)。日常计算事件的记忆。载于《人与计算机 XVI:2002 年人机交互会议论文集》, Faulkner, X.,Find-lay, J. 和 Detienne, F. 编,第 230-245 页。9 月 2-6 日。伦敦:Springer-Verlag。
- Czerwinski, M.,Horvitz, E. 和 Wilhite, S. (2004)。任务切换与中断的日志研究。载于CHI 04 会议论文集,第 175-182 页。纽约:ACM。
- Czerwinski, M.,van Dantzich, M.,Robertson, G. 和 Hoffman, H. (1999)。缩略图、鼠标悬停文本和空间位置记忆对 3D 网页检索的贡献。载于Interact 99 会议论文集,第 163-170 页。8 月 30 日至 9 月 3 日。伦敦:Springer-Verlag。
- Feiner, S.,MacIntyre, B.,Haupt, M. 和 Solomon, E. (1993)。《世界之窗:3D 增强现实的 2D 窗口》。载于UIST 1993 会议论文集,第 145-155 页。纽约:ACM。
- Franklin, N. 和 Tversky, B. (1990)。探索想象的环境。《实验心理学杂志:综合》 199: 63-76。
- Gillie, T. 和 Broadbent, D. (1989)。“是什么让干扰变得具有破坏性?一项关于长度、相似性和复杂性的研究”。《心理学研究》 50:243-250。
- Guimbretiere, F.,Stone, M. 和 Winograd, T. (2001)。流体与高分辨率墙面显示器的交互。载于UIST 01 论文集,第 21-30 页。纽约:ACM。
- Henderson, DA, Jr. 和 Card, SK (1987)。Rooms:在基于窗口的图形用户界面中使用多个虚拟工作空间来减少空间争用。ACM图形学学报 5 (3): 211-243。
- Hutchings, DR 和 Stasko, J. (2004)。重新审视展示空间管理:理解当前实践,为下一代设计提供参考。载于会议论文集

2004 年图形界面,安大略省伦敦,加拿大人机通信协会,第 127-134 页,5 月 17-19 日。

- Hutchings, DG, Smith, G., Meyers, B., Czerwinski, M. 和 Robertson, G. (2004)。单显示器和多显示器用户之间的显示空间使用情况和窗口管理操作比较。在 AVI 2004 论文集上,意大利加里波利。第 32-39 页,5 月 25-28 日。
- Kandogan, E. 和 Shneiderman, B. (1997)。弹性窗口:多窗口操作评估。载于 CHI '97 会议论文集,第 250-257 页。纽约:ACM。
- Mackinlay, J., Card, S. 和 Robertson, G. (1990)。通过虚拟 3D 工作空间实现快速控制移动。SIGGRAPH '90,第 171-176 页。纽约:ACM。
- Maglio, PP, Campbell, CS (2000)。显示外围信息的权衡。载于 CHI '00 会议论文集,第 241-248 页。纽约:ACM。
- Morris, J., Satyanarayanan, M., Conner, M., Howard, J., Rosenthal, D. 和 Smith, F. (1986)。Andrew:分布式个人计算环境。CACM 29 (3): 184-201。
- Myers, B. (1988)。窗口界面:窗口管理器用户界面的分类。IEEE 计算机图形学与应用 8(5): 65-84。
- Mynatt, E., Igarashi, T., Edwards, W. 和 LaMarca, A. (1999)。平面国:办公室白板的新维度。载于 CHI '99 会议论文集,第 346-353 页。纽约:ACM。
- Patton, BM (1990)。记忆艺术的历史。神经病学 40: 346-352。
- Pierce, J., Conway, M., van Dantzich, M. 和 Robertson, G. (1999)。工具空间和浏览:在 3D 桌面应用程序中存储、访问和检索对象。在 1999 年 4 月的《交互式 3D 图形研讨会论文集》中,第 163-163 页 168。纽约:ACM。
- Rekimoto, J. (1999)。时间机器计算:一种以时间为中的信息环境方法。载于 UIST '99 会议论文集,第 45-54 页,纽约:ACM。
- Ringel, M. (2003)。单方案不足:虚拟桌面使用策略分析及其设计启示。载于 CHI 2003 扩展摘要,第 762-763 页。纽约:ACM。
- Robertson, G., Card, S. 和 Mackinlay, J. (1993)。使用 3D 交互式动画实现信息可视化。CACM 36 (4): 57-71。
- Robertson, G., Czerwinski, M., Larson, K., Robbins, D., Thiel, D., 和 van Dantzich, M. (1998)。数据山:使用空间记忆进行文档管理。在 UIST '98 会议论文集,第 153-162 页。纽约:ACM。
- Robertson, G., van Dantzich, M., Robbins, D., Czerwinski, M., Hinckley, K., Risden, K., Thiel, D. 和 Gorokhovsky, V. (2000)。任务库:一个 3D 窗口管理器。载于 CHI '00 会议论文集,第 494-501 页。纽约:ACM。
- Siegel, A. 和 White, S. (1975)。大规模环境空间表征的发展。载于 H. Reese 主编,《儿童发展与行为进展》,第 10 卷,第 9-55 页。纽约:Academic Press。

138 乔治·罗伯逊等人

Smith, G., Baudisch, P., Robertson, G., Czerwinski, M., Meyers, B., Robbins, D. 和 Andrews, D. (2003). GroupBar:任务栏的演变。OZCHI 03 会议记录,澳大利亚布里斯班,11月26-28日。

Teitelman, W. (1986). 窗户系统十年回顾。收录于 Hop-good, F., Duce, D., Fielding, E., Robinson, K. 和 Williams, A. 编,《窗户管理方法论》。柏林:Springer-Verlag。

Wurnig, H. (1998).增强现实协作式多用户桌面系统的设计。载于《中欧计算机图形学研讨会论文集》,布拉迪斯拉发,布德梅里克,4月21-22日。

XDesk 软件 (2003)。关于虚拟桌面管理器。<http://www.virtual-desktop.info/>。

—

个人的社会维度

环境

第二部分简介

本书主要关注个人工作环境,而非群件。然而,对沟通和协作的支持是大多数贡献者关注的重点。这并不矛盾,因为创建个人工作环境对于成功协作的重要性丝毫不逊于创建共享工作环境。许多协作发生在协作者在其个人环境中工作时。第二部分的章节描述了创建“协作友好型”个人工作环境的新颖设计方法。

众所周知,我们的工作习惯和行为会根据我们在执行信息任务时所扮演的角色而改变。由于工作时间的增加、流动性的增强以及科技在我们生活中的普及,我们的个人活动、爱好、家庭沟通和正式工作正变得越来越紧密地交织在一起。Plaisant 和 Shneiderman 与 Baker、Duarte、Haririnia、Klinesmith、Lee、Velikovich、Wanga 和 Westhoff 合作撰写了关于个人角色管理的章节,试图设计和研究一个系统来支持我们数字生活中这一日益重要的方面。基于角色的任务管理理念被视为当今软件环境和工具设计中一种很有前景的方法。

Fisher 和 Nardi 对人与沟通的重视,为信息工作带来了耳目一新的视角。他们在章节中讨论的系统 ContactMap 和 Soylent 将人们视为计算环境中的“一等公民”,这一趋势正受到人们的重视,而这种关注也姗姗来迟。

这两章的相似之处在于,它们都运用了对个体工作社会背景的描述。然而,作者设计的系统中所体现的方法强调了

个人工作环境的社会维度的不同极点。

ContactMap 代表其他人（联系人）,而Personal Role Manager 则关注用户与其他人协作时的各种角色。看来,这些视角可以相互补充。这种可能性是第二部分讨论中提出的关于个人工作环境社会维度的众多有趣研究方向之一。

5

个人角色管理 :概述和 大学生电子邮件设计研究

Catherine Plaisant 和 Ben Shneiderman 与 H. Ross

Baker、Nicolas B. Duarte、Aydin Haririnia 和 Dawn E.

克莱恩史密斯、汉娜·李、列昂尼德·A·维利科维奇、阿尔弗雷德·O·旺加和马修·J·韦斯特霍夫

介绍

我们的日常活动丰富而复杂,因为我们在工作、家庭和社区中不断转换着各种角色。一位教授可能身兼数职,既是多门课程的教师,又是学生的导师、学术委员会成员、项目首席研究员、会议组织者、科学期刊的编辑,还是与业界的联络人。大多数职位描述都包含多项职责:即使是销售人员,也可能要处理不同类型的客户、培训新员工、管理公司拼车以及监督网站维护。工作需要兼顾个人角色,例如足球运动员或志愿消防员,以及家庭角色,例如为人父母、房屋装修工、家长教师协会成员或老年人的远程护理员。我们谈论“扮演不同的角色”,谈论我们在“其他生活”中所做的事情。这些语言暗示了这些角色的重要性和独特性。

不同的角色需要不同的心态、不同的压力、隐私和专业素养。这些帽子或许也象征着我们在不同角色中如何凸显不同的个性特征。尽管如此,我们仍然在相同的计算机环境中开展活动。某些应用程序或许可以自定义外观和行为以满足用户的需求和愿望,但当我们在这些通常截然不同的角色之间切换时,底层环境保持不变。设计师面临的问题是:如何设计兼顾这些不同角色的图形用户界面,从而提供更高效的操作?

有些用户试图通过为工作、家庭、爱好设立单独的电子邮件帐户甚至单独的计算机来创建不同的角色,

等等。我们将角色定义为个人持续（从一个月到数年）的工作，其中通常包含不同的人员、事件和文档。任务是个人的短期（从一小时到一周）工作，而项目则是一群人的持续工作。组织通常关注项目管理，因此他们强调个人之间的协调工具和关键路径技术，以加快团队工作的完成。由于我们关注的是使一个人能够管理组织内外的多个角色，因此我们强调文档管理、日历支持、沟通需求以及多个角色之间的注意力切换。

当前的图形用户界面基于文档、文件、文件夹和应用程序的物理桌面隐喻来操作它们。为了履行角色中的职责，用户必须思考一些低级操作，例如启动应用程序、打开文件、浏览目录结构以及搜索信息。然后，他们必须将结果保存为新文档并发送给其他人。此外，

尽管可以将文件保存到特定角色的文件夹中，但如今的图形用户界面并未考虑到以不同方式处理不同角色的需求。更糟糕的是，它们不支持快速角色切换。

一个典型的场景可能是这样的：约翰是一个大型基金的首席研究员。他已经花了一个小时写项目报告了。

文件浏览器和电子邮件工具现在会聚焦到正确的项目文件夹；文字处理器会显示正确的文档集；网页浏览器历史记录里现在充满了他刚刚浏览过的相关网页。他的窗口大小和布局也调整得更方便了。工作似乎进展顺利，但现在约翰需要切换到另一项任务。他的众多职责之一是主持一个研讨会，会议安排在五分钟后与印刷厂的代表进行电话会议。约翰首先浏览联系人列表，想找到印刷厂的名称，但由于列表太长，他没能找到。

他切换到文字处理器，打开可能包含打印机名称的规划文档。可惜的是，最近打开的文档列表都与撰写报告的首席研究员角色相关，所以他耐心地从“最近报告”目录向上浏览文件层次，然后向下浏览到“研讨会”目录。之后

他搜索了一下,找到了与打印机相关的文件名,并找到了要拨打的电话号码。但在开始通话之前,他切换到日历应用程序,紧张地翻阅着每周视图,试图回忆起之前设定的研讨会的照相准备截止日期和送至打印机的截止日期。最后,他打开网页浏览器,浏览长长的收藏夹列表,找到了研讨会的网页。电话会议结束后,他切换到为儿子预约医生,并给儿子的老师写信,请求允许儿子在午餐时间离开学校。他还回复了一个电话,要求立即更改他组织的节日派对的网络公告。之后,他返回到项目报告,但他不得不花几分钟重新打开窗口、调整窗口大小,并将每个应用程序重新导航到正确的文件夹。他意识到最近打开的文档列表现在毫无用处,而且默认文件夹被映射到了错误的位置。

相比之下,个人角色管理环境可以让 John 一步到位地从项目经理切换为研讨会组织者。他会发现他的环境专注于所选角色:文件浏览器会在相关的主目录中打开,最近打开的文件会根据他上次以该角色执行的任务进行排序,联系人列表会根据与该角色相关的联系人进行筛选(方便用户回忆起相关姓名),日历会突出显示选择该角色时输入的相关截止日期,关键的应用程序和文档可以立即保存和打开。

我们在 1994 年提出了个人角色管理策略,作为下一代图形用户界面的指导理念。第一代是命令行界面,要求用户了解计算机概念和语法。这些界面被第二代图形用户界面所取代,使用了桌面隐喻、图标和文件夹。接下来,第三代强调以文档为中心的方法,应用程序逐渐淡出背景,而多媒体文档成为关注的焦点。我们提出的第四代以用户为中心的设计强调用户的角色、协作者和任务,而不是文档。每个角色都涉及与……的协调

人群和在计划内完成的任务。

由于界面环境允许多任务处理,一些用户设法通过同时打开多个窗口来支持角色。通过运行允许多个桌面的窗口管理器(有时称为“房间”)(例如,Henderson 和 Card,1986; Robertson 等人,2000),可以模拟个人角色管理策略。

然而,这种方法并未解决组织文档、联系人、日历、网页收藏夹和最近文件等关键问题。在这样的多桌面环境中,关注某个角色意味着虚拟空间中位置的变化,但各个应用程序的行为保持不变,因为它们对其使用环境的变化视而不见。

与角色理论(Sarbin 和 Allen 1968;Biddle 和 Thomas 1979;Roos 和 Starke 1981)或计算机支持的协同工作(Singh 和 Rein 1992)的研究相比,个人角色管理主要关注组织内个人的协调,而个人角色管理则侧重于帮助个人管理他们的多重角色。

较新的框架,例如活动理论(Redmiles 2002),将工作视为由各种需求驱动的活动,人们通过这些活动寻求实现目标。活动理论的支持者为实现组织目标提供了有益的见解,但他们并未提供足够的框架来理解用户如何看待其在组织内外的多重角色。

早期探索

我们最初关于个人角色管理的研究基于一项针对世界银行员工的观察性研究。该研究考察了项目生命周期、文档管理、电子邮件实践、培训和软件工具的可用性,并发现了世界银行员工经常遇到的问题,这些问题在其他组织中似乎很普遍,也同样重要。其中一个问题是在组织内同时兼顾多个角色。管理人员通常同时监督多个项目,并且在其业务部门或团队中担任各种角色。例如,一名员工可能同时负责两个项目(阿尔及利亚的一家医疗诊所和马里的饮用水供应系统建设)、三个工作组的成员、杂志编辑以及组织……

节日聚会的参与者。需要大量的个人组织来管理这些目标、合作者、工具和文档大多不同的角色。

我们的探索最终形成了两个主要概念：个人角色管理和组织概览。组织概览旨在为数据库中搜索结果的呈现提供统一的背景，同时也能映射个人在组织中的多重个人角色，并充当可调整大小的控制面板，方便切换角色（图 5.1、5.2、5.3）。原型展示了个人角色管理

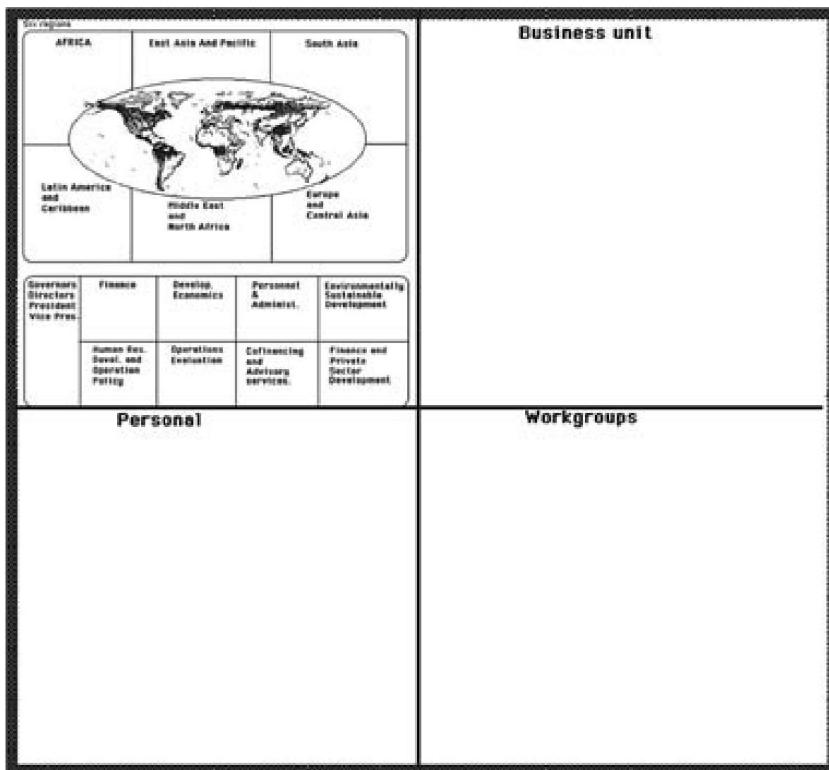


图5.1

银行新员工在被分配新工作之前，会看到这个角色概览。左上角是银行的主要组织结构。另外三个方框分别用于组织和聚合与业务部门相关的角色，以及用户所属工作组的个人角色。

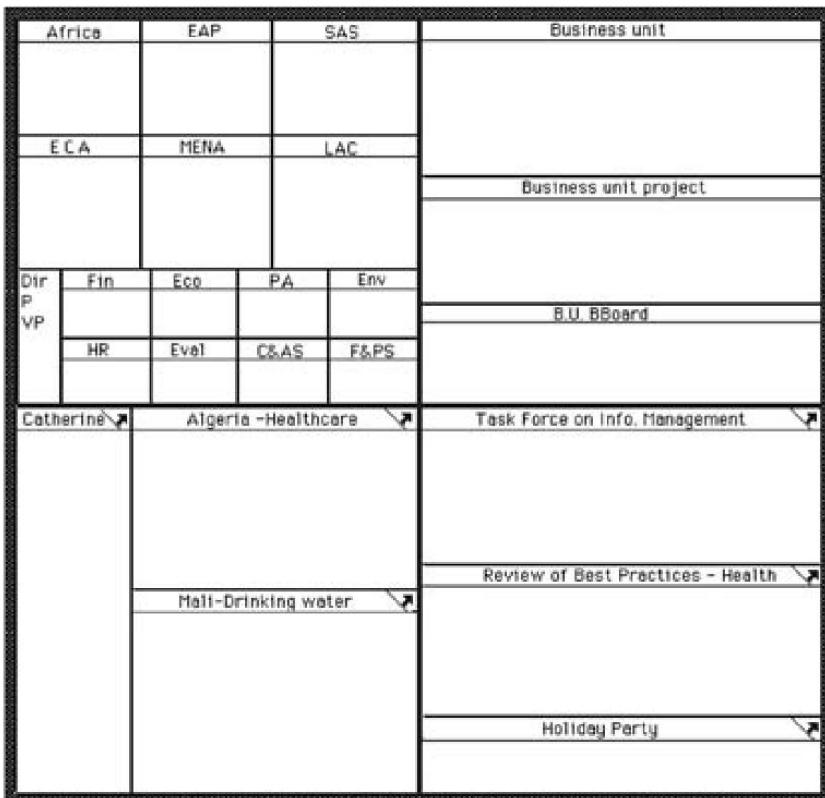


图5.2

当新员工被分配角色后,这些角色可以在概览中整理,供个人角色经理使用。此类角色概览还可以缩小为大图标大小,并始终显示,方便用户轻松切换角色。

键盘快捷键也可以加速角色切换。

这是一种允许知识型员工根据其在组织中的角色来组织信息的策略。角色被定义为具有愿景声明、一组人员、一个时间表和一个任务层级。愿景声明是由个人或上级制定的文件。它可以促进培训或责任移交。人员是一个联系人列表,经过精简,仅显示与该角色最相关的联系人。时间表专注于特定角色,并显示相关文件和工具。特定角色的进出或角色切换是即时且无缝的(图5.4)。
角色过度

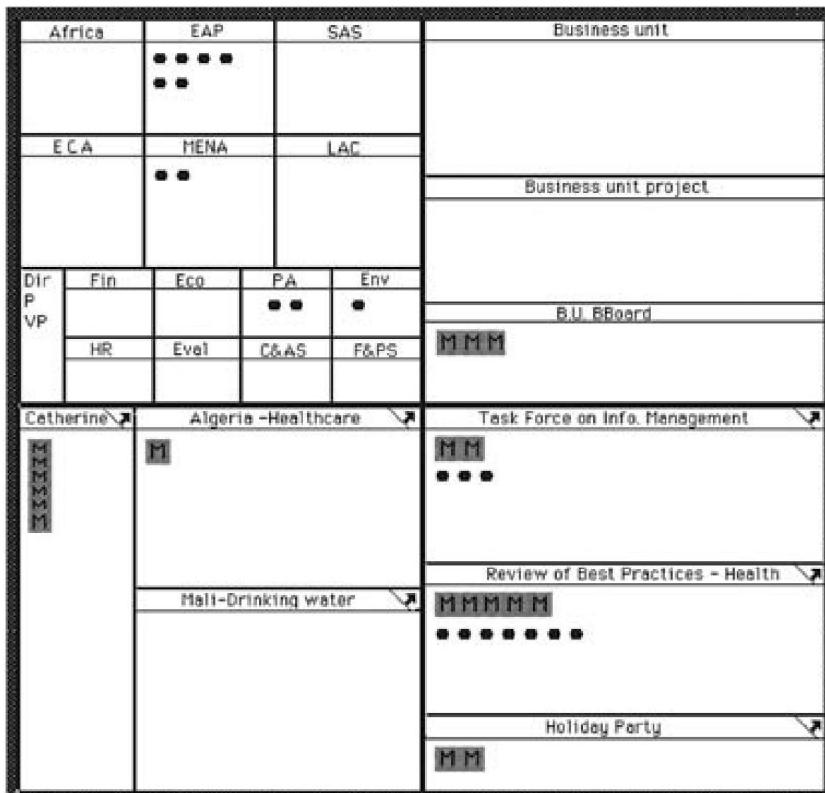


图5.3

在这里我们看到相同的组织概览,但简单的可视化技术总结了每个角色的未读电子邮件(M)和待办事项(*)的数量,提醒用户哪些角色今天可能需要更多关注。

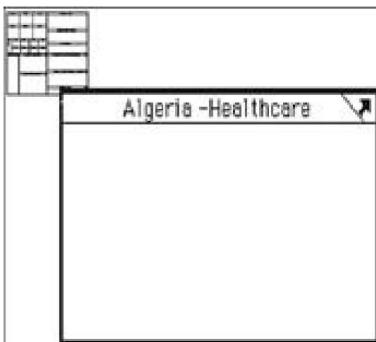
视图图标即使很小,也可以用来切换角色,并在需要时放大显示更多详细信息。当 John 接到有关其他角色的电话时,只需单击概览视图或使用键盘快捷键即可切换到该角色。

为了阐释个人角色管理策略,开发了一个低保真原型(Plaisant 和 Shneiderman 1995a)。该策略最初由 Ben Shneiderman 在 1994 年英国人机交互大会“人与计算机”主题上的主题演讲中提出(Shneiderman 和 Plaisant 1994),一年后,更详细的工作描述出现了(Plaisant 和 Shneiderman 1995b)。

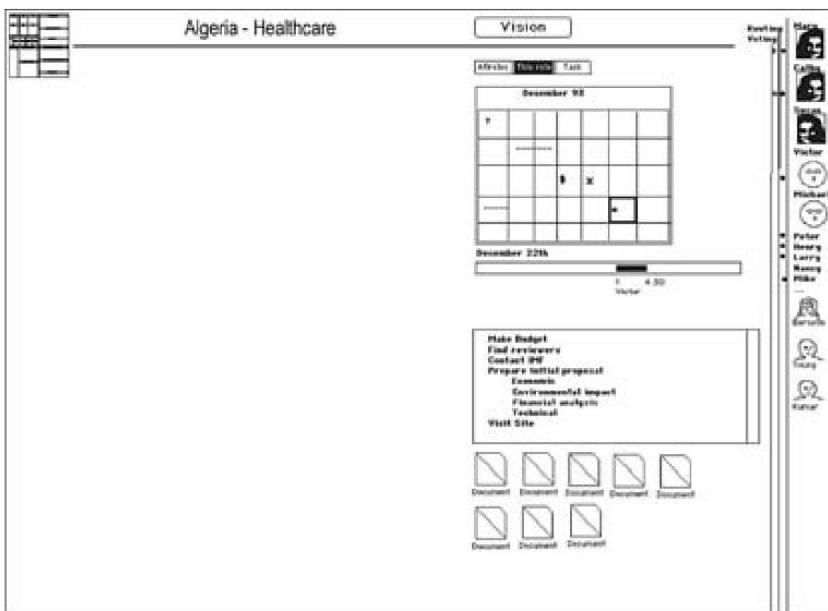
150 凯瑟琳·普莱森特和本·施奈德曼



(一)



(二)



(三)

图5.4

原型模型的三个动画步骤展示了 (a) 在角色概览 (左上角显示为一个大图标)中选择一个角色，然后打开所选角色 (b),该角色会填满整个屏幕 (c),并显示以该角色为中心的日历、联系人列表和文件层次结构。角色概览仍然显示在左上角，以便快速切换到其他角色。

个人角色管理也用于说明一个名为“弹性窗口”(Kandogan 和 Shneiderman, 1997)的窗口管理环境。层级布局 (图 5.5)通过窗口组织中的空间线索来指示窗口内容之间的层级关系。它为用户提供了所有角色的概览，以便他们可以选择任何角色或其中的部分角色并开始处理该角色。层级分组为信息组织提供了基于角色的上下文。

它还支持隐藏图形信息，因为窗口层次结构可以折叠为单个图标（或其他图元），从而使该方法具有可扩展性。折叠的窗口层次结构可以保存和检索，从而允许用户重用之前的窗口组织方式。

虽然大多数讨论者都赞同个人角色管理，但普遍的批评是，角色往往是相互关联的，而且它可以



图5.5

这是后来使用 Elastic Windows 原型实现的大学教授角色管理器的示例。这位教授是多位研究生的导师,负责多个研究项目(图中展示了三个近期项目和五个早期项目)。他本学期在大学教授两门课程(CMSC 434 和 828S),担任三家公司的行业联络员,并承担个人职责。

很难区分他们。有时用户会与同一个人合作多个项目,甚至会与他们一起组织社交活动或家庭度假。文档可能会在不同的环境中重复使用,日历也需要显示所有角色的所有事件,尤其是在用户安排新事件或只是计划一天时。管理角色可能会浪费比节省更多的时间,这是一个合理的风险。我们认识到角色管理可能并非对所有用户都有用,但尽管如此,我们相信许多用户拥有独特且稳定的角色(即拥有一组大致不同的合作者、文档和月、年或更长时间的日程安排)。因此,过滤界面环境以仅显示相关信息并允许用户将注意力集中在该角色上具有很大的优势。简短的联系人姓名列表可能会提醒用户要做的事情就像人们看到合作者时一样

走廊里的访客。日历的聚焦视图会提醒他们截止日期和计划,而文件系统的聚焦视图可以节省一些导航步骤。当然,个人角色管理器需要允许用户控制切换角色时聚焦的程度,在某些情况下,当出现新任务时,他们可能根本无法过滤或聚焦其环境。

在角色管理环境中,一种务实的方法是将角色定义为现实生活中角色的子集,这些角色足够独特,能够实现环境的自动化定制,从而带来益处。例如,在这种基于角色的环境中,一位大学教员作为研究员的多重角色可能不够独特,不足以被视为多重角色(因为同事可能参与多个项目,研究课题也存在重叠)。另一方面,会议主席、其他学院院长遴选委员会成员或小学家长教师协会成员等角色,很可能是截然不同的角色,很容易识别。

创建新角色也是一个挑战。有些角色可能继承自其他用户,或由组织提供。例如,一位教授被赋予一个教学角色,该角色包含一个时间表、一组学生和预设文档。一位休产假的新妈妈可能会将某个工作角色及其所有组成部分交给临时替代者。随着角色变得更加复杂,角色也可能会拆分,例如成立一个专门处理特定问题的小组委员会。或者,角色可能会合并,例如两个销售团队在一位经理的领导下重组。最后,只需启动一个角色并以该角色执行工作,即可快速创建新角色。例如,如果用户为自己创建一个拼车协调员的新角色,那么随着他发送电子邮件、创建文档或在日历中输入事件,该角色将变得更加明确。角色切换必须简单快捷,并提供有关当前角色的清晰反馈。

很自然地,我们会考虑利用组织设计人员提供的模板自动创建角色。教务长或系主任会根据注册数据、助教分配情况以及大学日历,向教授分配角色,这一点很容易理解。正如创建 Excel 或 Word 宏或模板已成为许多组织的一项专长一样,创建角色模板可以极大地促进个人角色管理方法的采用。

大学生基于角色的电子邮件案例研究

马里兰大学本科生团队最近重新审视了个人角色管理策略,研究如何改善大学生的电子邮件界面。虽然所有用户都会扮演多个角色,但大学生是一个有趣的例子,他们扮演的角色相当独特且可预测,至少在他们刚入学时是如此。他们的学校角色(或学生角色)是由课程、项目和考试的节奏和互动构成的。他们的家庭角色通常与学校无关;他们经常在校园外工作,扮演工作角色,与另一个独立的群体互动。

我们的学生团队进行了一项小型调查,旨在了解校园学生的电子邮件使用模式和主观体验。调查显示,电子邮件过载和功能限制是校园电子邮件沟通的主要障碍。

我们研究了电子邮件中的个人角色管理如何利用大学生电子邮件通信的分类特性。班级沟通中涉及的联系人、日程安排以及许多文档通常定义明确(例如,特定班级的学生和教授),这些信息可以提前获知并自动预设。这些知识使得电子邮件程序能够通过分组自动整理学生的许多消息和联系人。在必要的后端支持下,班级目录列表和日历的专用视图成为可能。我们描述了使用场景、界面模型和用户反应。我们的研究表明,将这些角色作为大学生电子邮件界面设计的驱动因素,或许能够解决我们在调查和访谈中发现的一些问题。

其他设置中的用户组也可能受益于基于角色的电子邮件界面,只要他们的某些角色足够独特,以允许某种程度的自动角色检测,并受益于针对不同角色的界面定制。

电子邮件相关工作

在对电子邮件超载的研究中,Whittaker 和 Sidner (1996) 发现人们使用电子邮件进行任务管理和个人存档。

他们将任务管理的目标描述为“[确保]信息

与当前任务相关的信息随时可用。”研究人员通过对 Lotus NotesMail 用户的一项研究得出结论,保持电子邮件井然有序对一些电子邮件用户来说是一个大问题,导致未读和未答复的邮件积压。

最近,Ducheneaut 和 Bellotti (2001) 进一步描述了电子邮件如何被广泛用作个人信息管理器 (PIM)。

通过访谈,他们调查了人们在工作中如何整理电子邮件以及如何在商业环境中处理杂乱的信息。研究人员建议:“为了更好地支持将电子邮件用作个人资讯管理工具,文件夹的组织方式应该更加灵活……应该支持电子邮件中的待办事项和提醒事项管理。” 访谈结果表明,现有的软件并未充分展现这些功能。

他们根据研究提出了以下问题:“在电子邮件客户端的设计中,是否有可能利用用户角色和组织环境的模型?一种可能的方法是,根据用户的角色,提供不同的界面和不同的电子邮件管理选项。”

Bellotti 等人 (2003)开发了一个以任务管理为中心的电子邮件客户端原型,并获得了测试过该客户端的企业用户的积极反馈。近期还有两篇论文探讨了按任务或活动组织电子邮件 (Kaptelinin 2003;Venolia 和 Neustaedter 2003)。选择任务管理而非角色管理可能更适合某些企业使用模式,在这些模式下,员工需要同时处理多项短期任务

所有这些都在一个角色中完成 (ESA&NTIA报告,2002)。然而,许多用户拥有多个不同的角色,并且经常将个人生活中与工作无关的部分融入到他们的电子邮件活动中 (Nippert-Eng, 1996),从而引入了不同的社交群体、事件和任务,这些应该与工作活动分开管理。我们的非正式调查显示,大学生经常承担多个不同的角色;因此,个人角色管理策略可能适合大学生。

由于我们的学生团队对大学生的生活有着第一手的了解,并且能够采访大量的朋友和同学,因此他们决定专注于这个特定的用户群体。在另一个领域,Barreau 和 Nardi (1995)论证了基于位置的保存和搜索的重要性,并表明用户对其信息空间的感知以及信息在该空间中的位置具有提醒功能。这与研究人员的观点形成了鲜明对比。

他们认为,用户只需要更好的工具,就能在仅按时间顺序组织的档案中查找文档(Fertig、Freeman 和 Gelernter 1996a,b)。另一个关键方向是使用基于计算机的工具,分析与特定个人的电子邮件交流频率,作为用户识别不同协作者群体的起点(Nardi 等人 2002;Fisher 和 Dourish 2004;Fisher 和 Nardi,本书)。ContactMap 和 Soylent 有效地关注了人,而角色管理器概念则将日程安排和文档作为角色的一部分。

美国商务部的调查显示,2002年美国普通民众的电子邮件使用率为45.2%,高于2000年的35.4% (ESA&NTIA, 2002)。大学生的使用率延续了这一趋势。皮尤研究中心“互联网与美国生活项目”2002年的一项研究表明,“与普通民众相比,大学生是互联网的重度用户……部分原因是他们从小就接触电脑。[互联网]已经融入他们的日常交流习惯,成为与电话或电视一样普遍的技术”(Jones,2002)。

本节其余部分总结了收集到的关于学生群体电子邮件使用情况的信息,并提供了一个界面模型,演示了如何在以角色为中心的电子邮件程序中实现“学生角色”。这项工作由马里兰大学由本科生组成的跨学科“Gemstone团队”1(本章最后八位作者)完成。

了解学生的需求

为了了解学生的关注点、偏好、态度和需求,开展了两项校园调查。第一份调查是一次规模较小的、面向全体学生的调查,于2001年11月在马里兰大学帕克分校的学生中进行。调查问卷在晚餐时间于宿舍餐厅外发放。调查显示,大学生每天都使用电子邮件进行交流。在接受调查的35名学生中,86%的学生每天查看数次电子邮件,100%的学生每天至少查看一次电子邮件。此外,89%的学生使用多个电子邮件地址收发电子邮件。

消息。

为了评估当前电子邮件软件在满足大学生需求方面的质量,研究人员要求学生列出他们经常使用的电子邮件功能。一些功能 (例如,发送附件、转发消息、删除消息)几乎所有学生都会使用,而其他功能 (例如,发送签名文件、发送自动回复消息)则只有少数学生使用。虽然有些功能与他们无关,但其他功能显然由于其复杂性和在电子邮件程序中缺乏可见性而未被使用。例如,100% 的受访者表示收到过垃圾邮件,43% 的学生使用过滤器来屏蔽不需要的邮件。同时,6% 的学生不确定过滤器是什么,40% 的学生认为过滤器功能应该改进,尤其是其易用性。

调查还探讨了电子邮件整理的话题。学生们被问及是否使用文件夹来分类和存储电子邮件。

80% 的受访学生使用文件夹;其中 75% 的学生文件夹数量少于 10 个。其余受访学生文件夹数量在 10 到 30 个之间。电子邮件整理对大学生来说至关重要,48% 的学生保存了超过一半收到的电子邮件,这证明了这一点。只有 21% 的学生保存的电子邮件不到十分之一。

学生们还被要求确定他们经常发送电子邮件的人。
正如预期的那样,学生使用电子邮件与朋友和家人沟通。63% 的学生也使用电子邮件与同事沟通 (图 5.6)。

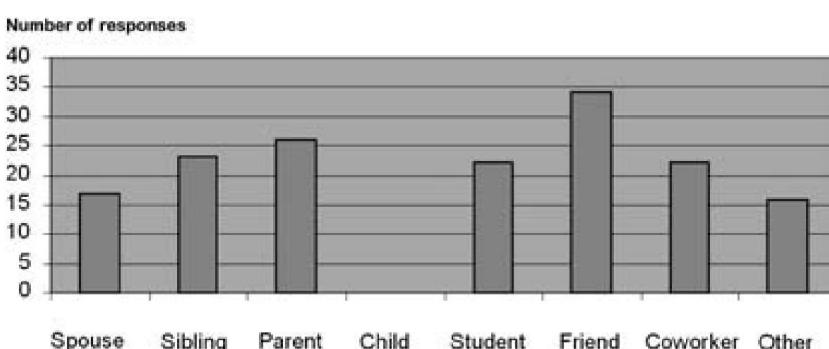


图5.6
三十五位调查受访者向其发送电子邮件的人员。

调查中有一节专门探讨了学生对当前电子邮件软件的评价。我们的学生样本使用了各种各样的电子邮件工具。学生们评论了他们喜欢和不喜欢的电子邮件功能，并经常提到以下积极功能：

- 简单
- 电子邮件通知
- 地址簿
- 文件夹
- 支持多个电子邮件地址

学生们还发现了当前电子邮件程序存在的问题。学生们承认的问题如下：

- 难以改变功能的工作方式
- 难以设置
- 缺乏拼写检查（在某些工具中）
- 功能过载

尽管样本量较小，但第一次调查提供了一些关于学生电子邮件使用情况的见解，并帮助我们开展了第二次调查。2002年4月，第二次调查分发给了47名学生，其中大多数仍然来自马里兰大学帕克分校。与第一次调查一样，第二次调查探讨了学生在使用当前电子邮件软件时遇到的问题。此外，第二次调查还考察了学生对用户界面潜在变化的态度。

为了调查过滤功能，研究人员询问学生是否理解并知道如何使用过滤器。9% 的学生承认他们不知道如何使用过滤器，32% 的学生对此只有模糊的概念。总体而言，学生对电子邮件自动化功能持开放态度。大多数学生（72%）表示，他们希望电子邮件能够自动分类。此外，66% 的学生对能够根据个人喜好进行调整的电子邮件程序充满热情。然而，大多数学生并不习惯使用能够追踪他们的使用模式并推断其意图（即使只是为了调整用户界面）的电子邮件程序。

系统模型

根据收到的反馈,学生作者着手独立设计一个用户界面,根据他们的观察解决大学生的两个主要问题:电子邮件超载和功能超载。

他们研究了个人角色管理如何通过两种方式解决这些问题。首先,按角色组织消息可以管理电子邮件过载。其次,选择当前角色的功能可以隐藏与该角色无关的功能,从而缓解部分功能过载。更精简的功能集也为特定角色的特殊功能留出了更多空间,例如按学期组织的自定义课程日历。

在设计角色管理用户界面时,学生们设定了简洁性标准。他们认为,界面应该足够熟悉,让现有电子邮件用户感到熟悉。理想情况下,新手在探索角色管理功能时,可以像使用现有电子邮件软件一样使用该程序。使用个人角色管理带来的额外开销应该最小化,以减少切换带来的损失。

最后,界面必须“优雅地”降级,并且在没有关于用户角色的信息或当

用户不愿意使用新功能。

在对纸质草图模型进行多次修改后,他们使用计算机图形工具生成了屏幕截图,以便更好地模拟实际界面。这些屏幕截图用于收集潜在用户的反馈。最后,他们实现了一个 Visual Basic 原型来演示部分交互。图 5.7 至 5.9 展示了原型的示例屏幕截图。

与标准电子邮件客户端最显著的区别在于角色选择选项卡的存在。每个角色都有一个单独的视图,由用户定义或由大学预设,其中仅显示与该角色相关的信息、联系人和功能。在图 5.7 的示例中,有两个特定角色可用 学校和工作 当前已选择学校角色。常规角色对应于电子邮件界面的“标准”入口,其中未选择任何角色。角色可以包含子角色;例如,每个班级(例如,ANTH 240、ENEE 435)都是一个子角色,可在学校角色中提供进一步的筛选。²

图 5.8 显示了学校角色的日历视图。学校日历以方便学校相关任务的方式显示数据,

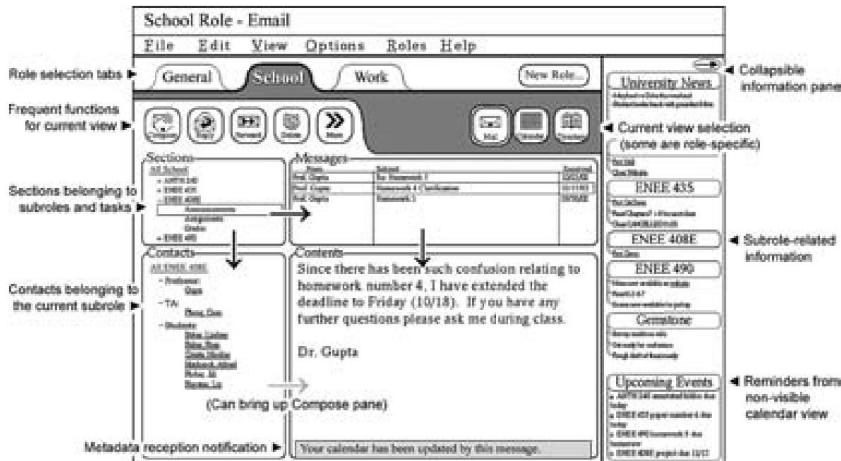


图5.7

邮件视图下学校角色中的角色管理电子邮件界面。屏幕截图中添加的箭头表示各部分之间的关联。图中,学生点击了ENEE 408E,仅显示与ENEE 408E课程相关的邮件。联系人列表也进行了筛选,仅显示已注册的学生和教授该课程的教师。邮件列表可以进一步筛选;此处仅显示公告。选择ENEE 408E角色后,点击“日历”即可切换到该课程的日历。

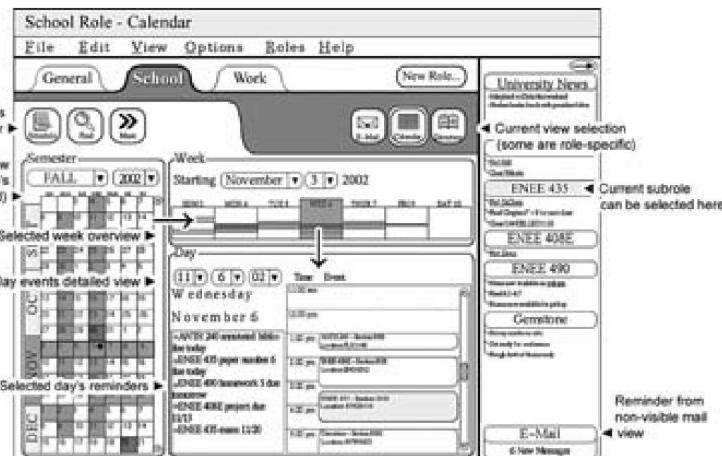


图5.8

学校角色中日历视图下的角色管理电子邮件界面。此处已选择ENEE 435课程(或子角色)。在黑白显示的完整学校角色日历上方,显示了与课程时长对应的学期日历,并以颜色编码表示课程时间、作业和考试。所有课程的每日活动也一并列出,但ENEE 435活动会突出显示。

例如,工作日历使用学期布局,而不是财务季度布局。日历可以使用不同的颜色来指示

上课时间、考试、作业截止日期等等。学校角色视图显示与该角色相关的所有事件,而关注特定班级(通过点击信息面板右侧的)则可查看该班级的时间表。

选择一个角色后,屏幕右侧的信息面板会汇总与该角色最相关的信息。例如

学校角色,它显示大学公告和课程列表。

对于工作角色,它显示一般公告和两个项目:“Circuit 项目”和“报告”(图 5.9)。对于上下文提示,不同的视觉主题或外观可以区分每个角色;在我们的原型中,这仅限于颜色,但可以包括字体、图标样式、音效等,以指示当前正在担任哪个角色。属于某个角色的子角色显示在屏幕右侧的角色信息面板中,并在层级浏览器中显示为文件夹。这

视图还允许用户在角色需要时创建传统文件夹。

信息面板的底部可以提供当前不可见的角色的摘要。图 5.8 显示它提醒用户有新邮件到达,可以在邮件视图中找到(点击

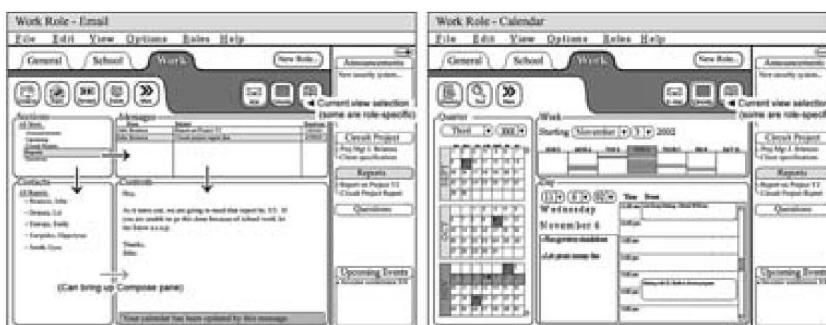


图5.9

工作角色中的角色管理电子邮件界面用于邮件(左)和日历视图(右)。工作角色的定义不如学校角色明确。用户仍然可以定义自定义日历(此处为季度日历)或其他选项,例如,不同的签名文件和自动拼写检查。

这些接触是有限的,并且与学校角色不同,并且在塑造角色本身的特征方面发挥着重要作用。

提醒器会切换到它）。每个角色也允许多个选项卡，但如果屏幕更大，所有信息都可以同时显示。

一般角色是独一无二的，因为它包含来自所有角色的消息和联系人，就像常规电子邮件应用程序一样，并允许用户在需要时从非基于角色的电子邮件转换回它。

一般角色的另一个目的是保存不适合任何定义角色的通信。

在考虑将个人角色管理作为一种组织方法时，会出现一些潜在的问题。最重要的是，角色并不总是相互排斥的。为了解决这个问题，可以将给定的消息或联系人设置为与其可能相关的任何角色可见。换句话说，角色充当筛选视图，用于查看可能与该角色相关的信息、联系人和事件。这允许角色重叠，例如，朋友发来的电子邮件请求家庭作业帮助，同时评论最近的社交活动。还有一个问题是，应用程序如何确定哪些消息和联系人适合哪些角色。一个潜在的解决方案是，我们调查中的许多学生使用多个电子邮件地址（他们通常有一个大学地址、一个以前的个人地址和一个工作地址）。这些不同的地址可以轻松地用于过滤不同角色的内容。一种限制较少的方法是，在用户将未标记的消息分配给角色（例如，将其拖动到角色选项卡或使用键盘快捷键）之前，不对其进行任何假设。新联系人将添加到角色中，之后与该联系人的所有后续通信都将分配给该角色，除非该线程被标记为其他角色。或者，如果用户从给定角色发起通信，则收件人将自动标记为该角色。在电子邮件标题中添加角色选择选项或许可以避免一些错误，但使用快捷方式提供快速无缝的角色切换更为实用。未关联的消息和联系人仍保留在常规角色中，该角色可以选择显示所有电子邮件或“仅限常规”电子邮件，以及所有人员和所有事件，并在日历中均匀显示。并非所有内容都能排序，但即使只有 20% 到 30% 的项目自动排序，也能节省大量时间并减少任务切换的开销。随着明确不同的角色数量的增加，其效益也将随之提升。

另一个问题是，如何提供有用的具体于角色的功能。某些功能可以在机构层面自动交付（例如，

使用基于学期的日历作为班级角色），并充分的用户控制可以进一步定制每个角色的界面。在机构

级别，大学可能会定义并向学生传播默认角色。

例如，学生报名参加课程后会收到一个新角色，其中包含课程大纲、教材信息、考试日期以及老师和同学名单；新当选的羽毛球俱乐部主席会收到一个新角色，其中包含截止日期日历、特定联系人列表和预算查看器。部分自动化功能可以通过在电子邮件中嵌入元数据来实现。

利用语义网研究人员（Hendler、Berners-Lee 和 Miller,2002）提出的理念，电子邮件客户端可以提供工具，用于将角色关联、新闻条目、事件变更或其他元信息嵌入到邮件中。收到此类邮件后，客户端可以可靠地读取并处理信息（并考虑必要的安全因素）。不了解元数据的电子邮件客户端则会忽略这些数据。元数据编码可以实现为图标工具箱，允许用户点击并拖动“表单”到邮件中。正确填写后，这些表单便可让收件人将日期、会议、联系信息或其他数据添加到其基于角色的电子邮件程序中。理想情况下，羽毛球俱乐部的前任主席可以通过电子邮件将他现已终止的职位告知新任主席。

对于高级用户，个性化角色定制将提升使用角色的优势。角色可以使用不同的签名（工作时使用正式签名，学校时使用非正式签名，亲朋好友使用家庭住址）。

自动拼写检查可以在工作角色中启用，但在朋友角色中则不启用，因为在朋友角色中沟通不太正式（学生们明确表示，在与朋友交谈时拼写检查很烦人）。可以选择不同的皮肤来匹配不同角色的心情。搜索

可以默认自动限制在角色信息范围内。可以在好友角色中关闭自动复制功能，在工作角色中打开存档功能。角色区分越多，时间就越长。

个人角色管理可以产生节省。

使用场景

考虑以下使用场景：Matt，一位典型的大学生，刚刚进入马里兰大学就读。他带着电脑来到学校，学校建议他下载并熟悉一个推荐的（基于角色的）电子邮件程序，该程序经过了量身定制。

给马里兰大学的学生。当他安装软件时,学校日历中已经填满了课程注册截止日期、大学假期、活动和最后一天的安排。当马特注册课程时,他会收到一条自动确认消息,其中包含课程的元数据信息。电子邮件程序会使用这些信息为马特设置学校角色。他的日历会更新(在他查看信息并确认日历自动加载后);联系人列表包含教师和助教的信息;课程大纲会保存在课程文件夹中。课程开始时,会收到一封提醒邮件,告知课堂变更,并加载同学的联系信息。

现在,Matt 在阅读电子邮件时可以选择一次性阅读所有邮件(使用“常规角色”选项卡),或者先关注他的学校角色,然后再查看其他邮件。阅读学校角色邮件时,他在信息面板中看到 ENEE 430 教授已突出显示即将到来的小组项目第一个截止日期。只需单击一下,他就可以切换到该班级子角色并查看班级日历。这很有用,因为他和大多数本科生一样,没有维护个人日历。他切换到电子邮件视图,但记不清要一起学习的同学的名字,于是浏览了大约二十名同学的列表。他认出了这个名字,并发送电子邮件安排了一次会议。

几个月后,马特在当地一家公司找到了一份兼职工作。起初,他所有与工作相关的电子邮件都出现在“常规”角色中。几周后,马特收到了公司许多人发来的电子邮件,他花了五分钟设置自己的工作角色。他将同事发来的消息拖放到工作角色上,将他们的名字添加到工作角色联系人列表中。几个月后,他已经从事两个项目,因此他在自己的工作角色下创建了两个子角色。一个月后,公司采用了基于角色的电子邮件系统,但由于马特即将毕业并辞职,他可以通过电子邮件将自己的角色信息(日历、联系人、选定的重要电子邮件、待办事项列表、报告)一次性发送给接替他的同学,从而将自己的角色转移给他。很快,他还将把自己的所有学校角色都存档在一起。

非正式用户反馈

我们进行了情景测试和访谈。2002年11月至12月,我们采访了来自马里兰大学帕克分校的20名学生。测试过程涉及印刷版

原型模型,旨在评估受试者对界面和角色管理概念的理解。测试前,记录了受试者的初步印象。之后,呈现了几个需要执行简单任务的场景。测试前未提供任何培训或演示。

研究人员鼓励受试者用语言表达他们的思考过程,并记录他们的言论。之后进行了一次后续访谈。

从最初的印象来看,许多受试者认为界面“过于繁琐”。这些受试者被问及他们会删除哪些信息,以及信息的组织情况如何。一些受试者认为信息面板并非总是有用,并认为它应该可折叠。日历的周视图和日视图似乎过于详细,因为许多学生很少使用日历记录个人信息。简化的日历被认为更有帮助。关于信息组织的反馈总体上是积极的,日历中的层级视图也获得了好评。一名学生评论说,它很容易让人专注于短期活动,而不会忽视长期目标。大多数受试者立刻就意识到了角色文件夹的用途;少数人最初将工作角色选项卡误认为是校园求职(实际上,这可能是尚未找到工作的学生的默认设置)。

在场景中,受试者很容易识别所需的界面功能,包括视图选择按钮、信息面板和联系人层级结构。当被要求查找下学期春假日期时,75% 的受试者正确选择了日历视图并操作了学期下拉菜单。从日历视图开始,受试者被要求给班主任发送电子邮件。60% 的受试者选择了最短路径,即使用信息面板中的教师电子邮件链接,而其余的受试者则倾向于切换到邮件视图并使用联系人列表。为了向班级中的每个人群发电子邮件,70% 的受试者做出了最佳选择,即点击邮件视图中联系人层级结构中班级的根节点。

后续访谈考察了该界面作为个人信息管理器的可行性。这个问题与目标受众息息相关,因为65%的受访者表示使用记事本或其他类型的日程安排程序。70%的受访者表示,他们会考虑使用类似本文介绍的程序来替代他们目前的日程安排程序;65%的受访者表示,他们会使用该程序查看他们的每日日程安排。尽管如此,

这些陈述可能无法预测实际使用情况,但它们表明总体反应积极。其余问题涉及自动化,学生仍然反对自动化变更:75% 的学生希望在课程表变更时收到通知并征求他们的同意。

教职员的反馈并不那么积极,因为越来越多的人担心系统能否按角色正确分类电子邮件。教职员通常承担大量角色划分不明确的任务,且同事之间也存在重叠。每个人的角色中,有些角色的划分足够清晰,可以准确识别,而有些角色的划分则比较困难。例如,教学角色或全校委员会成员的角色划分比较明确,但研究项目的角色重叠程度较高,可能需要将其归入一个大角色。

结论

我们关于校园电子邮件使用情况的研究结果对未来电子邮件客户端的设计者有几点启示:学生使用电子邮件的方式与普通商务用户不同,专门设计的界面将使其受益。虽然与学校相关的电子邮件使用量很大,但除了简单的短信功能外,其他功能很少使用,学生们用于整理电子邮件的时间也非常有限。



图5.10

新生踏入校园,准备开始新的生活,他们很可能会使用一些软件来帮助他们安排课程。根据他们的需求定制界面,并从一开始就提供个人信息管理功能,例如创建一个程序来收集和显示大学或课程安排,这很可能会提高使用率,并简化电子邮件和日历管理。大多数学生会保留相当一部分收到的邮件。许多学生希望邮件自动分类到文件夹中,但似乎不知道该如何操作,或者担心邮件会放错地方。个人角色管理策略或许有助于减少这些问题。

建议的界面展示了个人角色管理的一种实现方式,学生的初步反应非常积极。我们希望其他人能够继续开发电子邮件客户端的个人角色管理策略。开发一个功能齐全的原型将是评估此方法实用性的下一步。

其他用户群体也可能受益于基于角色的界面,前提是他们的角色足够独特,能够实现足够程度的自动角色检测,或者能够从针对不同角色的界面定制中受益。当用户切换到专注且独特的角色时,个人角色管理的优势将最大,因为它能够让他们快速关注与每个角色相关的信息。个人角色管理本身并不能解决所有信息或电子邮件过载的问题,但它有可能以一种对用户有意义且反映他们丰富多彩生活方式组织界面环境。³

致谢

我们感谢美国国家科学基金会信息技术研究基金 0086143 的部分支持,该基金旨在理解互联网的社会影响:一种多方面的多学科方法。

另外感谢 Harry Hochheiser 和 Rina Levy 过去帮助我们完善对个人角色管理的理解。

笔记

1. 马里兰大学的宝石项目注重学生在标准课堂环境之外的发展,挑战学生在研究、团队合作、沟通和

领导能力。团队成员来自土木工程、生物化学、电气工程、生理学与神经生物学、德语和计算机科学等专业。在导师 (Catherine Plaisant) 的指导下,他们每周会面一次,持续了三年。学生们主要独立进行研究。他们进行调查、设计原型、收集用户反馈,并撰写了毕业论文,本文对此进行了总结。

2. 学生们提出了子角色的概念。最初的个人角色管理方案明确避免使用子角色,因为会增加复杂性。

3. 关于学生作者:H. Ross Baker 是一位对语言学感兴趣的计算机科学家,现为西北大学语言学系的研究生。Nicolas Duarte 是一位对纳米技术感兴趣的电气工程师,现为宾夕法尼亚州立大学电气工程系攻读博士学位的研究生,研究纳米线的热传输。Aydin Haririnia 是一位对蛋白质核磁共振和晶体学感兴趣的生物化学家,现为马里兰大学化学与生物化学系的研究生。Dawn Kline-smith 是一位对结构工程感兴趣的土木工程师,现为加州大学伯克利分校土木与环境工程系的博士生。Hannah Lee 是马里兰大学的应届毕业生,对生理学和神经生物学感兴趣。Leonid Velikovich 是一位对计算机图形学感兴趣的计算机科学家,他最近毕业于马里兰大学计算机科学系。Alfred Wanga 是一位电气工程师,对电子材料和器件感兴趣。他现在是宾夕法尼亚州立大学的研究生。Matt Westhoff 是一位计算机科学家,对计算机图形学感兴趣。他最近毕业于马里兰大学帕克分校。

参考

Barreau, DK (1995). 情境作为个人信息管理系统的一个因素。《美国信息科学学会杂志》46 (5): 327–339。

Barreau, DK 和 Nardi, BA (1995). 查找与提醒:桌面文件整理。SIGCHI通讯27 (3): 39–43。

Bellotti, V., Ducheneaut, N., Howard, M. 和 Smith, I. (2003). 将电子邮件转化为任务:以任务管理为中心的电子邮件工具的设计与评估。载于《CHI 2003 会议论文集:计算系统中的人为因素》,

第 345–352 页。佛罗里达州劳德代尔堡,4 月 5 日至 10 日。

Biddle, BJ 和 Thomas, EJ (1979). 角色理论:概念与研究。
纽约:Krieger Publishing。

Ducheneaut, N. 和 Bellotti, V. (2001). 电子邮件作为栖息地:嵌入式个人信息管理的探索。《交互》8 (5): 30–38。

美国商务部 ESA&NTIA (2002)。经济和统计管理局以及国家电信和信息管理局

tion.一个网络国家:美国人如何扩大互联网的使用。

华盛顿特区 [Http://www.esa.doc.gov/508/esa/ANationOnlineEXSFeb02.htm/](http://www.esa.doc.gov/508/esa/ANationOnlineEXSFeb02.htm)。

Fertig, S., Freeman, E., 和 Gelernter, D. (1996a). “发现与提醒”的再思考。ACM SIGCHI 通讯 28 (1): 66–69。

Fertig, S., Freeman, E. 和 Gelernter, D. (1996b)。《生命流:桌面隐喻的另一种选择》。载于CHI 96 会议配套论文集:计算系统中的人为因素,第 410-111 页。加拿大温哥华,4 月 13-18 日。

Fisher, D. 和 Dourish, P. (2004)。日常协作中的社会与时间结构。载于CHI 2004 会议论文集《计算系统中的人为因素》。

奥地利维也纳,4 月 24 日至 29 日。

Henderson, DA, 和 Card, S. (1986)。房间:使用多个虚拟工作空间来减少基于窗口的图形用户界面中的空间争用。

ACM 图形学学报 5 (3): 211–243。

Hendler, J., Berners-Lee, T. 和 Miller, E. (2002)。语义网应用集成 [英文版]。日本电气工程师学会志 122 (10): 676–680。

Jones, S. (2002)。《互联网走进大学:学生如何利用当今科技面向未来生活》。华盛顿特区:皮尤互联网与美国生活项目。<http://usinfo.state.gov/usa/t091602.htm/>。

Kandogan, E. 和 Shneiderman, B. (1997)。弹性窗口:多窗口操作评估。CHI 97会议论文集:计算系统中的人为因素,第 250-257 页。佐治亚州亚特兰大,3 月 22 日至 27 日。

Kaptelinin, V. (2003)。UMEA:将交互历史转化为项目情境。载于CHI 2003会议论文集:计算系统中的人为因素,第 353-360 页。佛罗里达州劳德代尔堡,4 月 5 日至 10 日。

Nardi, B. 和 Barreau, D. (1997)。“查找与提醒”再探:桌面文件整理的恰当隐喻。SIGCHI 通讯 29 (1): 76–78。

Nardi, B., Whittaker, S., Isaacs, E., Creech, M., Johnson, J. 和 Hainsworth, J. (2002 年)。ContactMap:通过可视化个人社交网络整合沟通与信息。ACM 通讯 49 4 (4 月): 89–95。

Nippert-Eng, C. (1996)。家庭与工作:在日常生活中协商界限。芝加哥:芝加哥大学出版社。

Plaisant, C. 和 Shneiderman, B. (1995a)。组织概览与角色管理:未来桌面环境的启示。载于 IEEE 第四届使能技术研讨会论文集:协作企业的基础设施,第 14-22 页。西弗吉尼亚州伯克利斯普林斯,4 月 20-22 日。

Plaisant, C. 和 Shneiderman, B. (1995b)。组织概览与角色管理:未来桌面环境的启示。收录于 CHI 95 会议视频论文集:计算系统中的人为因素。科罗拉多州丹佛市,5 月 7-11 日。亦收录于 1994 年 HCIL 开放日视频,<http://www.cs.umd.edu/hcil/pubs/video-reports.shtml/>。

[cs.umd.edu/hcil/pubs/video-reports.shtml/](http://www.cs.umd.edu/hcil/pubs/video-reports.shtml/)。

170 凯瑟琳·普莱森特和本·施奈德曼

Redmiles, D. (2002). 活动理论与设计实践特刊导言。计算机支持的协同工作11 (1-2): 1-11。

Robertson, G., van Dantzich, M., Robbins, D., Czerwinski, M., Hinckley, K., Risden, K., Thiel, D. 和 Gorokhovsky, V. (2000). 任务库:一个 3D 窗口管理器。载于 SIGCHI 计算机系统人为因素会议论文集,第 494-501 页。荷兰海牙,4 月 1 日至 6 日。

Roos, LL, Jr. 和 Starke, FA (1981)。组织角色。收录于 Nystrom, PC 和 Starbuck, WH 编, 《组织设计手册》第一卷:使组织适应环境。牛津:牛津大学出版社。

Sarbin, TR 和 Allen, VL (1968)。角色理论。载于 Lindzey, G. 和 Aronson, E. 编, 《社会心理学手册》第二版。纽约:Addison-Wesley 出版社。

Shneiderman, B. 和 Plaisant, C. (1994)。图形用户界面的未来:个人角色管理器。载于《人与计算机》第九卷,第3-8页。苏格兰格拉斯哥,8月23-25日。

Singh, B. 和 Rein, G. (1992)。角色交互网络 (RINS) :一种过程描述形式主义。技术报告 CT-083-92。德克萨斯州奥斯汀微电子与计算中心。

Venolia, GD 和 Neustaedter, C. (2003)。理解电子邮件对话中的顺序和回复关系:混合模型可视化。载于 CHI 2003 会议论文集:计算系统中的人为因素,第 361-368 页。

佛罗里达州劳德代尔堡,4 月 5 日至 10 日。

Whittaker, S. 和 Sidner, C. (1996)。电子邮件超载:探索电子邮件中的个人信息管理。载于 CHI 1996 会议论文集:计算系统中的人为因素,第 276-283 页。加拿大温哥华,4 月 13-18 日。

纽约:ACM 出版社。

6

Soylent 和 ContactMap:工具 构建社会工作环境

丹尼尔·费舍尔和邦妮·纳尔迪

引言:探索和阐述社会工作景观

在计算机系统中,人们可以身处任何地方,也应该无处不在。我们相信,在计算机系统中呈现有意义的社交临场感,将使我们在日常工作环境中与计算机以及其他人进行更有意义、更有趣的互动。

知识型员工的工作很少是独自完成的。Nardi、Whittaker 和 Schwarz (2002)采访了员工,了解他们如何在工作中管理和与人互动。研究发现,员工们会谨慎地管理自己的社交网络,始终关注与他人互动的方式。人们管理着三项人际任务:建立社交网络、维护社交网络以及根据需要激活网络中的节点。

虽然用户可以使用通信和联系人管理工具来管理这些任务,但许多任务仍然难以完成。例如,三窗格邮件程序在提供上下文信息(例如有关消息发送者的通信历史记录)方面表现不佳。消息附件经常丢失,或与其所附加的消息分离。即时消息(IM)记录、邮件历史记录和其他形式的交互都是单独存储和呈现的。

用户试图绕过现有工具的问题。研究表明,电子邮件不仅用于异步通信,还用于任务管理和个人存档(Whittaker 和 Sidner 1996;Ducheneaut 和 Bellotti 2001)。

例如,Ducheneaut 和 Bellotti 发现,用户会把旧邮件保存为通讯录。他们把收件箱当作待办事项列表、备忘录,以及

未来的日历。他们使用电子邮件档案作为版本控制系统。

他们模糊了个人技术和协作技术之间的界限,将一个可用的应用程序重新利用起来,将持久信息与人联系起来,形成以人为本的个人数据库和通信中心。

在重新利用电子邮件的过程中,用户找到了将计算机系统中的工作对象(联系人、文件和信息)置于社交情境中的方法。换句话说,他们构建了一个社交工作环境。他们存档的电子邮件和维护的联系人构建了一个情境,使他们能够追踪并连接联系人与其信息之间的关系。

然而,这种临时的、缺乏原则的解决方案并不能令人满意。将附件留在邮件中意味着失去其文件系统功能(例如文件夹排序和搜索);将日期存储在收件箱中缺乏基于日历的实用界面。无论用户想出多少变通方案,当前的技术都无法完全支持社交工作环境。

本文探讨了在计算机系统中支持社会工作场景的意义。我们讨论了将计算机系统中的人员和关系表示为一级实体的想法:可分组、可选的对象,链接到其他资源,并连接到底层数据源。我们设想了一个围绕人员、活动和人工制品之间无处不在的连接而设计的系统。

这些无处不在的连接很有用,可能会被各种各样的人看到
频繁的信息和网络任务。用户可能会:

- 查找与特定人相关的文件。
- 获取与特定文件或任务相关的人员的当前位置或旅行信息,或查找与该任务相关的最有空的人员。

- 找出谁向他们发送了与特定主题相关的信息或请求。

- 收集与特定文件相关的所有信件和电子邮件文件。

- 查找个人或群体的交流历史。

我们将协助完成这些任务的系统称为以人为本;它是一个以人为本的系统。

我们对工作环境的愿景与Plaisant等人（本书）的愿景相似；他们提出的“角色管理”概念将与文档和系统的交互置于特定角色中。相比之下，我们描述的两个系统都摆脱了指定角色的管理任务，而是专注于理解我们网络的社会结构。

利用个人社交网络构建工作环境

这种系统的一个关键方面是人与人之间的关系。

有些系统会追踪大量的档案信息（Dumais 等人,2003），但并不试图追踪人与人之间的联系。我们使用“个人社交网络”的概念来组织这些联系。通过追踪网络中的联系人群组，我们可以将这些群组与其所处的社会和技术环境联系起来。

个人社交网络不同于最近流行的公共社交网络，例如 Friendster (boyd 2004) 支持的公共社交网络。公共网络系统试图通过将朋友的朋友彼此联系起来，连接互不相识的人们。个人社交网络代表单个用户对其所联系的人之间的联系的视角。本章将介绍两种不同的收集、跟踪和解读个人社交网络的系统。ContactMap 采用自上而下的方法，从用户界面开始，逐步深入到系统设计。Soylent 采用自下而上的方法，构建一个通用的基础设施来存储和显示人们周围的社交环境。随后，本章讨论了一系列将指导未来以人为本的计算机系统设计的理念。

ContactMap：自上而下的方法

ContactMap 最初基于这样的洞察：个人社交网络是当今经济中的关键资源（Nardi、Whittaker 和 Schwarz 2002）。

ContactMap 根据用户个人社交网络中的人员来组织计算机桌面（Nardi 等人,2002）。它通过显示用户社交网络中的联系人并提供与这些联系人相关的功能来实现这一点。联系人可以是用户熟悉并希望与其联系的个人或群组。每个

联系人有一个图标：联系人的照片，或其他助记图像。

174 丹尼尔·费舍尔和邦妮·纳尔迪

联系人可以聚类到一起形成群组;每个联系人可以不属于任何群组,也可以属于一个或多个群组,如图 6.1 所示。

ContactMap 将沟通和信息管理集成到一个用户界面中。点击每个联系人即可访问与其相关的信息或与之进行沟通。让我们

假设 Sally 是我们的用户, Sam 是 Sally 的 ContactMap 中的联系人之一。

在典型的场景中,Sally 点击 Sam 即可查看他发给她的电子邮件列表。Sam 的联系信息会显示提醒和通知。

与他相关的未读邮件。她读了他发来的最后几条消息,然后想给他打电话。她点击了他的图标,并使用 ContactMap 的点击拨号功能拨打电话。通话结束后,Sally 想起了一件她忘记的事情,于是她点击 Sam 发送了一封邮件。ContactMap 打开了一封发给 Sam 的新邮件。Sally 的工作变得非常轻松 无需查找电话号码或电子邮件地址,也无需启动其他应用程序。Sally 只会看到 Sam 发来的电子邮件,无需在文件夹中搜索。

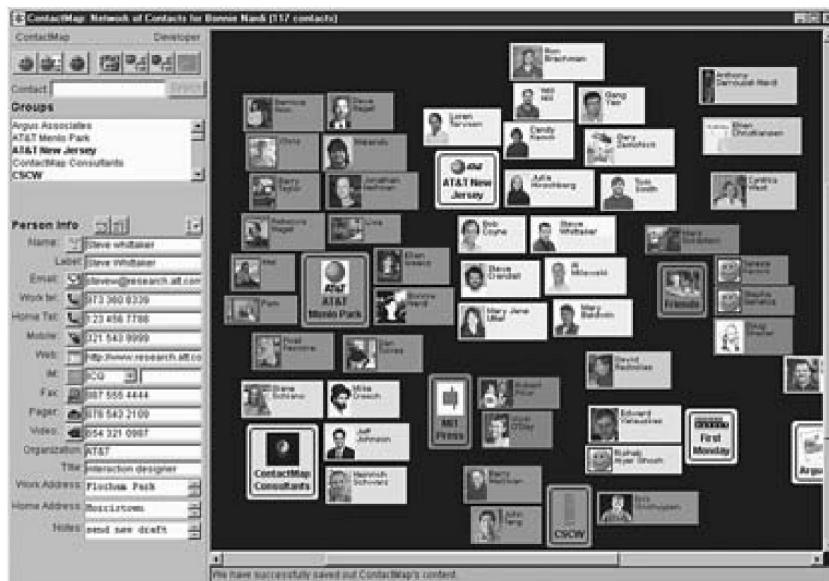


图6.1
联系地图。

ContactMap 帮助人们管理现代经济中与不同个人和团体进行的多任务工作（Nardi、Whittaker 和 Schwarz 2002）。

在该工具中，Sally 可以找到 Sam 发送给她的所有文档，因为这些文档都按联系人进行了索引。Sally 可以与 Sam 进行视频会议或即时通讯；她可以给他发传真，也可以访问他的网站。

ContactMap 的任何功能都可以针对群组而非个人执行。例如，可以发起电话会议、发送群发邮件、链接到网站等等。由于社交网络中的个人通常在用户的生活中扮演多个角色，因此 ContactMap 可以将单个联系人放入多个群组中。Sam 可能是 Sally 的同事，也是她园艺俱乐部的成员。

实证研究表明，大多数用户的社交图谱规模较小，平均有 95 人（Whittaker 等人，2004）。虽然图谱规模会随着时间的推移而增长，但人们通常只保留一小部分活跃联系人。这使我们能够摆脱多层次的层级设计：虽然存在人群，但没有人群的群组。复杂的联系人组织方式没有必要，而且会让许多用户感到困惑。随着活跃联系人在用户生活中的出现和消失，用户可以将联系人缩小为小图标，或在不需要时将其放置在屏幕外。联系人信息会被保留，但无需显示，以免造成显示混乱。

ContactMap 是一个社交工作环境，它重新整理了用户日常工作中最常用的操作，以反映用户与之互动的人员。它不会取代操作系统的功能，而是为该功能提供不同的用户界面。ContactMap 并非将文件和文件夹置于特权地位，而是以用户个人社交网络中的人员为中心。

ContactMap 的设置始于对用户电子邮件文件夹进行数值分析。ContactMap 会根据域名、联系频率和回复消息的频率，向用户呈现联系人列表（Nardi et al. 2002）。用户选择要添加到联系人图中的联系人，并根据需要进行分组。分组可以使用颜色编码。ContactMap 提供默认配色方案，用户也可以选择任何所需的颜色。

虽然目前尚未实现，但未来版本可以将联系人列表与地址簿、通话记录和其他数字资源连接起来。基于 Web 的更新可以处理联系人信息更新的繁琐工作。

信息。联系人信息可以像即时通讯一样在工作组或“好友”之间选择性地共享。

ContactMap 已由 15 位用户测试,其中包括研究人员、经理、行政助理和市场营销人员(Whittaker 等人,2004)。测试显示,所选联系人的平均数量为 95,范围从 15 到 184。即使只有 184 个联系人,在全屏图标显示下也相当容易管理。

用户将联系人分组,平均分组数为 11 个,范围为 2 到 23 个。在 ContactMap 的初始设计阶段,自动分组似乎是一个好主意,但经过几次失败的实验后,最终决定允许用户自行创建分组。由于联系人数量较少,分组任务变得非常简单,用户甚至似乎乐于在分组时回顾自己的社交网络。分组的平均规模为 8 个,几乎所有联系人都出现在分组中。只有 7% 的联系人是“单独联系人”。单个联系人可以出现在多个组中。

在整个测试人群中,群体的性质出奇地一致:工作组、工作项目、朋友、家人和特殊兴趣,在我们的样本中包括家长教师协会、摇滚乐队和股票俱乐部。

关于面对面互动在日常交流中的重要性的研究(Nardi 和 Whittaker,2002)表明,如果能够方便地使用联系人的照片,将会令人愉悦。用户只需找到一张数码照片或图像,ContactMap 就会调整其大小并将其放置在地图上。这项功能在用户测试中广受欢迎。

需要进一步测试才能更深入地了解此问题以及 ContactMap 典型用户的其他方面。目前,Contact-Map 仍处于原型阶段,但已停止开发。

Soylent:自下而上的方法

Soylent 构成了一套基础设施的基础,可用于构建类似 ContactMap 的应用程序。其名称虽然带有双关语,但实际上却表达了其目标:让计算机系统“由人构成”。

它包括创建、存储和访问个人网络的工具、可视化和与这些网络交互的方式以及将这些网络连接到应用程序的初步工具。

Soylent 的开发部分是为了回应 Net-WORK 和 ContactMap 研究中提出的问题(Nardi、Whittaker 和 Schwarz 2002;

Nardi 等人,2002 年)。ContactMap 的初衷是通过工具解决当前需求,而 Soylent 的设计则更侧重于探索社交信息基础设施的组装和设计方式。它利用电子邮件通信信息构建社交网络,并追踪互动的时间范围以及隐含参与互动的群体。

在本节中,我们讨论 Soylent 系统的构建,并描述在 Soylent 基础设施上构建的原型:一系列电子邮件程序的社交扩展。

Soylent 的数据从电子邮件存档中收集,并整理成互动历史记录。历史记录按邮件发件人、日期和时间、收件人以及附件存储和索引邮件。这些数据被用作基于邮件头内共现的网络的基础。

事实上,如果用户向两个人发送了一封电子邮件(共用一个“收件人”行、抄送双方,或者两者兼而有之),则可以证明从该用户的角度来看,这两个人有一些共同之处。

因此,通过检查传出消息中的聚类信息,Soylent 开发了用户工作环境的概念。

利用社交网络指导设计

Fisher 和 Dourish (2004) 探讨了从整体来看,这个庞大的个人电子邮件档案的一些主要特征。本文,我们将探讨如何解读图中较小的子集,以理解个人互动以及人们周围的直接环境。

分析网络电子邮件的网络可视化(例如 Eve-land 和 Bikson 1988;Tyler、Wilkinson 和 Huberman 2003)传统上会检查由谁向谁发送电子邮件而联系在一起的姓名对。这些技术提供了电子邮件记录的集体和全局视图,并使用“收件人-收件人”方法进行分析,绘制发件人和收件人之间的定向链接。这些社交网络用于将许多人联系在一起,并提供人们如何联系的广阔视角。相比之下,我们的系统旨在帮助理解单个用户的工作空间。在 Soylent 中,与 Boyd (2002) 的研究一样,一条同时发送给两个不同人的消息,无论是通过“收件人”、“抄送”还是“密送”,都被理解为隐性地将这些人联系在一起;发件人认为他们对该消息感兴趣。Soylent

因此,网络图将这两个人联系在一起。图 6.2 示意性地显示了这一机制。

这些图表的解读侧重于考察单个通讯员的视角。每张图表都将考察用户、通讯员以及通讯员周围人群(即用户向其发送消息的人群)之间的社交互动。这些以人を中心的小型图表展现了接收者周围的社交背景。例如,在图 6.2 中,假设“Z”是通讯员。在这种情况下,Z 与两个群体相连:一个由 Y 和 Z 组成,另一个由 V,W 和 Z 组成。

Soylent 利用这个网络,为单个收件人提供“个人雷达”视图。给定一个具体的名宇,这个雷达视图可以给出一个总体信息。

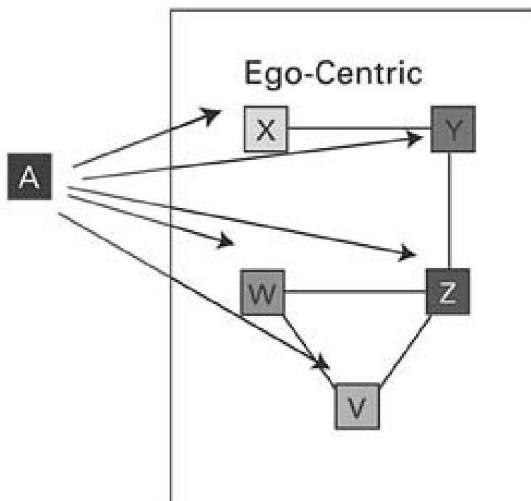
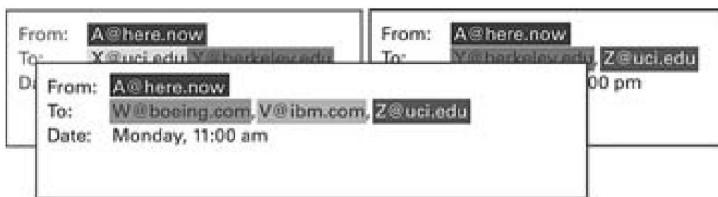


图6.2
Soylent “以自我为中心”可视化的架构。

与人的互动历史的视图:他们与谁有联系,这些联系最近发生的时间,以及发生了哪些互动。

举一个简单的例子,请考虑图 6.3 中的显示。用户是一位教授,而通讯员(中间)是他研究小组(左)的前成员。边按新近度编码;较近的通信用较深的颜色绘制。教授最近举办了一个研讨会(右);学生加入了该研讨会。虽然研究小组现已解散(灰色),但工作组仍在继续。学生位于这两个小组的连接处,因此存在于两个截然不同的

上下文。

图 6.4 中可以看到另一个例子,其中社交群体的构成随时间而变化。用户是一名学生;通讯员是该用户的一位朋友,也是一名社交协调员。请注意网络图中用颜色分隔的三个簇;它们是一个社交群体的各个部分。随着群体成员的毕业,其中一些簇失去了联系。

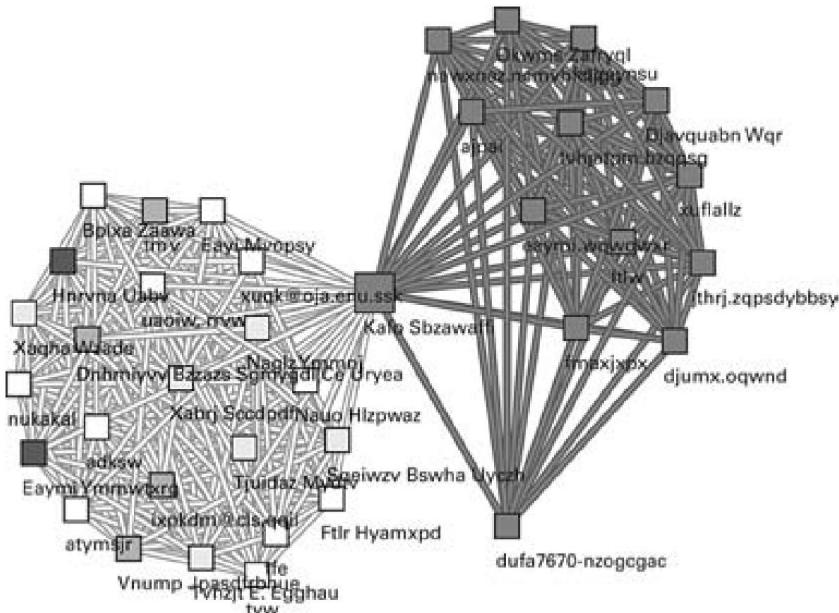


图6.3

以教授为中心的自我中心视角。近期的交流内容以较暗的颜色绘制。

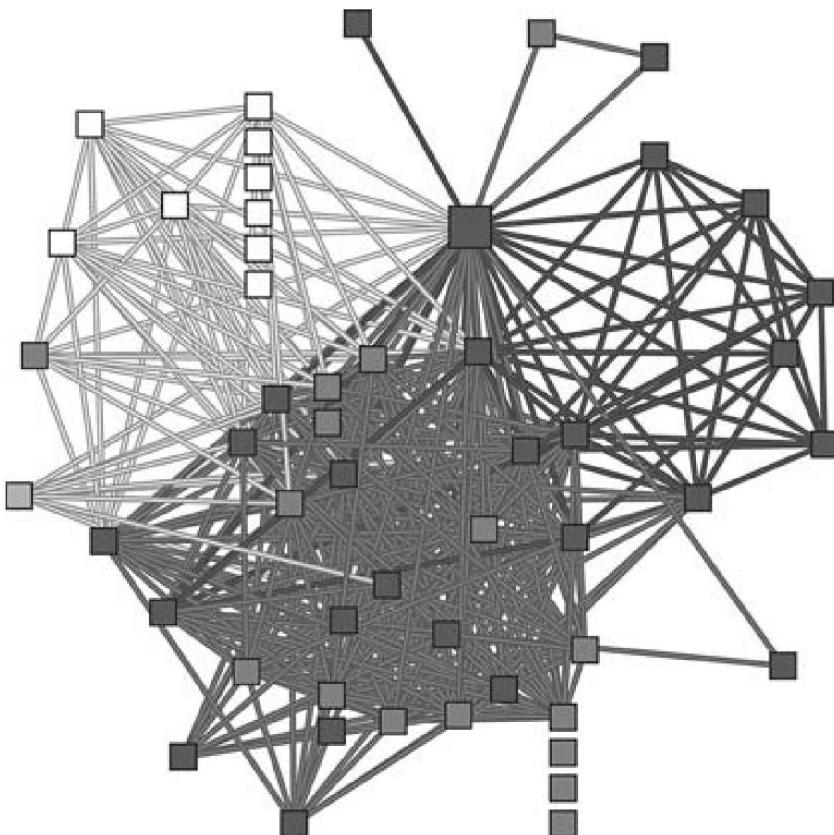


图 6.4

以自我为中心的观点展现了社会群体的变化。

利用EE4P将意识应用于日常任务

Soylent 可以将个人在线互动社交网络的部分内容用作电子邮件客户端的功能,从而为电子邮件消息生成上下文。一个名为“EE4P”(“Enhanced Email for People”)的原型电子邮件客户端利用该网络帮助用户获取信息,并解决上述一些问题。EE4P 是传统三窗格电子邮件客户端的扩展。其代码基于 ICEMail (Nourie 2001,<http://icemail.org>)是一个开源项目。EE4P 目前处于概念验证阶段。

EE4P 使用 Soylent 数据库和 API 为用户提供有关接收和发送邮件的注释信息。每条消息

每个人都与一系列其他消息、人员和群组紧密相连;当用户阅读或撰写消息时,EE4P 会提供有关当前与他们互动的辅助信息。例如,当用户向当前群组撰写电子邮件时,系统会在侧边栏中显示该群组成员过去收到和发送的电子邮件,以及过去与之关联的人员列表。

EE4P 提供三大功能:收件人预测、可轻松访问用户历史记录的增强显示以及增强的地址簿。

收件人预测收件人预测利用网络来推荐逻辑分组。当用户在邮件的“收件人”栏中输入姓名并按下逗号键时,就会触发收件人预测,这提示接下来会有更多姓名。然后,系统会搜索该姓名周围的直接联系人网络,并在下拉菜单中推荐这些姓名。系统可以轻松一次性调出完整的抄送名单(图 6.5)。该系统不受历史组合的限制;它能够推荐作为先前姓名逻辑扩展的分组。通过使用聚类算法,EE4P 能够推荐各种粒度的分组。例如,当一条消息发送给团队负责人时,它可能会根据网络的不同部分推荐四个不同的列表:

- 团队的核心成员;
- 核心成员加上开发人员;
- 核心成员加上设计师;
- 团队中涉及的每个人,包括核心成员、开发人员和设计师。

当然,这些只是建议;用户可以自由选择系统根本没有建议的名称。

用户历史记录和消息显示:当用户阅读或撰写消息时,EE4P 会利用该名称周围的网络,方便用户访问其他消息(基于相似的受众和时间)以及可能与用户相关的其他群组。这些名称、消息和群组均可选择,并可显示有关用户历史记录的广泛信息。具体而言,用户可以访问:

- 消息中涉及的每个人的地址簿条目;

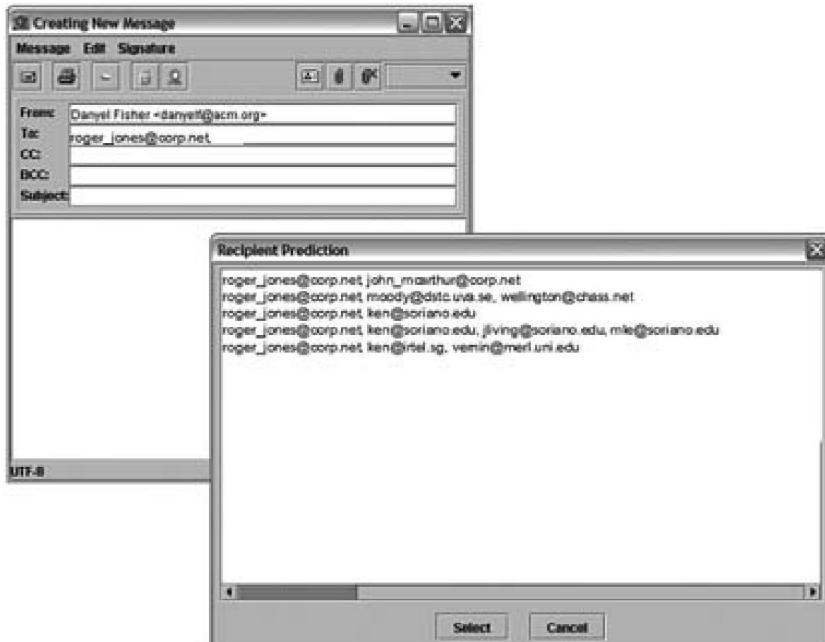


图 6.5
通过 EE4P 填写抄送清单。

- 与文中提到的人可能存在密切联系的其他人信息;
- 涉及该消息的每个人以及其他密切相关人员的消息历史记录;以及
- 涵盖全体参与的群组“消息历史记录”团体。

虽然目前的实现没有考虑到一些潜在有用的数据,例如附件链接或直接相关消息,但该机制设计得足够可扩展以添加这些功能。

增强型地址簿EE4P 提供标准地址簿,用于存储手动输入的个人信息。不过,每个条目都添加了附加信息注释:一个窗格显示社交网络视图,另一个窗格显示过去发送和接收的消息历史记录。

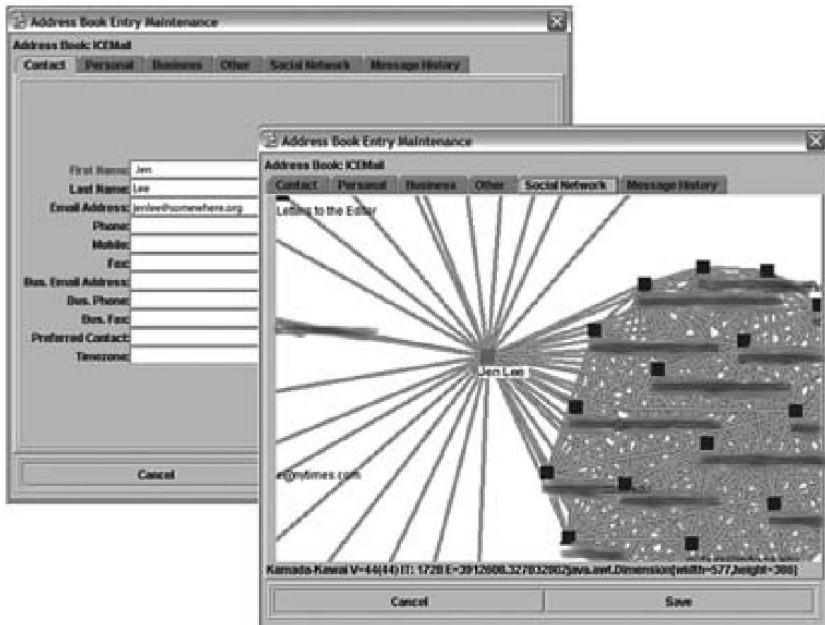


图 6.6
通过 EE4P 查看社交网络。

来自该人。EE4P 还可以为人群动态生成地址簿条目（图 6.6）。

Soylent 的哲学

Soylent 研究中的关键洞察在于，通过在线互动产生的隐性信息可以清晰地展现人们互动的情境（即便不完整）。这些痕迹可以累积成历史记录，并进行有效处理后提供给用户。在本案例中，我们展示了使用该工具进行联系人管理的可能性。

然而，在更普遍的情况下，Soylent 建议更广泛地使用人、物和时间之间的联系。ContactMap 有一种将文件信息与个人数据关联起来的方法；因此，Soylent 也建议在操作系统中提供人与资源之间的联系，从而将社交信息作为一种服务提供给系统组件。

虽然 Soylent 目前拥有由其收集的社交网络强制规定的固定群组概念,但该系统可以支持其他类型的互联信息。未来的工作需要实现可由用户输入的群组通用访问权限。这样,Soylent 可以支持层级结构图和组织群组成员资格集,也可以支持 ContactMap 支持的显式分组。这些动态群组反过来可以根据其成员资格和时间范围与工作动态关联:计算群组的系统还将跟踪与这些群组中人员关联的文件。

与 ContactMap 一样,Soylent 目前也存在工作原型;
然而,它尚未得到积极开发。

Soylent 和 ContactMap 的比较

Soylent 和 ContactMap 都是处理社交网络信息的平台。ContactMap 是一款终端用户工具,旨在为用户群体建模;而 Soylent 则是一个用于开发和构建社交工作空间的基础设施。ContactMap 可以使用 Soylent 构建为终端用户应用程序。

ContactMap 将社交网络清晰地可视化,方便用户在单一用户界面中整合沟通和信息任务。相比之下,
Soylent 则将社交网络作为一种背景信息,用于展示用户之间的互动方式。

Soylent 从电子邮件网络收集信息,而 ContactMap 使用自动生成的联系人列表,用户从中手动构建网络。然而,两者都以用户的通信历史记录为基础,了解需要建模的联系人集合。Soylent 和 ContactMap 对社交网络的理解逻辑是一致的。传统的社交网络分析倾向于关注网络的广度,而像 Friendster 这样的工具允许用户在远处探索自己的社交网络,而 Soylent 和 ContactMap 则强调用户的个人社交网络。它们只关注与用户互动过的人。这些信息反映了用户对世界的看法。

由于 ContactMap 和 Soylent 都只扫描用户的个人电子邮件文件夹(用户可以指定扫描哪些文件夹),因此不会出现常见的隐私问题。然而,两组用户测试

观察到一个不同的隐私问题,我们或许可以称之为“社交隐私”。ContactMap 凭借其照片和颜色分组,让用户的个人社交网络一目了然,以至于任何浏览用户桌面的人都能从中了解到用户对自己社交世界的看法。在我们与原型用户的非正式互动中,他们有时会感到尴尬,比如,如果他们的经理不在地图的中心位置。

同样,Soylent 的显示屏也清晰地标记了人们之间的联系。用户有时会担心,他们认为重要的联系对路人来说太不明显,或者担心人们在地图上显示不该显示的状态。

ContactMap 和 Soylent 之间的区别

Soylent 自动从档案中收集个人信息并构建网络。这些自动生成的网络并不完整。例如,Farnham (2002) 指出,这些网络对某些用户来说可能显得不完整,尤其是那些经常进行面对面交流的用户。ContactMap 采用了一种混合方法:它使用自动收集的姓名作为网络的种子,同时允许用户将条目分组,从而允许人们为甚至未记录的联系人分配重要性。

ContactMap 采用非层级结构;它允许选择一个群组或其成员。ContactMap 中群组的使用不如Soylent 灵活。例如,ContactMap 无法指定“群组中的所有成员,但不包括少数成员”。通过创建多个成员身份重叠的群组,可以在一定程度上缓解这种情况。相比之下,虽然 Soylent 允许将群组作为围绕通讯员的人员集群进行广义访问,但这些群组在系统内没有一致的身份。因此,将信息索引倒特定群组更加困难。

团体。

ContactMap 致力于将个人社交网络可视化。ContactMap 的设计者认为,查看人脸就像将面对面交流的精髓带入计算机媒介交流。联系人节点的存在对于轻松访问联系人、添加提醒和通知以及显示群组也至关重要。

ContactMap 和 Soylent “EE4P”界面都没有显示传统上与社交网络相关的明确的框线可视化。

相反,两者都将网络信息存储在后端。ContactMap 存储的是姓名集合。Soylent 的视图会反复处理大型网络的各个部分,以生成显示和推荐。虽然网络是处理社交信息的有效方式,但它们并非必需 而且,作为主要界面,它们实际上可能会造成混淆。

集成工作环境

我们利用这两个原型工具来探讨一个更加整合的工作环境。正如我们之前所说,在计算机系统中,人可以存在于任何地方,也应该存在于任何地方。如果“人”的概念成为计算机中一项基本可用的服务,那么应用程序就可以进行调整以利用这些信息。

例如,文件系统可以扩展,以考虑其中涉及的人员;日历条目可以添加个人信息注释。文件与几组人员关联:文件创建者、文件发送者、文件编辑者 以及后续步骤、文件接收者或最终受众。其中一些信息可以自动关联,而其他部分则可能需要手动关联。

同样,文字处理器和其他终端用户应用程序也可以效仿EE4P和ContactMap的思路:文档将自动与帮助生成该文档的资源和人员建立联系。例如,当Sally撰写论文的下一稿时,Sam的联系人 作为她论文的队友 可以立即在文字处理器中联系到,既可以作为历史通信记录,也可以作为具有即时通讯状态的实时联系人。

这就要求整个操作系统必须统一个人身份的概念。文件编辑者的名称必须与其即时通讯身份和电子邮件身份关联,并将所有信息收集到一起。

现有工具

Macintosh OS X 的“邮件”程序和微软的 Outlook 或许是朝着这个方向迈出的第一步。这两个程序都将即时通讯工具连接到电子邮件客户端,这样来自通讯员的消息

使用即时通讯工具的用户,其在线状态会通过符号进行标注。用户可以快速访问即时通讯连接,决定是否以及如何回复电子邮件。第一步就是连接不同的工具,共享统一的工作环境。

“我所见之物”项目 (Dumais 等人,2003) 创建了一个可搜索的电子邮件信息、浏览过的网页和文件的存档;它允许用户搜索个人存档,查找按作者和受众索引的文档。“我所见之物”可以理解为一个数据源

这些信息与社交网络和群组工具相结合,可以构成社交工作环境的核心。事实上,几家主流软件供应商最近发布的桌面搜索工具,为索引此类信息、充分利用其社交信息提供了新的机会。

未来方向

下一节将讨论如何从当前的实现方式过渡到这些更完善的形式。我们借鉴了“无位置文档”(Dourish 等人,1999)等技术,这些技术提出了在系统内广泛关联信息的方法。在“无位置系统”中,操作系统支持将任意一组标签与文件关联,然后可以动态查询和显示这些标签。

当这些标签上标注了个人信息时,无地点的文档系统就变成了以人为本的。

然而,仅仅为每个文件,甚至每条数据都标注一个名称是不够的。Soylent 的实地研究提醒我们,人员和项目与时间范围和社会集群密切相关。因此,人与人之间的互联互通为我们提供了宝贵的信息,让我们知道如何索引他们的信息。为了充分体现社交工作场景的概念,需要三个层次的信息。

- 首先,文件和消息必须有一层个人注释。这些注释将一个或多个名称与计算机资源关联起来,并且可以在各种时间关联:创建时、通过电子邮件发送、传输或接收时等等。

- 其次,这些信息必须能够将人们彼此联系起来。某种形式的数据存储应该能够追踪人们之间的互联互通,这些互联互通体现在文件共享编辑、通信发送和接收等方面。

•第三,必须有一种方法来指定和学习群组。ContactMap 的结果清楚地表明,允许人们之间自动和手动互联非常重要且实用。群组信息可以而且应该来自多种来源:社交网络信息、公司层级信息以及手动选择。

限制

本节强调了可以从邮件头、文件位置和附件连接自动获取的信息。此观点有意忽略了邮件和工件的内容。相反,它展示了一种严格基于交互模式的方法的强大功能。全文处理的计算能力并非获取关于人群如何交互的有用信息的必要条件;结构化信息是一个强大的工具,可以提供大部分所需的信息。

话虽如此,衍生信息中仍然存在许多歧义。新联系人发送的孤立消息几乎无法体现任何结构性信息。它究竟是来自拼车的新成员、工作团队的新成员,还是垃圾邮件?现代文本分析技术是解决这些歧义的潜在有效方法,也能帮助那些从事相关项目但从未收到过联合消息的人员聚在一起。这些技术可以检查来回发送的消息文本,并通过检查共享词汇和关键词的存在,收集有关人与人之间逻辑群组和联系的其他信息。

概括

在本文中,我们探讨了社交工作景观的概念,即将物品与其社交情境联系起来。我们讨论了两个探索和阐释工作景观的系统:ContactMap (一款终端用户应用程序)和Soylent (一款处理社交信息的基础设施)。最后,我们借鉴了ContactMap和Soylent的开发和使用经验,构建了统一工作景观的愿景。

当然,设计师可能无法在所有新系统中都融入以人为本的视角。然而,有一些重要的经验教训可以借鉴到各种协作应用中:

•人与人之间的互动意义非凡。应用程序应该记住曾经互动过的人员，并将这些分组提供给用户。

•相互作用并非仅限于两两之间。追踪三元组和更大群体之间的相互作用，并检测这些群体如何随时变化，至关重要。

•交互的时间维度非常重要。大多数应用程序忽略了历史信息，倾向于仅提供群组或交互的当前状态。这忽略了一个事实：群组的变化对于从网络维护到专业知识定位等各种任务都具有重要意义。

•信息必须与人保持关联。两个人之间交换的信息，或通过社交渠道传递的信息，不应脱离该渠道：应该有方法找到联系人发送的信息，反之亦然。如今，这一原则已在版本控制系统中得到体现，即使这些信息似乎不再相关，编辑或文件始终与修改该文件的人相关联。文件与其社交背景密不可分。秉承这套设计理念，未来数字环境的用户将在他们使用的工具中随时随地找到人物的形象。

笔记

1. 1973 年上映的电影《绿色食品》由查尔顿·赫斯顿主演，理查德·弗雷思彻执导，揭示了“绿色食品是由人组成的”。

参考

boyd, d. (2002). 多面身份/实体：管理数字世界中的表征。
未发表的硕士论文。麻省理工学院建筑与规划学院媒体实验室。

boyd, d. (2004). Friendster 和公开表达的社交网络。2004 年计算机系统人为因素会议论文集 (CHI)
2004）。奥地利维也纳，4 月 24 日至 29 日。

Dourish, P., Edwards, W.K., LaMarca, A. 和 Salisbury, M. (1999). Presto：一种用于流畅交互文档空间的实验架构。ACM 计算机人机交互学报 6 (2): 133–161。

190 丹尼尔·费舍尔和邦妮·纳尔迪

- Ducheneaut, N. 和 V. Bellotti (2001). 电子邮件作为栖息地。互动。8 :30-38。
- Dumais, S., Cutrell, E., Cadiz, J.J., Jancke, G., Sarin, R. 和 Robbins, DC (2003). 我所见过的东西:一个用于个人信息检索和再利用的系统。第26届信息检索研究与发展年会 (SIGIR 2003)论文集,第72-79页。加拿大多伦多,2003年7月28日至29日。
- Eveland, JD 和 Bikson, TK (1986)。不断发展的电子通信网络:一项实证评估。载于1986年 ACM 计算机支持协同工作会议论文集,第 91-101 页。德克萨斯州奥斯汀,12 月 - 至 3-5。
- Farnham, S. (2002)。基于自动生成的以人为中心的社交网络的话语架构可视化。发表于2002年话语架构研讨会。收录于2002年计算机系统人为因素会议 (CHI 2002)扩展论文集,第936-937页。明尼苏达州明尼阿波利斯,4月20日至25日。
- Fisher, D. 和 Dourish, P. (2004)。日常协作中的社会与时间结构。载于2004年计算机系统人为因素会议论文集 (CHI 2004) ,第551-558页。奥地利维也纳,4月24-29日。
- Nardi, B. 和 Whittaker, S. (2002)。面对面沟通在分布式工作中的作用。载于 Hinds, P. 和 Kiesler, S. (编)《分布式工作》,第 83-83 页。
112. 马萨诸塞州剑桥:麻省理工学院出版社。
- Nardi, B., Whittaker, S., 和 Schwarz, H. (2002). NetWORKers 及其在内涵网络中的活动。《计算机支持协同工作杂志》 11: 205-242。
- Nardi, B., Whittaker, S., Isaacs, E., Creech, M., Johnson, J. 和 Hainsworth, J. (2002)。ContactMap:通过可视化个人社交网络整合沟通与信息。《ACM通讯》 (4月) :89-95。
- Tyler, J., Wilkinson, DM 和 Huberman, B. (2003)。电子邮件光谱法:自动发现组织内的社区结构。载于 Huysman, M., Wenger, E. 和 Wulf, V. (编)《社区与技术》 ,第 81-96 页。
- 荷兰代芬特尔:Kluwer。
- Whittaker, S., Jones, Q., Nardi, B., Creech, M., Terveen, L., Isaacs, E. 和 Hainsworth, J. (2004)。ContactMap:利用个人社交网络在社交桌面中组织通信。ACM计算机人机界面学报11 (4): 445-471。
- Whittaker, S., Jones, Q. 和 Terveen, L. (2002)。联系人管理:识别联系人以支持长期沟通。载于《计算机支持协同工作会议论文集》,第 216-225 页。路易斯安那州新奥尔良,11 月 16-20 日。
- Whittaker, S. 和 Sidner, C. (1996)。电子邮件超载:探索电子邮件中的个人信息管理。载于1996 年 ACM SIGCHI 计算机系统人为因素会议 (CHI 96) 论文集,第 276-283 页。加拿大不列颠哥伦比亚省温哥华,4 月 13 日至 18 日。

三

从任务到活动

第三部分简介

第三部分包含两章,阐述了“活动”作为数字工作环境设计基本概念的可行性。作者再次描述了他们设计的新颖系统。Voida、Mynatt 和 MacIntyre 在本章中描述的 Kimura 系统,超越了桌面的隐喻,将普通电脑显示器与另一种信息显示界面(即办公室的墙面空间)相结合。墙面上会显示自动创建的用户项目可视化效果,帮助用户跟踪和切换项目。

Bardram 的章节介绍了 ABC (基于活动的计算)框架,以此支持医院的普适计算。尽管本书的大部分章节都涉及个人用户的工作环境 尽管具体系统的设计通常明确地旨在支持沟通和协作 但 Bardram 的章节描述了一个集成的工作环境,可供为共同目标而工作的团队使用。活动可以分配给多个人,它们是系统架构中的一级对象,这使得系统能够识别并支持个人对整个活动的贡献。

尽管这两章都探讨的是具体的设计,但它们的意图并不局限于展示特定的技术。它们也并非仅仅强调系统所体现的设计理念。这两章都提出了一个更普遍的观点,即强调设计师需要拓展视野,思考技术应该如何支持人类。这两章都认为,技术应该通过与他人合作,并通过使用……来支持实现有意义的目标。

多台计算机设备。换句话说，技术应该支持活动，而不是低级的、技术特定的任务。

这两章之间有许多相似之处。首先，它们都探讨了普适计算。普适计算不仅挑战了桌面隐喻，也使得扩展分析和技术支持的重点成为必要。设计师不能像过去那样将精力局限于某一特定技术。普适计算要求设计师关注人们如何整合各种技术以实现有意义的目标。其次，这两章的理论基础都是活动理论。这一源自俄罗斯心理学的框架，后来成为人机交互领域的一种流行方法（Nardi 1996）。

尽管表面上相似，但各章节中提出的观点也大相径庭。在某种程度上，它们甚至可以被视为截然相反的。Voida等人主要以传统意义上的“活动”来探讨人类主体与世界之间有意义的、社会性的、中介性的互动，而Bardram建议将活动作为计算架构中的一流对象，这表明他强调活动是一个计算概念。鉴于活动的“人性化”和“计算化”含义都与设计人们日常生活中的技术支持相关，那么这两种含义之间是如何关联的呢？第三部分的两章都对这个问题进行了初步探讨，这可能会激发人机交互研究的进一步讨论。

参考

Nardi, B. (ed.) (1996) 情境与意识：活动理论与人类
计算机交互。马萨诸塞州剑桥：麻省理工学院出版社。

7

桌面和 普适计算

斯蒂芬·沃伊达、伊丽莎白·D·迈纳特和布莱尔·麦金太尔

20世纪90年代初普适计算范式的出现标志着工作场所计算新时代的开始。

Weiser设想了一个世界，在这个世界里，我们在处理信息时不再把注意力集中在一个盒子上；相反，小型、强大、互联的计算设备的激增将使计算“消失在背景中”（Weiser 1991）。

尽管Weiser的“普适计算”愿景尚未普及，但主流计算技术已开始以Weiser十多年前预测的诸多方式发展。计算已成为许多个人信息设备（例如PDA、手机和数字音乐播放器）不可或缺的一部分，用户全天随身携带。最近，人们对平板电脑的兴趣激增，导致一些商务人士和学生在做笔记和注释文档时放弃使用纸笔，转而使用电子墨水。台式电脑本身正在超越其传统的米色机壳和显示器的界限。

曾经主要存储在PC硬盘上的信息正在进入网站和网络服务；多显示器的使用如今已变得相当普遍，在金融交易等许多领域，拼接显示器组成的虚拟墙正在完全取代传统显示器；可穿戴计算和增强现实领域的实验正在发展成为商业企业，旨在让用户随时随地都能享受台式电脑的功能。

在所有这些发展的交汇处，普适计算环境已经成为研究界中一个常驻的人物。
为设计师提供有形的工作台（例如，Ishii 和 Ullmer 1997；Leibe

智能家居（例如参见 Tran 和 Mynatt 2002）、情境感知教室（例如参见 Abowd 1999）和可重构会议空间（例如参见 Johanson、Fox 和 Winograd 2002；Streitz 等人 1999）都展示了当小型廉价计算渗透到空间并结合传感器、摄像头、投影仪和各种网络技术时，先进的交互技术和社会协作成为可能。

尽管普适计算范式的转变对新设备和应用的设计和部署产生了巨大的影响，但它也影响着技术和工作实践的研究。总的来说，计算机从桌面走向现实世界，使得人们的注意力从探究用户与计算机的对话转向探究计算机的使用环境。对用户如何开展工作的实地研究，从他们组织周围信息的方式（例如，Kidd 1994；Malone 1983；Mander、Salomon 和 Wong 1992），到他们使用白板等现有办公技术的方式（Mynatt 1999），再到他们处理多个同时进行的任务和处理干扰的方式（例如，González 和 Mark 2004），正成为设计新普适计算技术的一个先决条件，甚至比个人电脑时代更为重要。普适计算的愿景打破了以前基于单一、通用隐喻（如图形用户界面的“桌面”）创建应用程序设计的传统；相反，普适计算技术只有当其设计基于其使用环境并与之完美匹配时，才能实现其“隐形”的目标。

在本章中，我们将概述在数字和物理融合的工作场所中，从用户视角支持“活动”概念的议程和方法。这一视角涵盖计算机的使用环境、构成活动的众多工作构件以及活动随时间推移的历史轨迹。我们描述了将计算与活动匹配起来所面临的五个挑战，具体如下：

- 活动是多方面的，涉及各种不同的工作成果；
- 活动是动态的，强调工作成果的延续和演变，而不是结束和归档；
- 活动是协作的，在创造、交流和分享方面
工作成果的传播；

桌面计算支持活动 197

- 由于持续时间、复杂性和所有权的不同,活动存在于不同的粒度级别;
- 活动存在于各个地方,包括物理边界、信息安全和访问的虚拟边界以及固定和移动设置。

我们从两个互补的角度考察普适计算对工作场所活动的支持。首先,我们描述了设计 Kimura 系统的经验。Kimura 是一个集成桌面和交互式白板的环境,支持个体知识型员工管理和切换多项工作活动。在描述 Kimura 之后,我们将根据五大挑战对其设计进行批判性分析。然后,我们将从活动理论的视角考察对活动的支持。特别地,我们指出了活动理论的最新扩展如何解决与我们提出的五大挑战类似的理论缺陷,并提出了弥合日常实践与系统支持之间差距的方向。最后,我们探讨了如何将理论和实践视角相结合,为未来系统设计的五大挑战提供解决方案。

Kimura:以活动为中心的工作环境

我们的研究旨在设计一个能够更好地支持知识型员工(即解读和转化信息的商业专业人士)的办公室(Drucker,1973)。成功的知识型员工能够处理多项任务,与多位同事和客户有效协作,并利用其工作区域的空间组织来处理与当前任务最相关的信息(Kidd,1994;Malone,1983;Mynatt,1999;Grudin,2001)。这些工作实践的多样性以及灵活计算工具实施的复杂性,使得满足所有这些员工的需求变得困难。

我们花费了数年时间开发支持知识型员工的技术。我们在 Kimura 系统上的工作使我们能够开始探索桌面内外不同的活动概念(MacIntyre 等人,2001;Voida 等人,2002)。我们的经验表明,活动可能是一个适用于普适计算环境的实用且统一的框架,但也预示着普适计算环境未来研究的若干挑战。

为了解释 Kimura 系统设计背后的基本概念,我们首先介绍一个简短的场景,重点介绍在典型工作日中与系统进行想象交互的独特方面。

像这样的场景有助于我们集中设计并定义以活动为中心的数字工作环境中的关键用户交互。

木村实践:一个场景

温迪,一位知识型员工,在结束一周的休假后,于周一早上走进办公室。她扫了一眼桌上成堆的文件和白板上的内容,回想起等待她的工作。

在快速浏览了代表正在进行的活动的各种白板蒙太奇之后,她在预算计划上标注了“周三工作,周五截止”,然后把它扔到白板的另一边。

Acme 设计项目蒙太奇中的日历图像让她想起
当天晚些时候举行设计简报会。

她仔细研究了一会儿蒙太奇画面,努力回想休假前她参与设计简报会进行到哪一步了。她看到了最近处理过的一些文档的模糊图像:日历、一幅插图、一个演示文稿和一个网页搜索页面。蒙太奇画面中还包含几张过去文档的半透明图像 两封来自团队客户的重要邮件,以及最初的项目提案。她点击蒙太奇画面,将其加载到桌面上。设计简报文件重新出现在她的电脑上,就像她离开时一样。

快速浏览后,她继续在网上搜索一项有趣技术的详细信息,并对其中一张草图进行了微调。将新草图送去打印后,她决定花点时间整理一下即将举行的开放日的主题创意。她使用桌面控件切换活动(和虚拟桌面),Acme 设计活动的蒙太奇画面重新出现在她的白板上,并标注了打印机图标,表示打印作业正在进行中。

温迪正琢磨着如何回复一位同事提出的一个有趣的话题想法,这时她注意到他的脸出现在了白板上。啊,乔肯定在咖啡室。她觉得面对面的讨论比再发一条信息更有用,于是就去和乔一起喝咖啡,集思广益。

当天晚些时候,她决定继续处理一些预算数字。她眼角余光注意到Acme设计蒙太奇中日历正在轻轻变化。会议时间到了。她跑出办公室时,看到了已完成打印工作的图标。

她很庆幸有人(或有什么东西)掌控着一切,于是在去开会的路上走向打印机。

系统设计与实现

Kimura将用户的“桌面”划分为两个区域:焦点区域,位于桌面显示器上;以及外围显示区域,投影在办公室墙壁上。每项工作活动都与一个独特的虚拟桌面相关联,当用户进行该活动时,该虚拟桌面会显示在显示器上。背景活动则以视觉蒙太奇的形式投影在外围显示器上,如图7.1所示。

从木村的角度来看,一项工作活动(例如管理项目、参加会议或授课)被建模为一组文档和一系列线索,这些线索代表了与该活动相关的人员和对象的持续互动。我们将这些线索称为



图7.1
办公环境中的 Kimura 系统,包括监视器和外围显示器。

活动的工作环境。每个工作环境可能包含大量文档,包括文本文件、网页和其他应用程序文件。工作环境也可能包含正在进行的活动的图标指示,包括未回复的电子邮件和未完成的打印作业。Kimura 会自动跟踪每个工作环境的内容,并根据文档的相对重要性进行标记。与之前的系统(例如 Rooms (Henderson 和 Card,1986))一样,用户手动定义工作环境的边界。在我们的案例中,是通过创建虚拟桌面来实现的。我们之所以选择这种策略,是因为这些操作易于用户执行,并且易于监控以检测工作环境的变化,并且这种策略避免了依赖系统推断这些转换。

每个工作上下文都以从系统活动日志中收集的图像蒙太奇形式显示(见图 7.2)。这些蒙太奇类似于其他多上下文窗口管理器提供的“房间概览”。不过,这些系统会显示每个房间中窗口的精确布局,而我们的目标是提供上下文中过去活动的可视化效果。

这些可视化有助于提醒用户过去的操作;组件图像的排列和透明度会自动为工作情境创建一个图标。此外,蒙太奇图像可以作为从情境感知基础设施中收集的背景感知信息的锚点。

电子白板 蒙太奇可视化的主要显示界面 支持常见的白板实践(Mynatt 1999)。白板具有直观的用户界面,非常适合支持非正式的信息管理活动。我们的系统实现将现有的电子白板交互技术与蒙太奇和通知提示相结合(Igrashi et al. 2000; Mynatt et al. 1999, 2000; Hong and Landay 2000)。这使得用户能够使用非正式提醒对蒙太奇进行注释,并重新定位蒙太奇以指示后台活动的相应优先级。此外,白板的大显示区域是一个理想且不显眼的位置,可以显示与用户工作活动相关的上下文信息以及从办公室周围感知到的上下文信息。

白板让用户能够监控每项正在进行的工作活动,在活动之间顺利过渡,访问旨在促进协作的各种上下文信息,并随时了解相关活动的变化。此外,白板提供的交互性

桌面计算支持活动 201

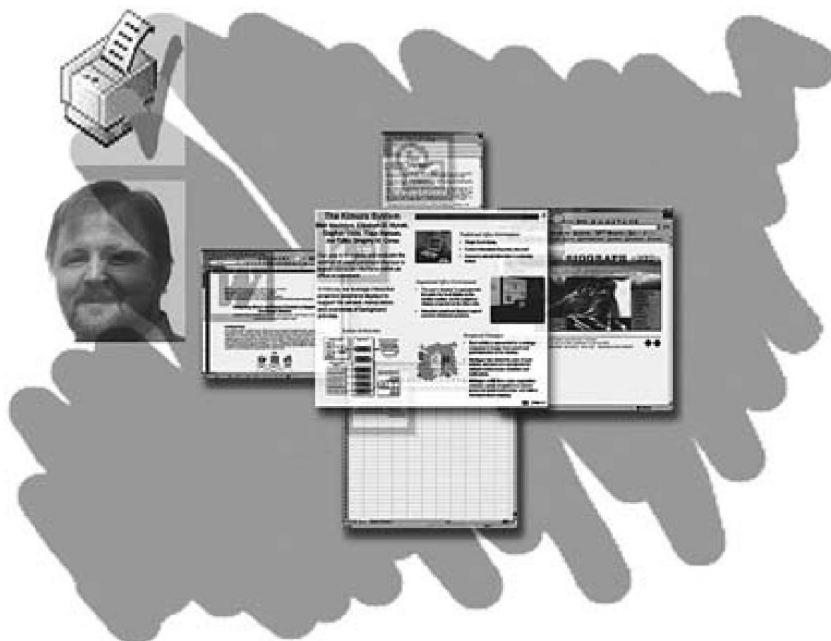


图7.2

工作环境的蒙太奇,包括多个应用程序窗口和两个外部环境通知提示,代表虚拟环境信息 (打印作业的完成)和物理环境信息 (同事的可用性)。

电子白板允许用户非正式地注释和空间组织蒙太奇。

蒙太奇设计减轻了用户维护大量信息 (包括每项工作活动及其相关上下文信息)的负担,也减轻了用户从数量庞大的信息源中动态整合这些信息的负担。蒙太奇的设计旨在以不干扰用户焦点活动的方式呈现这些信息,并满足知识型员工的需求。

Kimura 中的活动和情境意识

Kimura 系统允许用户继续使用他们在工作过程中通常使用的任何工具和实践,同时通过感知和响应虚拟和

用户活动周围的物理环境。大多数情境感知系统通常只关注物理环境（主要是位置）的获取和解读，以使应用程序适应用户的社交和物理环境，而 Kimura 则利用虚拟环境（即处理数字信息所涉及的流程和资源）。

我们的系统使用多个监控组件和代理来获取用户正在进行活动的虚拟上下文。我们的重点是捕获用户与每个活动关联的应用程序和文档窗口的交互。我们使用 Win32 系统钩子 API 开发了一个适用于 Microsoft Windows 的桌面监控系统。

Kimura 系统运行时，Windows 会将低级用户操作（例如，打开窗口、更改窗口焦点、按下按键、单击鼠标）的通知发送到桌面监控进程。监控进程对事件进行编码并将其转发到分布式活动日志。此外，每次窗口系统的输入焦点发生变化时，桌面监控器都会为每个窗口创建屏幕截图。上下文解释器将这些屏幕截图集成到蒙太奇中，以便用户活动的视觉呈现能够包含用户工作的实际图像。这些图像类似于缩略图，比通用图标或标签提供更相关的视觉提醒。我们使用一些指标（例如特定窗口的焦点持续时间以及在打开的窗口之间切换焦点的次数）来确定屏幕截图在电子白板上显示的蒙太奇可视化效果中的大小和位置。

Kimura 还通过电子邮件监控系统获取虚拟环境，追踪用户在工作期间与同事的互动。用户邮件服务器上运行的一个小进程会监控每个用户邮箱的变化。它监控用户发送的所有电子邮件，并将每个邮件收件人与当前工作环境关联起来。

该流程还将收件人添加到 Kimura 用户可能尝试联系的个人列表中，并指示位置监控组件通过观察他们在办公室公共区域的存在来主动监控该个人的可用性。

此外，Kimura 还会观察用户在工作过程中与分布式外围设备的交互。我们实现了一个打印机代理，用于记录待处理打印作业的 ID 和状态。

桌面计算中的支持活动 203

工作上下文。随着每个打印作业状态的变化（例如，打印作业被发送到后台处理程序、在长队列中被淹没后打印，或者由于打印机缺纸而停止），上下文解释器会将通知提示添加到相应的蒙太奇中。

Kimura 还能帮助用户重建工作环境周围的环境状况，并通过物理环境提示用户同事的位置和空闲时间。在我们目前的原型中，我们模拟了一个无处不在的位置感知基础设施（例如 Dey、Abowd 和 Salber 2001），并在办公环境中分布了一系列达拉斯半导体 i-Button 基座。我们设计了传感器网络来检测增强现实办公环境、办公室公共区域和外围设备附近（即打印机旁边）已知人员的到达和离开。此功能可让系统确定 Kimura 用户及其同事的大致位置，并允许系统推断这些同事何时可以协作或何时加入用户的增强现实办公室进行非正式会议。

普适计算环境中支持活动的挑战

Kimura 系统的设计基于我们对活动的理解，它取代了传统的“桌面”、应用程序和文档的隐喻，并允许用户以他们在现实世界中构想和管理任务的方式管理他们正在进行的活动。该系统还基于先前对知识型工作的研究成果，允许用户在空间上组织他们的工作，而无需明确命名或标记信息即可处理。我们在设计过程中秉持着这样的信念：尽管普适计算正在改变我们的工作方式、地点和时间，但在可预见的未来，台式计算机仍将在办公计算中发挥关键作用。

然而，我们做出了一些明确的设计决策来限制 Kimura 项目设计空间的范围，从而限制其复杂性。

例如，我们选择设计一个系统，供一位员工的个人办公室使用，并且主要由该用户使用。我们还将活动表示为“扁平”的文档集合，而不是采用层级结构或视角各异的表示方式，因此

我们将能够更容易地评估每个活动的蒙太奇可视化效果。

随着我们持续开发下一代 Kimura 系统,我们致力于以强调数字化工作环境中介作用的方式扩展该系统。我们非正式的使用经验表明,拥有一个组织和管理个人短期活动的机制固然有益,但如果 Kimura 能够允许用户在数月或数年内管理数量更多、更复杂的活动,并使其能够协调项目团队成员之间的活动,那么它将更加有用。

我们坚信,随着项目的推进,我们最初做出的许多设计决策将继续发挥作用。例如,近期关于工作场所多显示器 (Grudin 2001;Tan 和 Czerwinski 2003) 以及大显示器群件 (Fass、Forlizzi 和 Pausch 2002;Huang、Russell 和 Sue 2004;Johanson、Fox 和 Winograd 2002;Moran 等人 1996;Streitz 等人 1999) 的研究成果激增,这表明,我们关于利用电子白板作为组织空间的直觉将继续卓有成效。然而,如果我们想要取得成功,我们有限的设计空间所带来的副作用,例如我们系统相对简单的活动表示、这些活动的生命周期,以及当前随着时间的推移填充和管理这些活动表示的方式,可能需要彻底重新考虑。

基于我们使用 Kimura 系统的经验以及扩展其功能的尝试,我们确定了在集成数字化工作环境中呈现和支持活动所面临的五大挑战。这些挑战的存在很大程度上源于人类活动固有的复杂性、工作实践中使用的计算工具的技术可行性以及知识型工作的性质 (及其所处的文化环境)。

活动是多方面的。以活动为中心的计算与传统“桌面”隐喻的主要区别之一是,它认识到一个活动通常跨越多个应用程序,并包含多种类型的文档和信息资源。尽管“桌面”隐喻为用户提供了界面级的多任务支持,但应用软件已经变得如此专业化,信息来源也变得如此多样化,以至于一个典型的桌面窗口布局,如果是为了支持单个活动而组织起来的,可能包含数十个跨越多个窗口的窗口。

桌面计算支持活动 205

多个应用程序 除了在活动过程中引用的任何现实世界的工件之外。

Kimura 系统允许用户在活动层面组织和管理工作,而无需手动操作应用程序和文档。我们的设计旨在通过允许用户根据需要在相关的应用程序和文档组之间轻松切换来降低活动切换的开销。这与 Rooms (Henderson 和 Card, 1986 年)、Task Gallery 和 GroupBar (Robertson 等人,本书;Robertson 等人,2000 年;Smith 等人,2003 年) 等系统的初衷大致相同。Kimura 最初将活动与主桌面计算机上的各个虚拟桌面关联;用户虚拟桌面的数量和内容用于识别用户当前的活动,并将应用程序、文档和外部资源与这些活动关联起来。

在普适计算环境中支持活动的多方面性成为一个更为复杂的命题。如果要将活动用作跨各种设备 (例如传统台式机和笔记本电脑、PDA、移动电话、个人服务器式设备 (Want 等人,2002)、共享公共显示器等) 的统一组织结构,那么这些设备必须能够共享一组通用的活动表征,并将这些表征作为其提供的用户体验的组织基石。此外,活动表征必须足够灵活,以涵盖每种设备所使用的各种工作。虽然这听起来像是一个难以实现的愿景,但我们已经证明,可以在不显著改变操作系统或应用软件基本特性的情况下将对活动的支持添加到平台中。

活动是动态的。用户研究和直觉都表明,知识型员工所从事的活动会随着时间的推移而变化,有时变化幅度巨大。项目和里程碑来来去去,活动中使用的工具和信息资源也常常会随着时间推移而变化。此外,过去完成的活动及其结果往往会影响当前的活动,而正在进行的活动又会影响未来将要开展的活动。长期以来,捕捉随时问推移的活动一直是桌面计算领域的一个难题。

例如,保存的文件通常只包含

文档和用户必须经常采用不寻常的工作实践来捕获和访问文档的历史记录,例如使用辅助变更管理系统(如CVS)跟踪更改。¹另一个经常被引用的观察结果是,分层归档系统不容易反映单个资源可能在不同环境中使用的事实(例如,参见Dourish等人,2000年)。

Kimura系统的根本设计决策之一是,以用户实际正在进行的工作为基础,呈现和可视化活动。当用户创建新的虚拟桌面、打开和关闭应用程序、查阅文档以及与同事进行电子互动时,Kimura的用户活动模型会自动反映这些变化。我们呈现活动历史记录的方法是提供能够反映活动整个生命周期状态的可视化效果,而非仅仅提供其当前状态的快照。每个蒙太奇中的文档缩略图均取自每个活动的最新和最重要的部分,即使最重要的部分是已不再打开、因此无法立即访问的文档。

此外,外部情境通知提示的整合,使我们的可视化能够反映活动受“现实世界”感知到的变化影响的动态特性。我们认为,为了准确地呈现活动,这种对活动内容的整体视图将非常宝贵,尤其是在恢复长时间未进行的活动时。

然而,我们的一些实施决策也使得许多长期活动难以运行。为了最大限度地兼容所有桌面应用程序,而不是强迫用户采用一小部分定制的“Kimura感知”应用程序,我们最初选择仅使用窗口句柄、应用程序类型和窗口标题来跟踪和管理活动。不幸的是,这带来了一个限制:只有当活动的窗口在桌面计算机上仍然打开且可用(尽管是隐藏的)时,才能恢复活动。这项设计决策最初旨在实现更切合实际的评估 系统用户可以使用他们在工作过程中已经使用过的任何带有Kimura的应用程序 但实际上却破坏了对系统的长期研究,因为即使是功能强大的现代计算机也存在实际的局限性。

桌面计算中的支持活动 207

关于在给定时间内可以打开的应用程序和文档的数量。

许多其他系统在捕捉用户活动随时间的变化并将此记录呈现给用户方面也取得了相当大的成功。尽管这些系统提供了不同的浏览时间记录的方式 Designers Outpost 通过屏幕底部的“全局时间轴”(Klemmer 等人,2002) ,Flatland 通过可捕捉的、按“段”时间滑块 (Mynatt 等人,1999) ,TimeScape 通过呈现多个交互式桌面可视化效果 (Reki-moto,1999) 所有这些系统都通过允许用户将交互状态恢复到先前某个时间点的状态,间接地支持了界面中的活动概念。无论使用哪种特定的用户界面技术或技术来向用户呈现交互历史记录,这种通用方法都能成功地让用户沉浸在过去活动的情境中,并访问他们用于完成该活动的内容。

活动是协作性的。大多数知识型工作本质上是协作性的。如果活动不以项目团队的多名成员或用户与其直接工作组之外的若干个人之间的同步互动为中心,那么它们几乎肯定会利用其他人在更早时间点创建的信息。认识到数字化工作环境在促进用户有效协作方面的中介作用,是确保这些系统成功的关键一步。

然而,正如计算机支持的协同工作 (CSCW) 社区中大量且多样化的文献所表明的那样,支持有效的协作绝非易事。涉及信息交换、状态保存以及在网络故障情况下的流畅运行的技术问题,以及与感知、协作者角色协商以及隐私等相关的社会问题 仅举几例比比皆是。

我们最初将 Kimura 的适用范围限制为一位用户,以便简化设计空间,并使我们能够在较少的 CSCW 相关约束下迭代基础设施实现和蒙太奇设计。然而,Kimura 能够检测到某些电子通信模式

并将个人与正在进行的任务关联起来。我们还提供了一种可视化技术,根据从情境感知基础设施收集的信息,将同事的空闲时间作为电子白板上蒙太奇画面的一部分呈现。这似乎是我们对系统进行非正式评估时一个非常有用初始步骤。

除了我们单用户实现的 Kimura 系统之外,还有一些设计考量对于为工作活动提供更强大的协作支持至关重要。首先,在活动的计算模型中,其他个体必须被表示为一等对象。将同事纳入活动表示的一种潜在有效方法是利用并可视化正在进行的工作活动与自然发生的虚拟和现实世界社交网络之间的关系(例如,参见 Nardi、Whittaker 和 Schwarz 2002;Fisher 和 Nardi,本书)。此外,活动需要以一种能够共享内容的方式呈现,但需要注意的是,活动中的各个参与者可能对活动有截然不同的认知,他们可能会在活动过程中投入不同的资源,并且,尤其是在许多个人用户参与的大型活动中,用户本身也可能在活动的整个生命周期中来来去去。

此外,此类系统的社会背景;提供协作支持需要比简单地将所有参与者的活动表征和组成资源相互公开更为微妙。参与者可能希望对其资源和工作流程与同事共享的方式和时间进行不同程度的控制。他们可能还希望指定如何与不同的同事共享他们的空闲时间。最后,工作场所的组织结构可能导致每个协作者在活动中扮演不同的角色;因此,每个协作者可能需要访问不同的活动表征或关于其参与者活动和贡献的元信息(Shen and Dewan 1992;Sikkel 1997)。

活动存在于不同的粒度级别。在任何给定的时间点,单个用户可能报告参与了几种不同的活动,每种活动的粒度级别略有不同。例如,她可能正在撰写会议论文评论,编写……

桌面计算中的支持活动 209

为提交的提案编制参考文献列表，并努力争取晋升。论文评审活动只持续很短的时间，并且需要一套独特的资源 即被评审的论文。它也可能类似于其他活动，例如，去年大约同一时间的会议论文评审，并且可能会利用一些与其他活动相关的资源，例如经常用于项目文献评审的研究论文库。提交提案可能是一项相当漫长的任务，涉及更广泛的资源，并且通常需要多位同事的投入。争取晋升可能需要多年的工作，并包含许多其他次要活动。

活动可能存在于不同粒度级别的想法由来已久。Boer、van Baalen 和 Kumar (2002)提出了一个模型，解释了如何将某一分析级别的活动建模为另一分析级别的行动（活动的一个组成部分）。这适用于个人用户，如上例所示，但当从多个参与者的视角审视单个活动时，这种差异会更加明显。例如，经理和首席研究员可能都参与完成一个研究项目，但他们对该活动的重要性、工具、参与者以及具体目标的认知可能截然不同。

Kimura 系统基于主台式计算机上单个虚拟桌面的内容来表示活动，对跟踪活动的内容或生命周期几乎没有限制。我们的蒙太奇可视化设计也适用于不同粒度级别的活动。可视化算法仅显示与每个活动相关的使用时间最长和最近使用的窗口缩略图；无论活动的持续时间长短，也无论用户将其概念化的粒度级别如何，他们最有可能与该活动关联的文档都会显示在白板上。

当然，支持两个或多个用户共享的活动会使情况更加复杂。假设一个用户在高层次、面向项目的层面管理自己的任务，例如年度项目评审和教学；而参与相同活动的另一个用户则在更细粒度的层面管理自己的任务，例如项目评审、演示调试和准备计算机图形学客座讲座。这种情况

当具有不同角色的同事（例如团队成员和经理）协作完成一项活动时，尤其容易出现这种情况。虽然单独为这些用户提供活动级别的支持相对简单，但维护每个用户协作活动的共享表示（以他们偏好的粒度），为每个用户提供合适的活动视图，向每个用户生成活动相关变更的通知，以及协调活动结构随时间的变化，这些都变得非常复杂。

活动跨越地点。活动也跨越地点；也就是说，工作通常在办公环境之外进行。然而，当前的办公技术有时会在不同的物理和虚拟环境中呈现截然不同的信息视图。

例如，由于公司防火墙的存在，与工作活动相关的资源可能对实际位于工作场所之外的用户不可见。即使实际位于工作场所内，如果同事的计算机连接在不同的网络子网（即一台计算机接入有线网络，另一台计算机接入无线网络），他们之间也可能无法就某项活动进行协作。

此外，便携式设备目前的运行界面和层次与办公环境中的设备有很大不同。

台式计算机可能会存储复杂、详细的用户活动和相关资源的表示（当使用活动感知应用程序进行增强时更是如此），而 PDA 和移动电话通常存储非常简单、平面的信息集合，并且需要明确的用户操作来维持设备之间的信息同步。

我们使用 Java 编程语言实现了 Kimura 系统，并使用通用的 TCP/IP 网络协议实现了分布式计算，以便能够在各种设备上轻松实现可视化客户端和情境感知提供程序。虽然我们尚未创建用于 PDA 和手机的信息管理器，但使用 J2ME 虚拟机或使用我们现有的技术创建基于 WAP 的 Kimura 系统 Web 界面应该很容易实现。

服务器。

与网络连接相关的问题虽然超出了我们目前的研究范围，但对许多无处不在的

桌面计算中的支持活动 211

计算工作。虚拟专用网络 (VPN) 等技术允许公司域外的用户通过安全隧道将流量传输到公司的内部网络;零配置网络协议 (例如 Apple 的 Bonjour²) 允许用户查看和使用附近的资源,而无需承担网络设置成本;以及研究平台 (例如 Speakeasy),它促进了服务互操作性并支持临时网络桥接 (Edwards 等人,2002),所有这些技术都有助于减轻网络拓扑对移动用户网络资源可见性和可用性的影响。

理解挑战:理论框架

为了应对我们在设计以活动为中心的普适计算工作环境时所面临的挑战,我们正在开展更深入的实地研究,以了解用户在日常工作实践中对活动概念化的微妙之处。同时,我们也在寻求理论框架来理解活动在此类环境中的作用。

我们已经注意到,普适计算和集成数字化工作环境的出现对人机交互 (HCI) 及相关领域研究人员思考计算环境设计的方式产生了巨大影响。从历史上看,HCI 吸收并调整了人工智能 (AI)、认知科学和认知心理学的知识、流程和技术,以理解和建模用户行为,并通过设计实践将这些发现应用于新界面和技术的创造。由于这种传承,HCI 中用于建模用户的许多理论和技术都表现出明显的认知特征,“代理作为信息处理器”;HCI 中关于用户建模的大部分研究文献都基于模型人类处理器 (Card、Moran 和 Newell, 1983),该模型源于物理符号系统假说。其他重要的用户模型,例如诺曼的七个行动阶段模型 (Norman 1990),可以追溯到吉布森的系统感知学派 (Gibson 1979)。

在过去十年中,人机交互社区的重点开始从基于认知模型的用户界面定量评估转向更具生态意义的技术,包括

情境化和参与式设计（Beyer 和 Holtzblatt 1998; Kyng 1994）。这种“以用户为中心的设计”运动强调技术使用的社会背景，并将用户反馈和参与贯穿于整个设计过程。虽然这种转变在开发既具有可用性又具有高可用性的传统计算机系统方面具有不可估量的价值，

普适计算不仅功能强大，而且实用性强，也为人机交互（HCI）从业者带来了一系列挑战。尤其值得一提的是，大多数用户现在才刚刚开始体验普适计算的愿景，并将这种全新独特的技术融入到他们的工作实践中，这表明另一个焦点的转变可能即将到来：“从以用户为中心的设计到基于情境的设计的转变，与普适计算网络及其连接设备的最新发展相呼应，这些发展正在彻底改变我们与个人计算设备的关系”（Gay and Hembrooke 2003）。

然而，人机交互研究人员和从业人员研究用户与其设备之间关系的方式的变化并不局限于尖端的有形媒体计算或沉浸式环境。

在整个领域，我们正在开展更多的工作来了解用户现有的工作实践，通常涉及传统的台式计算机系统，并开发更好的用户与各种计算设备交互的模型。

近年来，活动理论是提出此类问题的框架之一，并获得了广泛关注。活动理论高度重视工具和社会实践在实现目标过程中的中介作用。这似乎与我们在开发基于活动的计算工具时所面临的挑战相呼应，因此我们相信，活动理论可以作为一个有用的框架，为以活动为中心的数字化工作环境的设计提供指导。

活动理论与以活动为中心的设计

活动理论的起源可以追溯到前苏联，它属于由维果茨基、列昂捷夫和卢里亚创立的文化历史心理学学派。活动理论并非将行动作为分析单位，而是着眼于更广泛的活动层面，并融入认知的社会和文化背景（Halverson 2001; Leont'ev 1978; Vygotsky 1978）。

桌面计算中的支持活动 213

在他们著名的“活动清单”中,Kaptelinin、Nardi 和 Macaulay (1999)提出了活动理论的五项基本原则：

1. 活动的层级结构 在活动理论中,分析单位是指向激发该活动的对象的活动。活动由有意识的、目标导向的行动组成;为了完成任何既定目标,可以采取不同的行动。行动通过自动操作来执行,这些操作本身没有目标。这种层级结构是动态的,可以在活动的整个生命周期中发生变化。

2. 对象导向性活动理论认为,人存在于一个广义的客观现实中,即我们周围的事物具有对自然科学以及社会和文化而言都是客观的属性。

3. 内化/外化活动理论同时考虑内部和外部行动,并认为两者紧密相关。

内化是指将外部过程转化为内部过程,以便在不影响外部世界的情况下计划或模拟某个行动。外化将内部行动转化为外部行动,常用于解决内部行动失败的问题,并

协调独立代理之间的行动。

4. 中介作用 活动理论的核心原则之一是,活动以工具为中介,这些工具在活动过程中被创造和转化,从而使活动的文化和历史融入工具之中。维果茨基对工具的定义非常广泛;他最感兴趣的工具之一就是语言。

5. 发展活动理论以发展作为其主要研究方法之一;也就是说,“实验”通常包括受试者参与某项活动,并观察受试者在活动过程中的发展变化。此外,民族志方法也经常用于识别活动的文化和历史根源。

Engeström (1987)提供了一个经典的可视化模型,概括了活动的结构(图7.3)。该模型基于三种相互关系:行动者(主体)与社区(其他参与的行动者)之间的关系,主体与活动客体(就目标而言)之间的关系,以及客体与社区之间的关系。

这些相互关系由活动的其他组成部分所中介。例如,主体与客体之间的关系由工具（中介性人工制品）中介；因此,主体对客体的体验受到所使用的工具的限制,而作为活动副产品而产生的工具则直接受到主体和客体的影响。这些工具还嵌入了活动其他组成部分的文化和历史,例如管理社区的社会规则、社区本身以及社区的组织（例如,其成员的角色）,有时也称为劳动分工。

然而,Gay 和 Hembrooke (2003)指出了活动理论原始表述的一个缺陷：“活动理论模型……传统上被理解为对活动的共时性、特定时间点的描述。它并没有描述近期许多活动理论研究关注的转变和发展过程。”

Boer、van Baalen 和 Kumar (2002) 提出了一个有趣的建议,即如何将活动理论的范围扩展到时间和组织的各个层面,以解释不同活动之间的联系以及一项活动可能对自身产生的影响：

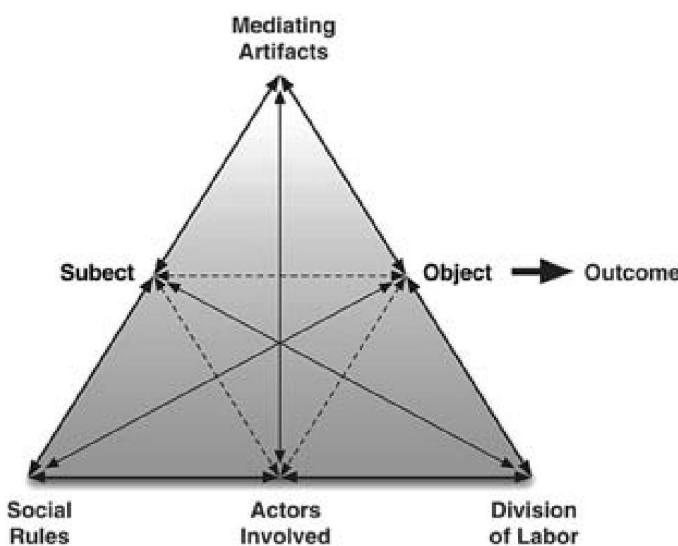


图7.3

改编自 Engeström 对活动和中介关系的分析。

一项活动除了位于影响活动系统的网络中之外,还位于时间中.....

因此,为了理解所研究的活动系统,必须揭示其时间上的相互联系。.....与其将活动系统分析为一幅静态的现实图景,不如描述和分析活动系统内部的发展和张力。.....在分析特定情境层面的活动系统时,还应考虑其与其他情境层面的活动系统(例如,经济系统、行业、供应链、组织、部门或生产流程)的关系。.....所研究的活动系统不仅受到其他情境层面的活动系统的影响,它自身也会对它们施加影响(图[7.4]中的双向扭曲箭头)。这与吉登斯的结构化理论相一致,该理论认为,一方面,人类行为受到社会系统制度属性的制约,而另一方面,这些制度属性又是人类行为的产物。

(Boer,van Baalen 和 Kumar 2002,作者重点)

Boer等人还在不同的分析层面上考察了一项活动在其他活动中可能发挥的作用。他们认为,一个活动系统的组成部分可能在不同文化背景下(例如,在一个项目团队、一个部门或整个公司中的)更广泛或更狭窄的活动中发挥不同的作用(见图7.4)。

这些扩展不仅增加了活动理论模型的复杂性,也有助于解释现实世界系统中存在的矛盾,例如当一个代理在两个目标不同的系统中扮演不同角色时。此外,这种方法使活动理论在表示复杂的分布式认知方面具有与竞争理论方法(例如分布式认知(Hutchins 1995))类似的灵活性。

Nardi (1996) 认为,活动理论的内在优势之一在于它能够捕捉人机交互用户模型中的情境概念。这一概念在普适计算范式及其自身的设计运动 所谓的以活动为中心的设计(Gay and Hembrooke 2003) 中正日益受到重视。Gay 和 Hembrooke 设想的世界依赖于以活动为中心的设计,而非以用户为中心(这目前是人机交互领域的主流观点),因为活动理论为未来由普适计算设备介导的各类交互提供了正确的“方向”。

实用主义与理论主义的交汇

活动理论既被描述为分析工作实践观察的指导框架,也被描述为交流这些观察的语言

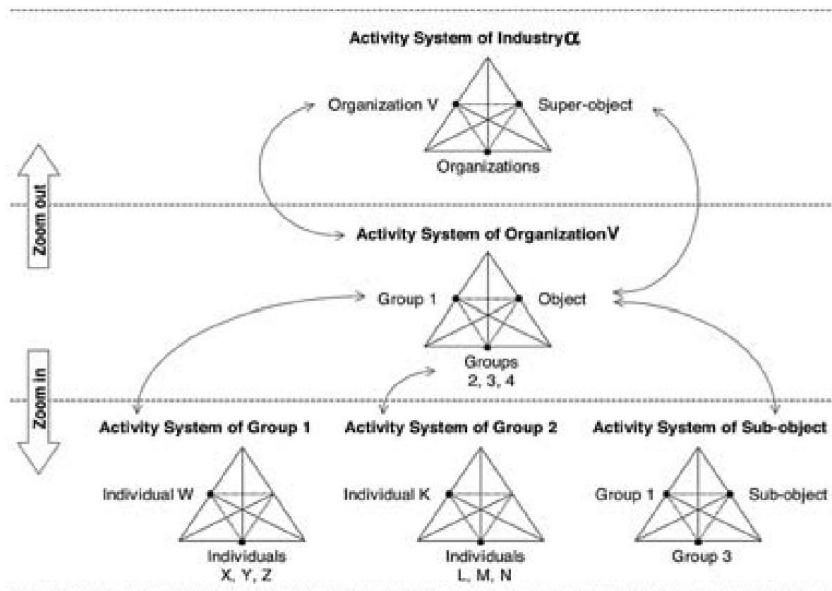


图7.4

不同分析层次之间的关系。(摘自 Boer 等人,2002 年)

经许可转载。2002 IEEE。)

从业者群体内部的发现 (Halverson 2001)。在设计以活动为中心的普适计算环境时,活动理论可以帮助塑造此类系统试图支持的活动的定义。它有助于聚焦和组织对工作实践的现场观察,并顺利地将这些观察转化为设计规范。它还可以在这些集成数字化工作环境中支持活动时遇到的一些最棘手的挑战提供解决方案。

活动理论的核心在于,它提供了一个实用的模型,展现了单个用户视角下完成某个目标的过程。该模型反映了我们在开发 Kimura 系统时所做的许多基本假设,其中最显著的是面向对象的理念。用户在心理上围绕活动(及其构成动作)组织他们的工作,并使用各种工具来实现这些活动的目标。这种视角与人机交互(HCI)界的传统原则形成了鲜明对比,后者强调用户与系统之间的对话,而不是系统在活动情境中作为众多中介工具之一的角色。Kimura 系统反映了这种转变。

通过淡化应用程序-文档隐喻（该隐喻假设用户能够在单个应用程序中完成一项任务）来打破这种视角。相反，Kimura 向用户展示了应用程序和文档集群，并增强了从工作活动的其他虚拟和物理方面感知到的上下文提示。这些集群成为用户管理活动的中心交互点，使他们能够在应用程序和文档之上的抽象级别进行交互，而无需采用新的和不熟悉的工具。

活动理论框架也有助于拓展我们研究现场工作实践的方式，并探索新技术在用户活动中可能发挥的作用。虽然研究工具的使用方式以及工作场所中至关重要的协作方面无疑大有裨益，但活动理论鼓励研究人员从每个参与者的视角审视活动，并理解社会规则和参与者角色的作用，以及人工制品和信息资源的使用。

但或许最引人注目的是活动理论模型如何与我们在与 Kimura 合作的经验以及对其他以活动为中心的普适计算环境的调查中发现的挑战互动。活动理论围绕活动的多面性构建了一张广泛而明确的网络，表明用户的同事和活动对象至关重要，但工具、社会规则以及社区内合作者的角色也必须作为该活动的关键组成部分反映给用户。活动组成部分反映其随时间推移的使用历史，这一观点为以活动为中心的系统提供了几种支持动态工作环境的方法；例如，它们可以将过去的活动记录在档案中，以便在未来的相关任务中快速（并可能实现自动化）参考；而过去和正在进行的活动中使用的工具（例如，文档和信息资源）可能需要始终可用，并带有关于它们过去使用方式的元信息标记。Boer 等人的层级结构。活动理论模型的调整有助于协调我们工作中发现的粒度差异和支持协作的困难；未来以活动为中心的用户界面可能会利用可缩放的用户界面范例或对界面中呈现的细节级别（LOD）进行功能控制，以更准确地反映特定用户概念化自己的任务或同事的任务的深度。

虽然活动理论为理解用户的工作实践提供了一个有用的视角，并提供了一种交流用户行为模型的语言，但有些工作实践方面已被证明对知识型工作至关重要，但却未被活动理论框架所涵盖。例如，知识型员工已被证明依赖于正在进行的活动中所用信息的组织来完成工作，尤其是在这些信息的价值或作用尚未完全确定的情况下（Kidd 1994;Malone 1983;Mynatt 1999）。

活动理论确实提到了工具反映其使用历史的事实，但它并未着重强调知识工作中这一关键组成部分。这一观察意味着，要在普适计算环境中更好地支持活动，我们可能需要借鉴各种活动模型和探究技术，以理解现实世界中工作的完成方式。

然而，理论框架仅提供了理解活动在普适计算环境中作用的一个视角。另一个宝贵的资源是越来越多的研究文献，它们描述了与活动集成到其他计算工具相关的设计决策和实践经验。活动越来越多地被用于组织和管理电子邮件等超载的通信渠道（例如，参见Bellotti等人，2003年；Gwizdka，2002年），作为台式计算机上个人信息管理的索引（例如，参见Kaptelinin，2003年；Kaptelinin和Boardman，本书），以及作为协调用户群体之间行动的一种手段（例如，参见Bardram，2005，本书）。这些实验的结果将进一步帮助阐明与在用户界面中表示活动相关的问题和挑战，并为社区提供更多样化的活动建模方法组合，并将这些模型展示给系统用户。

当设计师面临创建下一代集成数字工作环境时，活动理论等理论框架和从我们在木村系统方面的工作中获得的实用观点将在指导这些系统的设计和克服支持现实世界工作实践所带来的挑战方面发挥关键作用。

笔记

- 1.<http://www.cvshome.org/>。
- 2.<http://developer.apple.com/networking/bonjour/>。

参考

Abowd, GD (1999). 课堂2000:一个基于仪器的生动教育环境实验。IBM系统期刊:普适计算特刊38 (4): 508–530。

Bardram, JE (2005). 基于活动的计算:对普适计算中移动性和协作的支持。《个人与普适计算》 9 (5): 312–322。

Bellotti, V., Ducheneaut, N., Howard, M. 和 Smith, I. (2003)。将电子邮件转化为任务:以任务管理为中心的电子邮件工具的设计与评估。载于ACM 计算机系统人为因素会议 (CHI 2003) 论文集,第 345-352 页。佛罗里达州劳德代尔堡,4月 5 日至 10 日。

Beyer, H. 和 Holtzblatt, K. (1998)。情境设计:定义以客户为中心的系统。旧金山:Morgan Kaufmann。

Boer, N., van Baalen, PJ 和 Kumar, K. (2002)。“一种研究知识共享情境性的活动理论方法”。刊于第 35 届夏威夷系统科学国际会议 (HICSS-35 02)论文集。夏威夷大岛,1月7日至10日。

Card, SK, Moran, TP, 和 Newell, A. (1983). 《人类心理学》
计算机交互。新泽西州希尔斯代尔:劳伦斯·埃尔鲍姆。

Dey, AK, Abowd, GD, 和 Salber, D. (2001)。支持情境感知应用程序快速原型设计的概念框架和工具包。

人机交互杂志16 (2-4): 97-166。

Dourish, P., Edwards, WK, LaMarca, A., Lamping, J., Petersen, K., Salisbury, M., Terry, DB, 以及 Thornton, J. (2000). 使用用户特定的活动属性扩展文档管理系统。ACM信息系统学报18(2): 140–170。

Drucker, PF (1973). 管理任务、职责与实践。纽约:Harper and Row出版社。

Edwards, WK, Newman, MW, Sedivy, JZ, Smith, TF, Balfanz, D., Smetters, DK, Wong, HC, 以及 Izadi, S. (2002)。使用 Speakeasy 进行临时点对点协作。载于ACM 2002 计算机支持协同工作会议 (CSCW 2002) 论文集,第 256-265 页。明尼苏达州明尼阿波利斯,4月 20-25 日。

恩格斯特罗姆,Y. (1987)。通过扩展来学习。赫尔辛基:Orienta-konsultit。

Fass, AM, Forlizzi, J. 和 Pausch, R. (2002)。MessyDesk 和 MessyBoard:两款旨在提升人类记忆力的设计。载于“交互系统设计会议论文集:流程、实践、方法和技术” (DIS 2002) ,第 303-311 页。英国伦敦,6月 25-28 日。

Gay, G. 和 Hembrooke, H. (2003)。以活动为中心的设计:一种设计智能工具和可用系统的生态方法。马萨诸塞州剑桥:麻省理工学院出版社。

Gibson, JJ (1979). 《视觉感知的生态学方法》。波士顿:霍顿·米夫林。

220 斯蒂芬·沃伊达、伊丽莎白·D·迈纳特和布莱尔·麦金太尔

González, VM 和 Mark, G. (2004)。“持续不断的多任务疯狂”：管理多个工领域。载于ACM计算机系统人为因素会议 (CHI 2004) 论文集。奥地利维也纳,4月 24 日至 29 日。

Grudin, J. (2001). 分割数字世界：多显示器使用中的焦点意识与边缘意识。载于ACM计算机系统人为因素会议论文集 (CHI 2001) ,第458-465页。华盛顿州西雅图,3月31日至4月5日。

Gwizdka, J. (2002). TaskView：基于任务的电子邮件界面的设计与评估。载于IBM高级研究中心会议论文集 (CASCON 2002) ,第136-145页。加拿大多伦多,9月30日至10月3日。

Halverson, CA (2001). 活动理论与分布式认知：或者，CSCW 需要与理论做什么？计算机支持的协同工作 (CSCW) 11 (1-2): 243-267。

Henderson, JDA 和 Card, SK (1986)。房间：使用多个虚拟工作空间来减少基于窗口的图形用户界面中的空间争用。

ACM 图形学学报5 (3): 211-241。

Hong, JI 和 Landay, JA (2000)。SATIN：一款非正式墨迹应用工具包。载于ACM 用户界面软件与技术研讨会论文集 (UIST 00),第 63-72 页。加州圣地亚哥,11月 6-8 日。

Huang, EM, Russell, DM, 和 Sue, AE (2004)。IM Here：在大型共享显示屏上实现工作组交互的公共即时消息。载于ACM 计算机系统人为因素会议论文集 (CHI 2004),第 279-286 页。奥地利维也纳,4月 24 日至 29 日。

Hutchins, E. (1995)。野外认知。马萨诸塞州剑桥：麻省理工学院出版社。

Igrashi, T., Edwards, WK, LaMarca, A. 和 Mynatt, ED (2000)。基于笔的电子白板交互架构。载于《高级可视化界面工作会议论文集》,第68-75页。意大利巴勒莫,5月23-26日。

Ishii, H. 和 Ullmer, B. (1997)。《有形比特：迈向人、比特和原子之间的无缝接口》。载于SIGCHI 计算机系统人为因素会议 (CHI 97) 论文集,第 234-241 页。佐治亚州亚特兰大,3月 22-27 日。

Johanson, B., Fox, A., 以及 Winograd, T. (2002)。交互式工作区项目：普适计算房间的体验。IEEE普适计算1(2): 67-74。

Kaptelinin, V. (2003)。UMEA：将互动历史转化为项目情境。载于ACM计算机系统人为因素会议论文集(CHI 2003),第353-360页。佛罗里达州劳德代尔堡,4月5日至10日。

Kaptelinin, V.、Nardi, BA 和 Macaulay, C. (1999)。活动清单：一种表示情境“空间”的工具。《交互》 6 (4): 27-39。

Kidd, A. (1994)。分数取决于知识型员工。载于《ACM 计算机系统人为因素会议论文集》 (CHI 94) ,第 186-189 页。

191. 马萨诸塞州波士顿,4月 24 日至 28 日。

- Klemmer, SR、Thomsen, M.、Phelps-Goodman, E.、Lee, R. 和 Landay, JA (2002)。网站从何而来?捕捉并融入设计史。载于ACM计算机系统人为因素会议论文集 (CHI 2002) , 第1-10页。明尼苏达州明尼阿波利斯,4月20-25日。
- Kyng, M. (1994). 北欧设计:产品开发中的用户。载于ACM计算机系统人为因素会议 (CHI 94)论文集,第3-9页。马萨诸塞州波士顿,4月24-28日。
- Leont'ev, AN (1978).活动、意识与人格。新泽西州恩格尔伍德克利夫斯:普伦蒂斯霍尔出版社。
- Leibe, B., Starner, T., Ribarsky, W., Wartell, Z., Krum, D., Weeks, J., Singletary, B. 和 Hodges, L. (2000). 迈向与感知工作台的自发交互。IEEE计算机图形学与应用20(6): 54-65。
- MacIntyre, B.、Mynatt, ED、Voida, S.、Hansen, KM、Tullio, J. 和 Corso, GM (2001)。利用交互式外设显示器支持多任务处理和背景感知。刊于第14届ACM用户界面软件与技术研讨会 (UIST 01)论文集,第41-50页。佛罗里达州奥兰多,11月11日至14日。
- Malone, TW (1983). 人们如何整理办公桌?对办公信息系统设计的启示。ACM办公信息系统学报1(1): 99-112。
- Mander, R.、Salomon, G. 和 Wong, YY (1992)。“堆”的隐喻支持信息的随意组织。载于SIGCHI计算机系统人为因素会议(CHI 92)论文集,第627-634页。加州蒙特雷,5月3-7日。
- Moran, TP、Chiu, P., Harrison, S., Kurtenbach, G., Minneman, S. 和 van Melle, W. (1996)。持续协作工作过程中的进化参与:案例研究。载于1996年ACM计算机支持协同工作会议论文集,第150-159页。加拿大温哥华,4月13日至18日。
- Mynatt, ED (1999). 不祥之兆。载于INTERACT 99 会议论文集,第 196-204 页。苏格兰爱丁堡,8月30日至9月3日。
- Mynatt, ED, Igrashi, T., Edwards, WK, 和 LaMarca, A. (1999)。《平面国:办公室白板的新维度》。载于ACM 计算机系统人为因素会议 (CHI 99) 论文集,第 346-353 页。宾夕法尼亚州匹兹堡,5月 15-20 日。
- Mynatt, ED, Igrashi, T., Edwards, WK, 和 LaMarca, A. (2000)。设计增强书写表面。IEEE计算机图形学与应用20 (4): 55-61。
- Nardi, BA (1996). 情境研究:活动理论、情境化行动模型与分布式认知的比较。收录于 Nardi, BA 主编的《情境与意识:活动理论与人机交互》,第 69-102 页。
- 马萨诸塞州剑桥:麻省理工学院出版社。
- Nardi, BA, Whittaker, S. 和 Schwarz, H. (2002)。NetWORKers 及其在内涵网络中的活动。计算机支持的协同工作11:205-242。

222 斯蒂芬·沃伊达、伊丽莎白·D·迈纳特和布莱尔·麦金太尔

Norman, DA (1990). 《设计心理学》。纽约:Doubleday出版社。
Rekimoto, J. (1999). 时间机器计算:一种以时间为信息环境方法。载于ACM 用户界面软件与技术研讨会议论文集 (UIST 99),第 45-54 页。北卡罗来纳州阿什维尔,11 月 7-10 日。

Robertson, G., van Dantzig, M., Robbins, D., Czerwinski, M., Hinckley, K., Risden, K., Thiel, D. 和 Gorokhovsky, V. (2000)。任务库:一个 3D 窗口管理器。载于 ACM 计算机系统人为因素会议 (CHI 2000) 论文集,第 494-501 页。荷兰海牙,4 月 1 日至 6 日。

Shen, H. 和 Dewan, P. (1992)。协作环境中的访问控制。载于 1992 年 ACM 计算机支持协同工作会议 (CSCW 1992) 论文集,第 51-58 页。加拿大多伦多,11 月 1 日至 4 日。

Sikkel, K. (1997)。基于群组的合作系统授权模型。载于第五届欧洲计算机支持合作工作会议 (ECSCW 97) 论文集,第 345-360 页。英国兰开斯特,9 月 7-11 日。

Smith, G., Baudisch, P., Robertson, G., Czerwinski, M., Meyers, B., Robbins, D. 和 Andrews, D. (2003)。GroupBar:任务栏的演变。载于 OZCHI 03 (澳大利亚计算机人机交互会议)论文集,第 34-43 页。

澳大利亚布里斯班,11 月 26 日至 28 日。

Streitz, NA, Geißler, J., Holmer, T., Konomi, S., Müller-Tomfelde, C., Reischl, W., Rexroth, P., Seitz, P. 和 Steinmetz, R. (1999)。i-LAND:一个面向创造力和创新的互动景观。载于 SIGCHI 计算机系统人为因素会议 (CHI 99) 论文集,第 120-127 页。宾夕法尼亚州匹兹堡,5 月 15-20 日。

Tan, DS 和 Czerwinski, M. (2003)。在多个显示器上分发信息时视觉分离和物理连续性的影响。载于 INTERACT 2003 会议论文集,第 252-265 页。瑞士苏黎世,9 月 1 日至 5 日。

Tran, Q. 和 Mynatt, ED (2002)。Cook's Collage:两种探索性设计。
明尼苏达州明尼阿波利斯市 ACM 计算机系统人为因素会议 (CHI 2002) “家庭新技术”研讨会立场文件。

Voida, S., Mynatt, ED, MacIntyre, B. 和 Corso, GM (2002)。整合虚拟与物理环境以支持知识型员工。IEEE 普适计算 1(3): 73-79。

维果茨基,LS (1978)。《社会中的心智:高级心理过程的发展》。马萨诸塞州剑桥:哈佛大学出版社。

Want, R., Pering, T., Danneels, G., Kumar, M., Sundar, M. 和 Light, J. (2002 年)。个人服务器:改变我们对普适计算的思考方式。载于《第四届普适计算国际会议 (UbiComp 2002)》论文集,第 194-209 页。瑞典哥德堡,9 月 29 日至 10 月 1 日。

Weiser, M. (1991)。21 世纪的计算机。《科学美国人》265 (3): 94-104。

8

从桌面任务管理到 无处不在的基于活动的计算

雅各布·E·巴德拉姆

介绍

传统的计算机技术是基于以应用程序和文档为中心的模型设计的,部分原因是为了满足用户对特定、有针对性的应用程序的需求,这些应用程序支持特定任务并处理特定类型的信息,例如写信或做预算。这种以应用程序为中心的计算模型深深植根于当今的硬件、操作系统软件、用户界面软件和开发框架中。事实证明,它非常适合桌面办公,但这种以个人和任务为导向的方法几乎无法支持更高级别活动所需的资源和工具的聚合。用户需要根据手头的活动将这些资源和工具聚合成有意义的包,并且在并行活动之间进行多任务处理时,通常需要手动重新配置这种聚合。例如,在撰写商务备忘录时,人们会使用一整套应用程序(文字处理器、电子表格、图形工具、统计软件包、ERP系统等),每个应用程序都使用一组特定的数据和文档。当切换到其他活动(例如阅读电子邮件和/或浏览网页)时,需要对应用程序、文档和文件进行全新的配置。尽管已有研究致力于解决这一难题,并提出了诸如Rooms(Henderson和Card,1986年)、Task Gallery(Robertson等人,2000年)、Kimura(MacIntyre等人,2001年)、GroupBar(Schmidt等人,2003年)和Topos(Grønbæk等人,2001年)等系统,但当今大多数操作系统几乎或根本不支持在这些活动之间切换。

当用户从一个工作环境移动到另一个工作环境时,移动和游牧工作会放大重新配置的开销,可能会使用不同的

224 雅各布·E·巴德拉姆

计算机和不同类型的设备。因此,用户通常被“绑定”到他们的个人计算机,从而在个人计算机和用户之间建立了一对一的关系。

计算世界正逐渐走向一个无处不在、无处不在的计算世界。一方面,用户使用各种各样的异构设备,例如汽车、家庭娱乐电脑、自动冰箱、手机以及各种大型和小型计算机。另一方面,许多用户使用各种各样的公共可用设备,例如冰箱、公共显示器、电视等等。因此,现在存在着一种多对多关系。

在用户和计算机之间。

在本章中,我将描述一种新的普适计算系统概念,我称之为基于活动的计算(ABC)(Christensen and Bardram 2002;Bardram 2005b,2004)。在基于活动的计算中,基本计算单元不再是文件(例如文档)或应用程序(例如MS Word),而是用户的活动。最终用户直接由计算活动支持:计算活动可以在任何时间点在基础设施中的任何计算设备上启动、暂停、存储和恢复,也可以移交给其他人或由多人共享。此外,活动的执行会根据用户的使用情境进行调整,即活动具有情境感知能力。我们的目标之一是使临床应用程序的开发人员能够通过在医院运行的这种普适计算基础设施中设计和部署他们的程序,来整合对移动性、中断、并行活动、协作和情境感知的支持。

ABC框架提供了一个运行时基础设施,其中包含支持医疗工作中这些核心挑战的服务以及用于开发ABC服务和应用程序的编程模型。

先前的工作

Norman在其著作《隐形计算机》(2000年)中简要提及了基于活动的计算(Activity-Based Computing)的概念。Norman基于对使用PC(Macintosh电脑)的办公室用户的观察,提出了将应用程序或组件根据当前活动收集到逻辑包中、用于活动恢复以及共享活动空间的需求。

这些想法是在20世纪90年代初在苹果公司开发的,其核心技术

普适的基于活动的计算 225

逻辑上的想法是将这种方法建立在 OpenDoc 标准之上。

OpenDoc 是苹果公司在桌面上实现基于组件软件的方法,类似于 Windows 平台上的 OLE/COM 组件方法。遗憾的是,该项目未能获得苹果公司的管理层支持,因此从未实现。苹果公司这种基于活动的计算方法的灵感源自 Rooms 系统 (Henderson 和 Card,1986) ,它是所有虚拟桌面程序的鼻祖。Rooms 提供了将桌面上的应用程序窗口按逻辑捆绑 (即“房间”)排列的机制,并允许在它们之间轻松切换。与我们方法中提出的基于活动的计算理念相比,苹果公司的 ABC 和 Rooms 原则仍然针对的是非移动的、个人桌面办公人员的计算。使用 OpenDoc 组件技术的方法本质上将组件捆绑到一台物理设备,并且不支持将“活动”从一个设备移动到另一个设备,也不支持在协作用户之间共享该活动或其中的一部分。¹

基于活动的计算概念与 Aura (Sousa 和 Garlan,2002)中的任务驱动计算概念有相似之处,包括对人工任务的支持、跨异构设备的用户移动性、对情境感知自适应的支持以及本地资源发现。然而,基于活动的计算更侧重于工作环境中的本地移动性,而非 Aura 中讨论的远程移动性。此外,ABC 框架的固有设计旨在支持异步和同步协作,而 Aura 项目中均未提供这两种协作。此外,Aura 专注于普适计算中间件的软件架构,该项目尚未对用户界面的设计或此类计算环境的使用进行研究。从用户界面的角度来看,像 Windows 的 Task Gallery (Robertson 等人,2000 年)、GroupBox (Schmidt 等人,2003 年)和 Kimura (MacIntyre 等人,2001 年)这样的系统已经设计出了在基于窗口的用户界面中处理多任务的方法。这些“虚拟桌面”方法将任务视为一个紧密结合的应用程序集合。当用户引用某个特定任务时,系统会自动调出与该任务相关的所有应用程序和文档。这减轻了用户启动和查看任务的负担。

单独管理和安排应用程序和文档。在我们的工作中,我们扩展了这一概念,将活动建模为与能够处理这些服务的应用程序解耦的抽象服务集合。这

226 雅各布·E·巴德拉姆

将 Activity 与特定应用程序解耦，允许 Activity 在不同主机上运行不同的支持应用程序，并在不同的环境中进行实例化。这也意味着参与 Activity 的不同用户可以在 Activity 中分别使用自己喜欢的应用程序。

从理论和概念层面来看，多年来对“工具和材料”的支持一直是人机交互的主导设计理念。这一设计理念可以追溯到帕洛阿尔托研究中心（PARC）的早期研究以及乌托邦项目（Bødker et al. 1987），并在概念上被设想为用户界面设计的“直接操作”方法（Norman and Draper 1986）。这一设计理念主张直接支持用户使用工具（即人工制品）和他们正在处理的材料（活动对象）所进行的操作。这一设计理念与活动理论（Leont'ev 1978）的传统解读非常契合，活动理论指的是木匠、铁匠和其他工匠的工作。正如Bødker（1991）所建议的那样，这种方法是基于活动理论的人机交互设计方法的起点。此外，这种设计理念和人类活动的概念化也包含着对工作流系统的根本怀疑，因为这类系统包含（或用活动理论的方式将其具体化）了人类活动的概念化，即作为一种控制人类工作的心理建构（在工作流系统中则是一种计算建构）。这与活动理论截然相反，活动理论强调心理建构（动机和目标）为活动提供方向，但活动的执行要根据具体情境的物质条件进行调整。这一原则被萨奇曼（1987）称为“情境行动”。因此，源自传统活动理论解读的设计理念主张支持工具和材料，这允许用户根据活动发生的情境调整活动的执行（即操作层面）。通过这种方式，活动保持了其与世界的辩证关系，一方面，活动受人类认知（目标）的引导，另一方面，活动又受其执行的物质条件的影响。

我们提出的基于活动的计算（ABC）方案听起来像是一个工作流系统，它试图对人类活动进行建模，包括构成活动的操作。我们甚至讨论过（并且目前正在研究）如何将人类意图（即活动的目标）表示为

普适的基于活动的计算 227

我们计算支持的一部分。然而,ABC 不应被视为一种工作流系统方法。在基于活动的计算中,“计算活动”是“人类活动”的数字化对应物,前者仅仅是后者的一种表征,后者是活动理论所定义的活动。因此,ABC 并不试图通过对活动进行建模来控制人类活动的执行 恰恰相反。在工作流系统中,计算活动控制并因此定义了人类活动。而在 ABC 中,人类活动定义了计算活动。

仔细审视为工具和材料(活动的中介和对象)提供支持的设计理念,ABC 也同样如此。将工具和材料支持转化为对图标、文档、滚动条等基本操作级制品和对象的低级支持并非唯一选择。根据活动理论,尤其是维果茨基(参见 Wertsch 1985)和恩格斯特罗姆(Engeström,1987)的著作,中介和对象也是人类活动的更高层次的方面。例如,语言及其概念、生产和工作计划、组织中人员之间的分工以及社会规则和法律,都是复杂的现代社会中中介的例子。因此,帮助人们协调和执行活动的人类活动表征也是主要的中介。同样,工作的对象不一定是物理实体,例如木匠的木屋、铁匠的马蹄铁或猎人的猎物。现代人类活动的对象还包括治疗和护理病人、制定汽车生产制造计划以及进行科学研究。

随着计算机技术在我们的职业和个人生活中扮演着越来越重要的角色,这些对象通常以数字化形式呈现,其中一些可能仅存在于数字世界中,例如包含生产计划或软件程序的计算机辅助设计/计算机辅助制造(CAD/CAM)系统。因此,需要计算支持来处理日益增长的复杂性以及海量的数字对象和中介。作业行为理论(ABC)的目标是为处理人类活动提供更高级别的工具和材料(中介和对象),而人类活动需要处理大量的数字对象。然而,当重点从工具和材料的运行支持转向更高级别的活动和行动支持时,活动理论的基本原则仍然适用。活动的执行

228 雅各布·E·巴德拉姆

作业成本法（ABC）仍然发生在特定的物质世界中，因此其运作会根据具体情况的具体机遇和条件进行调整。这种根据具体情境的具体物质条件来执行活动的做法在作业成本法中得到了保留。此外，与动物活动相比，人类活动的一个显著特征是其协作性。因此，人类通过相互分配活动行动，并使用中介工具（包括计划、时间表和规则）来协调这些分布式活动，从而进行合作。因此，作业成本法的另一个核心方面是通过创建基于计算机的协作工具来协调协同工作活动，从而支持人类活动的这种合作性（Bardram 1998）。

实证背景

基于活动的计算原则的实证背景是自1995年以来对丹麦医院工作进行的广泛实地研究。在分析医院的临床工作和患者治疗（以及作为本研究一部分的计算机技术应用）时，显而易见的是，当代计算机、操作系统和应用程序与临床医生日常工作中许多并行活动中存在的中断性、分布式、游牧式和繁忙的工作模式并不相符。个人电脑、笔记本电脑、PDA和平板电脑主要适合办公室职员，他们通常在固定地点（通常是桌面）长时间、相对不受干扰地处理个人事务。因此，当代计算机技术在医院中面临着一系列挑战，这使得医院成为研究和设计“超越桌面”的普适计算技术的理想应用和研究领域。在本节中，我们将更详细地探讨其中一些挑战。

应用和数据导向

临床医生将他们的工作视为由一系列活动组成，其中一些活动是相互关联的。这些活动包括“治疗佩德森夫人”和“培训实习医生汉森先生”。这些活动以一系列动作的形式进行，这些动作又通过一系列具体的物理操作来实现。例如，治疗佩德森夫人的活动涉及各种各样的动作，例如查看X光片、查看血液

普适的基于活动的计算 229

医生们会查看检查结果、安排新的血液检查、分析血液检查结果、监测患者的体温和脉搏,以及开药和给药。然而,临床医生们对这些行为并不怎么在意。在采访他们时,这些行为并非主要关注点 在描述他们的工作时,他们谈论的是“对佩德森夫人的治疗”,而不是查看血液检查结果。

观察临床医生如何使用计算机 尤其是电子病历 (EPR) 通常会发现,一项活动中的不同操作由不同的计算机应用程序支持。例如,用于查看 X 射线图像的应用程序由图像归档和通信系统 (PACS) 支持,医疗模式作为 EPR 的一部分显示,而安排血液检查是预约和排程系统的一部分。即使所有这些应用程序都用于支持同一项活动 例如治疗患者 但很少有人支持将相关的应用程序和服务集合聚合到与该活动相对应的逻辑束中。本质上,大多数当代计算机技术都是以应用程序和数据为中心的。

因此,几乎没有研究支持在活动之间进行交替。医院里的临床医生同时参与许多活动,他们不断地从一项活动切换到另一项活动。因此,在巡房期间,一名护士可能要照顾三名病人,同时还要监督一名实习医生,并帮助一些家属寻找他们的父亲。此外,干扰是医院工作的重要组成部分,护士和医生经常会互相打断,讨论病例,接听电话,或者必须赶赴紧急情况现场。值得注意的是,与许多关于办公室工作干扰的研究 (Conaill 和 Frohlich 1995;Rouncefield 等人 1995)不同,医院里并非所有干扰都被视为滋扰,而是在繁忙的工作环境中紧密合作的重要组成部分。

固定工作

大多数当代计算机技术都是为固定桌面使用而设计的。然而,在医院工作的临床医生流动性极大,他们中的大多数人甚至没有办公桌或椅子 (Bardram and Bossen 2005)。此外,医院的计算机通常位于病房的小型办公室中,这意味着临床医生必须从病人床边的工作地点走到办公室才能使用计算机。

230 雅各布·E·巴德拉姆

因此,使用计算机和电子病历系统 (EPR) 可以提高医院的流动性 (另见 Bellotti 和 Bly 1996)。从更专业的层面来看,临床医生 (包括医生和护士)并不认为“在特殊房间使用计算机”是其工作的一部分。他们的工作与患者的治疗和护理以及学生的教育有关。我们已经观察到,电子病历系统的引入迫使护士不得不坐在电脑前使用个人电脑 (图 8.1)。

(Bardram 2005c),这对他们来说并非典型的工作环境。他们不满意自己不再能在病人床边完成工作,而是不得不走到电脑前,登录,启动EPR系统,找到病人,找到记录或药物方案,并记录病人的治疗情况。

显然,通过无线局域网连接的笔记本电脑、平板电脑和PDA等移动设备在医院的使用日益增多 (例如,参见Bardram,Kjær和Nielsen 2003a;Munoz等人2003)。然而,在很多情况下,我们也看到了此类技术在使用中存在的问题。首先,笔记本电脑



图8.1
护士们在办公室的办公桌上工作。

普适的基于活动的计算 231

如果不将笔记本电脑和平板电脑放置在稳定的水平表面上,使用起来会非常困难。因此,在大多数已启用笔记本电脑的医院,它们会被安装在推车上,然后推着走,而平板电脑则通常放在病人的病床上。其次,目前市面上大多数移动设备的设计并不适用于像医院这样恶劣的环境,而且它们太脆弱,即使掉在地上也无法幸免。

例如,当临床医生洗手时,将平板电脑放在水槽边缘,平板电脑可能会掉落摔坏;或者设备经常会被各种液体弄湿,其中一些液体需要彻底清洗并用酒精消毒。最后,临床医生无法在工作的所有环节都使用移动设备,因此需要支持他们在工作流程中使用不同的设备。

同构设备上隔离

临床医生在执行一项活动时,会使用许多不同的计算机和设备四处走动。例如,如图 8.1 所示,当办公室里的一名护士起身给病人用药时,她的座位通常会被其他人占据。因此,当她看完病人回来时,需要找到另一台空闲的计算机,登录,启动 EPR 应用程序,找到病人,找到药物模式,滚动到相应的药物,并标记已给病人用药。这对她来说非常烦人且耗时,因为她刚刚在第一台计算机上花费时间和精力建立了这个视图,而这台计算机现在不幸被别人占用了。大多数计算机应用程序和底层中间件或操作系统很少或根本不支持在不同计算机之间传输用户会话,因此在工作日内必须不断地手动重新建立执行活动的计算环境。

问题在于应用程序在同类设备上孤立运行。

将一组应用程序或服务从一台计算机移动到另一台计算机是很困难的,而在不同类型的设备之间移动它就更加困难了,例如从 PDA 移动到大型台式计算机。

单用户任务

“个人电脑”及其操作系统是为单用户任务而设计的。然而,日常活动的核心在于协作性 尤其是在医院这样的工作场所。由于专业化

232 雅各布·E·巴德拉姆

医疗工作的本质在于,治疗和护理本质上是专科医生、护士、护理助理等人员之间的协作活动。在治疗佩德森夫人的例子中,放射技师拍摄X光片,放射科医生进行描述,医生根据图像、描述、血液检查结果和既往病史做出进一步治疗的结论。护士负责执行治疗,包括准备和给患者用药,并将其记录在病历中。因此,一项活动的组成动作通常由合作的临床医生负责(Bardram 1998)。在分析纸质记录的使用情况时,我们经常发现医生和护士同时查看和书写同一份文件。例如,医生使用药物方案开药,护士则使用药物方案记录患者用药情况,而他们两人并肩站立。使用电子病历(EPR)时,这种协同协作通常难以实现,因此需要使用两台个人电脑等。而且,当护士不在同一地点工作时,系统不支持护士将其“记录药品”操作与医生的“开药”操作关联起来。他们不共享应用程序。

因此,协作是临床工作的固有特质,而通常很少支持在参与活动的人员之间分配和聚合操作。目前,协作是由应用程序“外部”的专用应用程序支持的,这些应用程序可用于通信或应用程序共享。

对工作环境不敏感

计算机本质上对用户的工作环境不敏感。

因此,计算机无法在人机交互中考虑情境信息。这就是为什么图8.1所示的案例中,护士必须不断查找病人。计算机或EPR根本无法获取任何关于她工作情境的信息,包括她目前正在护理哪位病人。这种缺乏情境意识的情况在人机交互中变得更加具有挑战性。

在医院使用移动设备时,由于EPR等应用程序的工作环境不断变化,因此需要用户进行手动重新配置。

普适的基于活动的计算 233

基于活动的计算

为了缓解上述现代计算面临的挑战,我们引入了基于活动的计算(ABC)的概念。基于活动的计算是一种普适计算方法,专注于为移动、协作和分布式人类活动提供计算支持。我们认为,对整体活动而非单个任务的支持是普适计算的根源。当用户使用众多异构计算设备时,在活动层面支持用户的需求至关重要。如果每次切换到新的计算设备和/或活动时都必须重新安排应用程序和服务,那么在普适计算的世界中,我们将无法生存。此外,普适计算融合现有计算设备的理念使得这些设备需要根据用户的环境及其当前活动进行自我调整。

也就是他或她的活动。

基于活动的计算具有以下核心原则,每个原则
它解决了上面提到的挑战。

以活动为中心。“计算活动”将一系列支持用户执行特定“人类活动”所需的服务整合成一个连贯的集合。例如,在医院治疗患者的协作活动可以在ABC中建模为一种计算活动,其中包括显示和操作患者的用药方案、血液检查结果、近期X光片等服务。图8.2展示了这一原则,它展示了计算活动如何包含一组服务,每个服务处理一组特定的数据,例如文件、文档或服务器中的远程数据。这一原则解决了以应用为中心的计算的挑战,并通过允许用户在其参与的活动之间轻松切换来支持工作中断。

活动暂停和恢复:用户参与多个活动,可以通过暂停一个活动并恢复另一个活动来切换。恢复活动将恢复用户活动中的所有服务和数据。此原则解决了对中断支持不足的问题。

234 雅各布·E·巴德拉姆

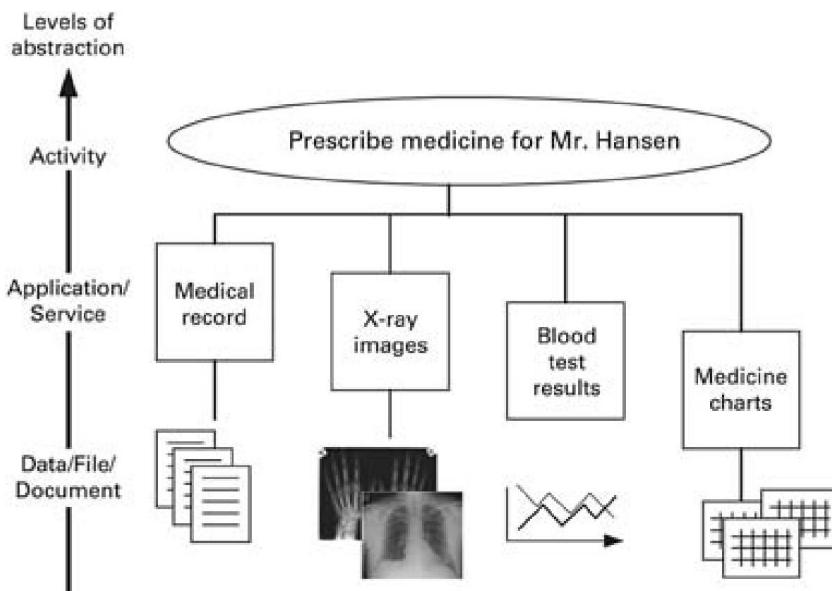


图8.2
单一活动涉及许多服务和应用程序,它们又会访问广泛的数据。

活动漫游:活动存储在分布式基础架构中。活动可以在一个工作站上暂停,然后在另一个地方的另一个工作站上恢复。这一原则解决了移动性挑战。

活动自适应:活动会根据其恢复设备上的可用资源进行自适应调整。这些资源包括网络带宽、CPU 性能和屏幕尺寸。因此,无论是在墙面大小的显示器上恢复,还是在 PDA 上恢复,活动的外观可能截然不同。这一原则解决了孤立且同质设备的挑战。

活动共享:协作用户通过拥有一个可以访问和恢复活动的参与者列表来共享活动。两个用户(例如上文中的护士和医生)可以同时进行同一活动,从而合作治疗同一患者。用户可以轮流进行活动,让一个用户接替另一个用户离开的活动;或者,他们可以同时、同地或远程协作。这一原则解决了个人电脑及其缺乏协作支持的挑战。

普适的基于活动的计算 235

情境感知:活动具备情境感知能力,也就是说,它能够根据使用情境进行调整。情境感知可用于调整用户界面以适应用户当前的工作情况,例如,显示当前正在接受治疗的患者的医疗数据。

或者,也可以从更技术的角度来理解,即活动的执行及其服务的发现,会根据其附近可用的资源进行调整。这一原则解决了上下文不敏感的挑战。

场景

让我们考虑一系列场景,以说明基于活动的计算如何在典型的一天中为医生提供支持。

A科室的医生们聚集在宽敞的会议室参加晨会。此次会议的目的是讨论一些常见的重症病例。一方面是为了征求第二意见,另一方面也是为了教育。会议室配备了两台大型壁挂式显示器,桌子上也内置了一个显示器。一些医生配备了PDA,但平板电脑已经不再使用,因为它们装不进白大褂的口袋里。然而,大多数医生更喜欢使用医院各处的公共显示器,而不是随身携带计算设备。

Christensen 医生开始为一位病情危重的癌症患者 Jensen 夫人 做演示。他准备了一个名为“Jensen 夫人演示”的活动,其中包含相关的医疗数据,例如血液检查结果的历史视图、全身扫描的 X 光片、病历和用药史。他走到一个墙面显示器前,自动登录,活动恢复,所有之前准备好的医疗视图立即显示出来。在演示过程中,一位资深医师开始使用桌上的显示器。由于他是该活动的参与者,因此他可以在桌上继续操作。这样就可以进行协作会话,其中一个显示器上的更改会反映在另一个显示器上。资深医师突出显示某个血液检查结果并询问相关信息,同时他的突出显示会反映在墙面显示器上。由于参与者在同一个房间,因此两个显示器之间无法建立语音连接。

会议结束后,Christensen 医生走向病房巡房。途中,一名护士打断了他,询问另一名护士的情况。

236 雅各布·E·巴德拉姆

病人。他拿起PDA,从活动列表中选择与该病人相关的活动。PDA上会恢复该活动,但由于PDA的屏幕尺寸和处理能力有限,只能恢复部分活动。例如,他无法查看X光片。因此,他走到走廊上的一个公共显示屏前,靠近显示屏即可登录并恢复当前活动。在这里,他可以观看X光片并帮助护士继续工作。在查房期间,Christensen医生和一名护士在病床边探望病人。他使用病床上的内置显示屏查找每位病人的医疗数据。当他靠近病人并登录时,电脑始终会建议恢复与病床上病人相关的活动。如果Christensen医生为其他病人恢复活动,而不是他正在探望的病人,电脑还会显示一个警告。

后来,在巡房期间,他收到放射科医生的邀请,邀请他参加一项活动。他通过PDA收到了通知。从描述中,他看到放射科医生正在分析他今天早上为詹森夫人安排的一些紧急图像。他赶紧跑到病房会议室,在墙上的显示器上继续进行这项活动。他进入了与放射科医生的实时活动共享环节,放射科医生为詹森医生提供了解答。

克里斯滕森。

下一节将描述 ABC 框架如何支持这些场景。

ABC框架

ABC 框架是基于活动的计算原则的当前实现。ABC 框架的主要目标是为开发和部署可用于基于活动的计算的计算机应用程序提供一个技术平台。

概念。

ABC 框架的组件可以分为三类:运行时基础架构、用户界面和编程模型。运行时基础架构是一组组件,它们通过适应特定环境中可用的服务或资源来处理管理分布式和协作活动的计算复杂性。

环境。用户界面使用户能够访问和操作活动，并在移动和协同工作环境中使用支持ABC感知的应用程序。编程模型是一组接口，用于构建新的ABC组件，这些组件可部署在运行时基础架构中。

ABC 运行时基础设施

本节描述了

ABC 框架。其与活动相关的职责包括：管理、存储、激活和分发活动；管理和分发共享状态信息；确保协作活动的同步方法；以及管理协作会话。图 8.3 展示了 ABC 运行时基础架构。它由运行在一台或多台服务器上的一系列服务器进程以及支持 ABC 应用程序执行的一系列客户端进程组成。

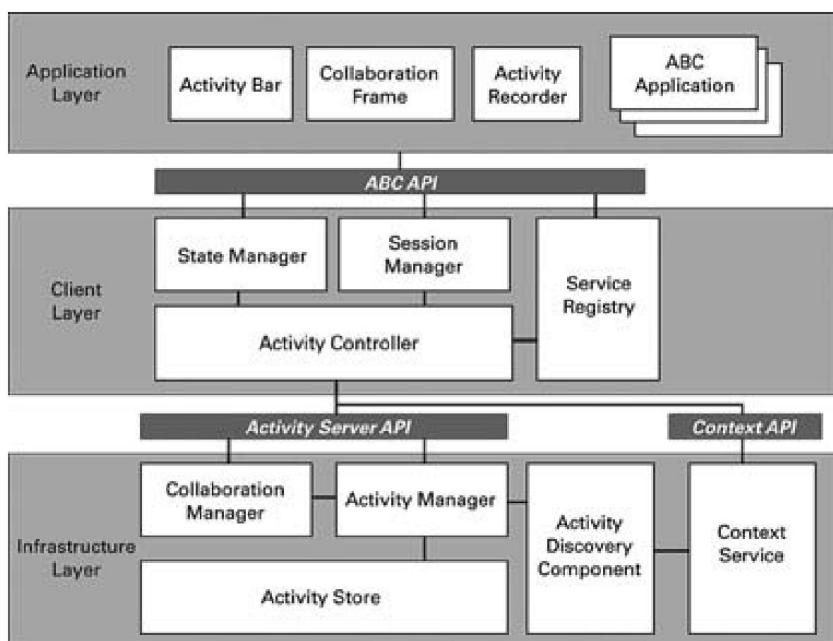


图8.3

ABC 运行时基础架构，分别展示了服务器端和客户端流程。

238 雅各布·E·巴德拉姆

ABC 基础设施的服务器部分采用可扩展的方式构建,因此组成活动服务器的不同进程可以部署在不同的主机上。ABC 基础设施由以下关键进程组成。

活动存储:提供创建、删除和访问活动的接口,并通过引用或查询新活动的模板来处理活动的持久化。该存储会跟踪用户当前正在参与的活动以及用户的使用历史记录,使用户可以在活动列表中前进和后退。

活动管理器:通过允许客户端创建、初始化、暂停、恢复和终止活动,管理活动的运行时行为。管理器跟踪在其上注册的 ABC 客户端,并提供订阅-发布-通知接口,用于通知客户端特定客户端正在运行的活动的相关变化。

协作管理器负责处理活动中活跃参与者之间同步协作的实时需求。为此,它管理每个正在进行的协作活动的会话对象,这些活动当前由一个或多个用户在不同主机上激活,包括同一用户在多个主机上激活。基本上,如果会话或其相关活动发生变化,会话会通知其活跃参与者。典型的变化包括用户在会话中的进入、移动和离开,以及活动状态的变化。有意监听会话变化的各方可以向会话添加一个会话监听器。**中央监听器**

会话对象是下面描述的客户端会话管理器。

上下文服务:上下文服务负责获取、存储和管理基础设施中的上下文信息。上下文服务通过上下文监视器(图中未显示)获取上下文信息,并为客户端提供请求-响应和基于事件的发布-订阅机制来访问此类上下文信息。该上下文感知基础设施构建了

基于 Java 上下文感知框架 (JCAF) (Bardram 2005a)。

活动发现组件(ADC)尝试代表用户发现相关活动。ADC 持续监控上下文服务的变化,并基于一组一阶逻辑规则创建新活动,并将其发送给活动管理器 (Christensen 2002)。

普适的基于活动的计算 239

活动控制器这是客户端和服务器之间的链接。

客户端的活动控制器在一个或多个活动管理器上注册，并维护与活动栏（ABC 基础架构的用户界面，参见图 8.5）的链接，活动栏通过控制器获取用户的活动列表。活动控制器还可以由活动管理器远程控制，例如，可以强制客户端更改用户。活动控制器还会收到来自服务器进程的相关事件通知。例如，如果当前用户被邀请参与另一个活动，活动控制器会收到通知，并通过活动栏向用户发出相应的信号。当活动控制器激活某个活动时，本地状态管理器会收到通知，状态管理器会使用注册表查找合适的ABC 应用程序，该应用程序可以处理活动中收集的服务。

在上述场景中，有一个服务器集群运行着活动服务器进程，包括活动存储、管理器和协作管理器，以及一个中央上下文服务。医院中部署的每个客户端，包括所有公共显示器和PDA，都运行着基础设施的客户端部分，包括活动控制器、状态管理器、会话管理器和服务注册表。能够处理不同服务请求的应用程序已在服务注册表中注册。每个用户都注册在上下文服务中，该服务以目录服务的形式运行。³当ABC客户端空闲时，它会显示空白屏幕。当用户使用用户名和密码或基于邻近度的用户身份验证（Bardram, Kjær 和 Pedersen 2003b）登录计算机时，该计算机上的活动控制器会加载用户活动列表并将其显示在活动栏中（参见下一节关于ABC GUI 的内容）。它还会向活动管理器请求用户的当前活动，并在客户端上恢复该活动。当活动恢复时，活动控制器会遍历该活动的服务集，并针对每个服务描述，询问本地服务注册表中是否有本地应用程序可以处理该服务描述。如果找到匹配的服务应用程序，状态管理器将获得该应用程序的句柄，然后状态管理器会生成一个单独的线程来启动该应用程序，并将服务描述的状态信息部分交给该应用程序。然后，该应用程序负责恢复服务的正确状态。例如，一个医疗模式应用程序应该显示正确患者的医疗数据，并滚动到模式中的正确位置。

240 雅各布·E·巴德拉姆

当用户选择另一个 Activity 或退出时,Activity 控制器会向状态管理器请求 Activity 的状态。状态管理器会遍历所有正在运行的应用程序,并为每个应用程序请求状态信息,然后将状态信息返回给控制器。这些状态信息

保存在活动中,交给活动管理器。管理器将活动存储在活动存储中,并更新历史记录。

这种基本状态管理机制还支持实时活动共享。如果一个活动的两个或多个参与者同时在不同主机上在线,则一个客户端上的状态变化将保存到服务器,然后服务器通过协作管理器将此状态变化广播给其他在线参与者。在每个客户端上,状态将如上所述进行管理。诸如语音链接和远程指针之类的协作小部件由客户端的会话管理器初始化、管理和最终确定。例如,远程指针在同一会话中的客户端之间以点对点的方式建立,并且不会作为状态信息复制到服务器(详情请参阅 Bardram 2005b)。

ABC 用户界面

图 8.4 展示了适用于台式电脑、平板电脑和壁挂式电脑的 ABC 用户界面。此屏幕截图展示了放射学会议活动的外观。主要的用户界面组件包括活动栏、协作框架、ABC 应用程序示例和远程指示器。

活动栏是用户界面的核心组件,用于访问 ABC 框架。图 8.5 详细展示了活动栏。从左侧开始,活动栏包含以下几组按钮:(i) “开始”按钮

用于启动 ABC 感知应用程序(在服务注册表中注册的应用程序);(ii) 两个按钮用于创建和完成活动;(iii) 两个按钮用于邀请参与者参加此活动和显示协作框架(图 8.4 中的 2 号);(iv) 两个按钮用于在活动历史记录中前进和后退,以及一个下拉框用于从活动列表中选择活动;(v) “灯”图标,用于通知用户列表中添加了新活动或现有活动发生变化;(vi) 一个按钮用于启动活动记录器,以及用于启用和禁用声音和麦克风的按钮;(vii) 登录按钮,显示当前用户的名称并可用于用户登录和退出。

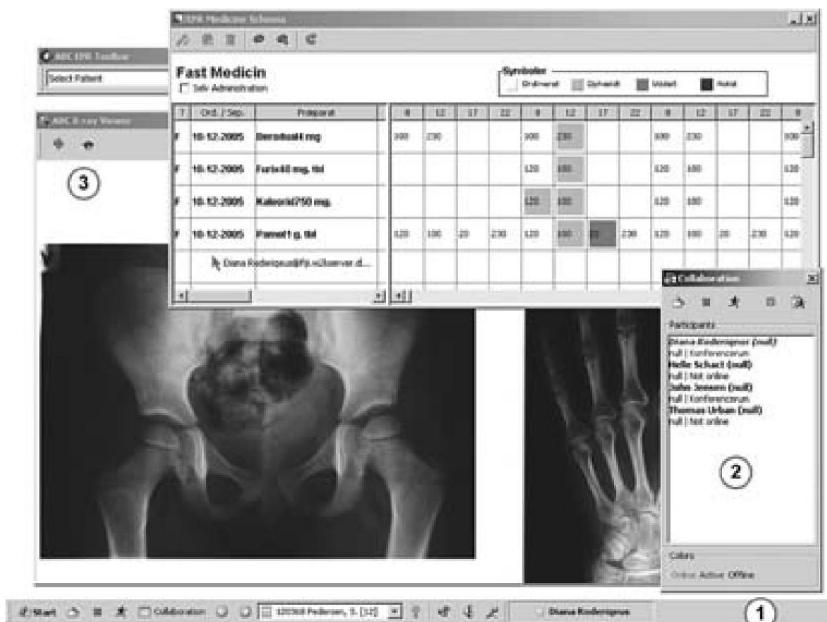


图 8.4

ABC 框架的用户界面包含底部的活动栏、右侧的协作框架、两个 ABC 应用程序（一个显示放射图像，另一个显示患者的病历）以及来自远程用户的远程指示器。活动名称显示在活动栏的选择框中，表明该活动与一位名叫 Pedersen 女士的患者有关。

让我们研究一下 ABC 用户界面如何支持前面提到的基于活动的计算的核心原则。

以活动为中心如图 8.2 所示，活动由一组服务组成，这些服务同样操作一组数据。在用户界面中，可以通过活动列表立即访问活动，也可以使用前进和后退按钮在活动历史记录中向前或向后移动。用户始终在活动中工作，也就是说，始终有一个活动处于恢复状态。我们称之为“活动活动”。当用户登录时，上次使用的活动将被恢复，并恢复到之前暂停时的状态 可能在另一台设备上。

服务映射到应用程序。在图 8.4 中，“放射图像查看器”服务映射到“ABC X 射线查看器”应用程序（编号 3）。

242 雅各布·E·巴德拉姆

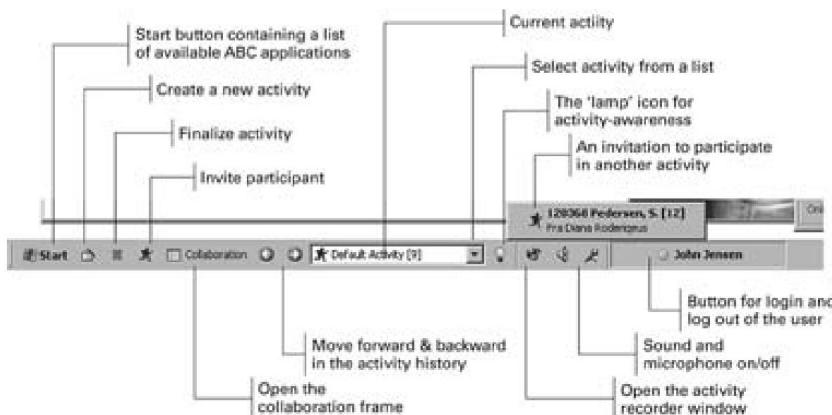


图 8.5
活动栏。

(如图 8.4 所示)。支持特定服务的应用程序能够传递与该服务相关的信息,并相应地重建其状态。这包括访问数据元素,无论它们是存储在活动中,还是通过分布式文件系统或服务器访问。在 X 射线查看器的案例中,X 射线图像存储在医院图像服务器上,活动保存着关于在何处以及如何访问这些图像的状态信息。大多数应用程序的状态信息还包括窗口的大小和位置。

通过从“开始”菜单启动服务,可以将服务添加到活动中;通过关闭窗口,可以将服务移除。活动暂停时,所有状态信息(包括数据引用)都将存储在活动中,并发送到底层基础架构中的活动管理器。

活动漫游:由于活动通过活动存储持久存储在底层基础架构中,因此活动可以分布在不同的支持 ABC 的设备上。活动漫游由一组生命周期事件控制:

注册表 当ABC 客户端启动时,该客户端会在活动管理器中注册。

登录 当用户登录时,客户端会向活动管理器请求该用户的活动列表。

普适的基于活动的计算 243

恢复 当用户恢复某个活动（例如，从活动列表中选择该活动）时，将从活动管理器中获取该活动，并将其服务映射到本地可用的应用程序，这些应用程序将根据活动中的状态信息启动和恢复。

暂停 当活动暂停时，所有服务将返回到其当前状态，该状态存储在活动中并交给活动

经理。

注销 当用户注销时，活动将被暂停，并且用户的活动将从活动列表中删除。

取消注册 当ABC客户端停止时，它将在活动管理器中取消注册。

如果用户在两台相同的设备之间漫游（例如，在两台台式电脑之间），则活动将恢复到完全相同的状态，包括窗口大小和位置。所有参与的临床医生都认为此功能至关重要，因为它使他们能够在医院内部移动时保持其工作空间的原有外观和感觉。现有客户端-服务器系统的主要抱怨之一是，在计算机之间移动时恢复用户会话会产生大量开销，因为服务器上只存储临床数据，而不存储用户会话。恢复所有窗口的精确大小和位置的主要缺点是，台式电脑（以及类似平板电脑的设备）的显示分辨率可能不同，范围从 1024×768 到 1600×1200 。因此，当您从大屏幕移动到小屏幕时，可能会出现部分或全部窗口不可见的情况。然而，实际上，临床医生并不认为这是一个问题，因为医院里的所有机器和屏幕通常都是同一类型的。尽管如此，这是我们当前工作中正在解决的问题。

活动适配从用户界面的角度来看，活动适配由本地运行并实现不同服务的应用程序处理。如果本地应用程序映射到某种服务类型，则会向该应用程序提供服务状态信息，并通过解析此状态信息来决定如何在特定设备上恢复服务。

在某些设备上，可以使用或调整窗口大小和位置（例如，调整以适应屏幕分辨率）；在其他设备上，此信息

244 雅各布·E·巴德拉姆

可能会被忽略（例如，在全屏显示所有服务的 PDA 上）；某些服务根本无法支持，如图 8.4 所示的 X 射线查看器应用程序在 PDA 上不可用。

活动共享通过将多个参与者与同一活动关联来支持活动共享。协作框架（图 8.4 中的 2）列出了当前活动的参与者。这些参与者之间的协作通过三种方式支持。异步协作通过允许参与者轮流恢复活动来支持，也就是说，参与者可以轮流处理某个活动。由于状态保存在活动中，因此一个参与者可以在另一个参与者暂停活动时完全接管该活动。此外，通过使用活动漫游机制，不同的用户可以在不同的地方恢复活动。为了允许参与者之间进行简单的通信，存在活动聊天（图 8.4 中未显示）。聊天特定于活动，并保存参与者之间的对话。此聊天作为活动状态信息的一部分永久保存。

如果两个或多个参与者同时在不同设备上恢复相同的活动，则会发生同步协作。在这种情况下，活跃的参与者将参与由服务器端的协作管理器和客户端的会话管理器处理的同步会议会话。协作机制确保所有参与用户之间的活动（包括其状态信息）同步。从用户界面的角度来看，这意味着用户界面状态信息（包括窗口位置、大小以及各个服务的状态）是同步的。除了在参与方之间同步用户界面状态信息外，协作机制还包括参与方之间的语音链接以及远程指针。

图 8.6 展示了同步协作活动共享的用户界面支持，其中展示了一个 ABC 应用程序的顶部框架和两个远程指针。由于我们希望支持同时在同一活动中活动但关注活动不同部分的用户，ABC 框架并不强制执行严格的“所见即我见”（WYSIWIS）原则。因此，两个不同的用户可以在两个重叠的窗口中保持焦点，而不会互相干扰。每个窗口的顶部框架仅显示每个用户当前关注的窗口。

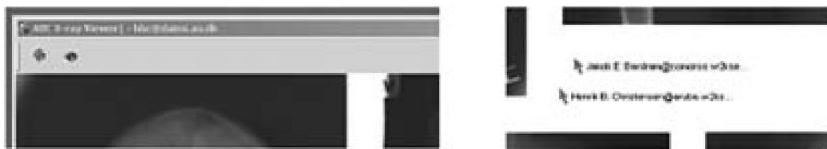


图 8.6

ABC 用户界面中的协作小部件。左侧窗口顶部框架装饰有当前窗口焦点用户的颜色和用户名。右侧显示了两个远程指针。

通过显示用户名并用用户特定的颜色装饰框架来实现。这样,一个用户可以说“看看这个”,另一个用户就能知道正在访问哪个窗口,并可以将焦点移到该窗口。远程指针会显示用户名及其计算机的主机名。我们之所以附加主机名,是因为同一个用户可以同时在不同的客户端上执行相同的操作。事实证明,这项功能在我们的评估过程中被频繁使用。

同步协作在医院中非常常见,尤其是在会议场合。ABC 框架允许在多台计算机上进行临床桌面会议。这可以在同一地点进行,例如上文提到的场景中,一组医生使用不同的设备参加会议。或者,它也可以发生在参与者彼此分离的情况下,例如,无法参加“真实”放射学会议的医生可以通过参与放射学会议室中运行的“放射学会议”活动来远程收听。在 ABC 框架中,这意味着远程医生可以看到放射学会议室大屏幕上显示的 X 射线图像,可以通过远程指示器看到放射科医生使用鼠标的手势,并且可以通过语音连接收听。然而,医生也是会议的积极参与者,他们可以提问,使用远程指示器指示 X 射线图像的区域,还可以重新排列或从 EPR 中调出新的医疗数据。

ABC 框架支持的第三种协作类型是时间协作。时间协作是同步和异步协作的混合,它允许参与者跨时间协作,就像他们在同一时间一样。

246 雅各布·E·巴德拉姆

时间。在用户界面中,ABC 备忘录播放器支持时间协作,如图 8.7 所示。该备忘录播放器是一个活动记录器,能够记录活动的进展,并捕获活动状态信息、鼠标事件和声音。从技术上讲,该活动记录器使用与同步桌面会议相同的机制,但不是将状态事件、鼠标事件和声音传输到另一台计算机,而是将这些数据传输到存储在活动存储中的持久数据对象。因此,活动记录器可用于为活动中的其他参与者录制多媒体消息,这些消息稍后可以从活动管理器中获取并在同一台或另一台计算机上重播。然后,其他参与者可以通过记录他们继续使用该活动,同时进行出声思考来回复。在图 8.7 所示的示例中,第一个用户 (Jakob E.)

Bardram) 发起了一场讨论,随后另外两位用户 (John Jensen 和 Diana Roderiqueus) 做出了回应,现在又回到了第一位用户的讨论。

在大多数医疗工作中,时间协作至关重要。通常很难确保两名或多名临床医生能够同时开会,而且医院使用各种各样的留言机制。

这些机制包括录音机、便利贴和答录机。活动记录旨在支持此类活动内的信息传递,从而确保信息在正确的活动情境中被记录和回放。通过这种方式,临床医生可以留下与他们合作的特定任务直接相关的多媒体信息。



图 8.7
活动记录器,用于记录和重放活动的展开。

普适的基于活动的计算 247

活动和情境感知：一旦我们通过活动建模来支持医院的工作，提供对活动展开的感知就变得至关重要。在基于活动的计算环境中，保持对活动进展、其他参与者正在做什么以及是否存在需要注意的问题的周边感知变得至关重要。当前的用户界面通过两种机制支持这种活动感知；一种是图 8.5 所示的灯图标，另一种是向用户启用活动的手机发送消息的功能。灯图标用于通知用户当前活动以外的活动的变化。如果当前用户收到新活动、被邀请参与其他活动或活动中添加了新的记录，灯图标将亮起并播放声音。这些事件也会发送到用户的手机上，以便在用户不使用电脑时提供简单的活动感知。然而，手机仅支持显示基本的活动信息，因此仅用于查看已更改的活动。然后，用户可以决定在附近的计算机上继续此活动，以便仔细查看或参与活动共享会话。诚然，这种对活动感知的支持相当有限，我们目前正在努力扩展它。

如图 8.3 所示，上下文服务是 ABC 基础设施的核心组件。上下文信息添加到此上下文服务中

来自各种来源，包括 ABC 客户端，它们会提供有关谁在不同计算机上登录以及当前正在进行的活动的信息。其他上下文信息（例如位置和状态）来自其他来源。此类上下文信息显示在

协作框架如图 8.4 所示。

然而，上下文信息最有趣的用途是活动发现（Christensen 2002）。图 8.3 中的活动发现组件（ADC）存储了一组一阶逻辑规则，这些规则会根据上下文服务中可用的上下文信息不断进行评估。ADC 能够根据上下文的变化识别不同的典型活动。

例如，如果一位护士在药房为一位特定病人领取了药盒，那么 ADC 就会推断出包含该病人医疗信息的活动对该护士有用。然后，ADC 使用活动工厂为用户当前的

248 雅各布·E·巴德拉姆

并将其推送到活动管理器。如果护士在线，活动管理器会通知她的 ABC 客户端，在用户界面中，护士将看到活动栏中的通知灯亮起，并听到一声小提示音。这会通知她新的活动，点击灯图标即可查看并恢复建议的活动。为了支持应急情况（例如，护士为两位患者拿着两个药盒），可以创建更多活动，并在点击灯图标时列出。这样，护士可以选择与她当前情况最相关的活动，或者选择不使用任何建议，而是保留当前活动。

因此，我们努力使活动发现尽可能不造成干扰，同时仍然通知用户。我们称之为非侵入式设计

情境感知。

ABC 编程模型

运行时基础架构既支持编程模型，又能利用它。该编程模型旨在帮助程序员通过添加新类型的活动、组件、应用程序或协作小部件来扩展 ABC 框架。编程模型包含一系列接口，程序员可以实现这些接口并将其添加到运行时基础架构中。这些接口共同构成了 ABC 框架中包含的标准 ABC 功能的分布式扩展。构成 ABC 编程模型的接口包括：

- Activity 接口定义了创建自定义活动类型的方法。例如，在我们基于 ABC 的电子病历实现中，有一个 EPRActivity 接口，它能够处理 EPR 相关的活动，包括与特定患者相关的活动。

- 活动存储、活动管理器和协作管理器接口构成了活动服务器的接口。通常，应用程序员会使用客户端层的活动控制器访问这些接口，但这些接口也可供程序员创建自己的客户端层功能或新的用户界面。

- 有状态应用程序接口，使程序员能够创建可以参与 ABC 运行时基础架构的客户端应用程序。

普适的基于活动的计算 249

•事件和通知接口,用于订阅某些组件的变更。最常用的是用于监听活动变更的 Activity Listener 接口、用于监听协作实时会话变更的 Session Listener 接口,以及用于监听上下文信息变更的 Entity-Listener 接口。

•会话管理器和会话接口,可以根据基础设施中的实时协作支持中的特殊目的进行定制。

ABC 框架中一个关键的设计不变性是应用程序是有状态的,这意味着它们可以交接和恢复自身的状态。运行时基础架构负责收集、管理、分发和同步这些状态信息,无论是在用户在物理机器之间移动,还是参与同步协作会话时。环境中运行的所有应用程序的状态信息集合都保存在活动中,因此可以假设活动始终包含共享状态。状态管理器保证了这一不变性:它是一个在客户端运行的单例进程,它在活动控制器(以及作为服务器进程运行的活动管理器)和在客户端机器上运行的应用程序之间建立链接。

该编程模型提供了接口以及有状态应用程序和 UI 组件的默认实现。为了帮助应用程序程序员构建能够处理状态信息的 ABC 感知应用程序,该编程模型包含有状态用户界面组件。在基于 Java 的 ABC 框架版本中,这些有状态用户界面组件是 Swing 组件的包装器 (Bardram 2005b)。例如,我们扩展了核心 Swing 组件(例如 JFrame、JScrollPane 和 JComboBox),以设置和获取状态信息。

这些用户界面组件旨在帮助应用程序员轻松实现状态管理。使用这些 ABC Swing 组件,程序员无需担心用户界面状态,只需管理特定于应用程序的状态信息即可。

实施情况

上面描述的 ABC 框架是版本 3,已在 Java 2 标准版 1.4 (J2SE) 中实现,使用 Java RMI 作为其分发机制,并使用 Java 媒体框架 (JMF) 作为

250 雅各布·E·巴德拉姆

设备之间的音频广播;所有支持 ABC 的应用程序均基于 Java Swing 用户界面框架编写。我们创建了一个专门的 ABC Swing 库,方便程序员创建支持 ABC 的应用程序 (Bardram 2005b)。由于 Java 提供的平台独立性,版本 3 可在 Microsoft Windows 和 Linux 上运行。图 8.4 所示的 ABC 客户端界面设计用于在墙面大小的显示器、平板电脑和台式电脑上运行。

版本 3 并不完全支持小型设备 (Java 2 Micro Edition,J2ME) ,例如 PDA 或手机。然而,精简版客户端可以在 PDA 和手机上运行,仅显示活动的基本信息 (活动名称、参与者和所涉及的服务)。用户可以在小型设备上激活活动,当用户接触到完整的 ABC 客户端 (例如墙面显示器) 时,该活动会恢复。版本 3 不支持 Word、Emacs 或 PowerPoint 等原生应用程序,所有支持 ABC 的应用程序都需要使用 ABC 编程 API 进行开发 (详情参见 Bardram 2005b) ,或者至少需要使用 ABC API 制作原生应用程序的包装器。

目前,我们正在基于.NET框架实现ABC框架的第4版。此版本已集成到Windows操作系统中。例如,我们将用我们自己的“活动栏”替换Windows任务栏,并支持将原生Windows应用程序纳入ABC框架。此外,Java RMI中紧密同步的通信范式已被松耦合、异步发布-订阅的基础架构所取代,这使得其对常见故障和异常的鲁棒性更高。我们还致力于为PDA和手机实现ABC客户端,使其能够参与活动漫游和活动共享。

讨论

直接评估一个运行时基础设施及其相应的编程框架非常困难,尤其是在研究全新类型的普适计算技术时 (Abowd 和 Mynatt,2000)。然而,为了评估基于活动的计算的概念原理及其技术实现是否真的能够帮助用户管理复杂的普适计算环境,

普适的基于活动的计算 251

我们在 ABC 框架之上实现了电子病历，并在与奥胡斯大学医院的临床医生进行的一系列设计和评估会议中使用它。我们举办了 11 场这样的研讨会，要求临床医生用一整天的时间共同设计、使用、评估和测试该框架。我们设计研讨会中的一种常用方法是让临床医生角色扮演各种临床场景（Bødker 和 Christiansen 1997），尝试不同的设计方案。在实时活动共享的设计中，我们应用了与群件演练（Pinelle 和 Gutwin 2002）非常相似的演练方法。此外，我们还与从未见过 ABC 框架或未接触过基于活动的计算概念的临床医生进行了四场全天的评估研讨会。所有研讨会均进行了录像，随后对录像进行了分析，对“有趣的”概念和可用性问题进行了分类。

我们举办的一系列评估研讨会的总体印象是，临床医生对基于活动的计算的基本概念给予了非常积极的反馈。该计算平台支持移动性、中断、并行工作、协作以及基于情境感知的用户界面自适应，旨在解决他们日常临床工作中面临的一些核心挑战（Bardram 2004）。

在此过程中，我们不断汲取各种改进意见和建议，最终形成了当前版本的 ABC 框架，这体现了我们与众多临床医生密切合作所积累的大量设计知识。然而，在评估过程中，我们也发现了当前基于活动的计算支持在设计和实施方面的局限性。我们想在此更详细地讨论其中的一些局限性，因为它们与我们目前正在改进基于活动的计算支持的工作息息相关。

在我们的评估过程中，如何区分不同的活动是一个反复出现的问题：“为汉森太太开药”什么时候不再是处方活动，而变成了“记录为汉森太太开的药”？在很多情况下，我们观察到一个活动会直接转换为另一个活动，而无需用户选择或创建新活动。因此，“为汉森太太开药”活动有时也会演变成“为佩德森先生检查药品”活动，因为护士只需在“病人”列表中选择佩德森先生作为当前病人即可。

252 雅各布·E·巴德拉姆

EPR,尽管她正在从事致力于汉森夫人的活动。

已经设计了几种解决这些“活动分离”问题的建议（Bardram 2004）。这些建议包括使用活动模板、事后创建活动（而不是在开始使用活动之前创建）、在活动展开时为其添加书签（这可以自动完成，例如，当用户在患者之间切换时），以及使用编程模型来建模 EPR 活动。这样的活动可以确保活动与患者之间的紧密联系，从而帮助用户避免在活动进行过程中切换患者。这在评估中被认为是相当关键的问题。

从理论角度来看，区分一项活动与另一项活动的问题与活动理论中识别现实世界活动的问题密切相关。活动理论的核心在于能够分析性地区分一项活动与另一项活动。这基本上是通过考察活动的动机或目标来实现的。因此，询问人们“为什么”做某事可以揭示个体活动的特性。在临床环境中，目标通常与特定患者的治疗和护理相关，因此，创建与患者相关的计算活动的技术方案似乎是合适的。

从这个角度来看，汉森夫人的“处方”和“记录”究竟属于两种不同的活动，还是同一种活动中的两个动作，这一点值得怀疑。如果

如果是后者 我们也相信是这样 那么我们所说的“活动”

从活动理论的角度来看，ABC 框架中的“行动”或许可以正确地称为“动作”。这也与在协作人员之间分配活动中的行动的概念相呼应，其中医生负责开药，护士负责记录。

将活动彼此分离的问题也与可扩展性问题息息相关（Bardram 2004）。在现实环境中，临床医生可能参与数十甚至数百项活动。ABC 框架中基于活动的计算原则的当前实现在其用户界面上无法扩展。例如，活动栏中的活动列表很快就会变得过长，不切实际。目前活动的线性排序在概念上也无法扩展。用户如何理解如此多的活动，却无法将它们彼此关联或与某些上下文信息联系起来？这存在潜在的危险。

我们只是将导航和管理大量数字数据的负担从更传统的工具（例如电子病历）转移到基于活动的计算框架。

这些在分离活动、活动与行动之间的关系以及活动的可扩展性方面的经验和理论挑战使我们思考如何改进基于活动的计算。

我们当前的建议集中在三个设计理念上：(i)在基于活动的计算支持中表示人类意图，(ii)支持活动空间中的关系和观点，以及 (iii)支持本机应用程序。

正如我们表征人类活动一样，我们认为将人类活动的目标作为活动计算表征的一部分也是值得的。显然，这只能弱化活动人类动机的表征，但它是一种外化，有助于用户管理活动。此外，共享（即外化和内化）活动的共同目标在协同工作中至关重要，在协同工作中，协作者会根据共同目标协调各自的行动（Leont'ev 1978；Bardram 1998）。但表征活动目标最有前景的用途在于支持主动性 让计算机在与用户的关系中保持主动而非被动。主动性和适应性是普适计算必不可少但又充满挑战的方面。超越当前对情境感知的支持，并将主动适应性建立在活动意图或目标的表征之上，似乎是基于活动的计算迈出的有希望的一步。例如，我们正在致力于扩展情境感知功能，当接近某位患者时显示默认的患者活动，以显示一系列相互关联的活动，这些活动都与该患者的治疗有关。

支持共享活动空间旨在帮助用户组织、管理和关联大量相互关联的活动（其中一些可能是行动，即被纳入其他活动）。我们目前的工作是设计一个超媒体结构，其中包含一个由活动和相关行动组成的网络 即一个由相互关联的活动和行动组成的大型网络，用户可以在其上应用不同的视角。目前，ABC 框架仅支持用户特定视角来查看可用的活动 即用户可以获取其活动列表。此视角可以扩展以支持与时间、

254 雅各布·E·巴德拉姆

地点、背景、患者、同事或疾病类型。此外，我们还在创建对复制、克隆、合并、拆分和链接活动的支持。

最后，出于一些经验和理论原因，对原生应用程序的支持似乎是理所当然的。经验评估表明，用户在调整或使用不熟悉的电子病历（例如，我们在 ABC 框架之上实现的病历）时会遇到问题（Bardram 2004）。尽管 ABC 框架及其编程模型可以被视为在特定平台（例如操作系统）上编写用户应用程序的一组新的基础类，但仍然需要思考如何将现有应用程序和系统集成到基于活动的计算平台中。因此，从经验和实践的角度来看，对原生应用程序的支持至关重要。从理论角度来看，这一论点也得到了支持，因为与熟悉的应用程序保持一致有助于用户将活动理解为活动操作层面的例行程序。然而，从更技术的角度来看，处理未构建为支持基于活动的计算的现有应用程序可能相当繁琐。例如，在许多应用程序中获取和设置状态信息相当困难；跨异构设备迁移一个应用程序很困难；在基于活动的普适计算环境中使用为桌面使用而制作的应用程序也很困难。

因此，我们当前的工作致力于双重策略，既尝试对遗留应用程序进行技术修复，又设计用于构建基于本机活动的应用程序的编程模型和一组基础类。

结论

在本章中，我们介绍了基于活动的计算方面的工作。基于活动的计算这一概念旨在将日常使用的计算技术从双重意义上移出桌面——既在物理上远离当今大多数计算机所处的桌面，也在概念上远离支持单个应用程序的桌面用户界面隐喻。许多关于我们所谓的基于活动的计算的建议已被提出和研究，这种方法包含了许多有趣的建议，可以解决大多数问题。

普适的基于活动的计算 255

基于活动的计算似乎是创建超越桌面的未来计算基础设施和操作系统的新型设计理念的良好候选者。基于我们在设计、实现和评估此类基于活动的计算基础设施方面的经验，我们提出了基于活动的计算的六项原则，它们支持：(i)以活动为中心的应用程序、服务和数据收集；(ii)活动的暂停和恢复；(iii)活动在分布式计算设备之间漫游；(iv)活动适应异构计算设备上的可用资源；(v)在同一活动中的多个参与者之间共享活动；以及 (vi)通过使活动能够适应其执行上下文来实现上下文感知。

基于我们让大量临床医生评估我们的ABC框架的经验，我们认为，应对分离活动和处理大量活动的挑战，对于进一步发展基于活动的普适计算方法至关重要。正如在Rooms系统 (Henderson和Card,1986)的早期研究中所讨论的那样，这些挑战也出现在虚拟桌面的使用中，用户在设置哪些应用程序实际上属于哪个房间时遇到困难的情况并不少见。例如，这也可能适用于Kimura (MacIntyre等人,2001)和Aura (Sousa和Garlan,2002) 等基于任务的方法，其中提到了“意图”的概念，但其似乎并未在计算机技术中发挥任何作用。

同样，工作流系统也常常因其严格区分活动而受到批评。这与现实世界的活动并不相似，现实世界的活动通常高度关联，没有严格的界限，并且往往服务于多种目的（即，根据活动理论，活动具有多重动机；参见 Kaptelinin 1996）。因此，研究如何将对活动的支持与对此类界限模糊的相互关联活动的支持结合起来，是基于活动的普适计算方法的核心挑战。

基于活动的计算的概念和技术源于我们为大型医院的临床工作设计普适计算基础设施的实验研究。正如引言中所述，这对于计算技术来说是一个极具挑战性的环境，因此非常适合研究普适计算架构和平台。临床工作的特点是

256 雅各布·E·巴德拉姆

仅治疗一名患者就需要处理海量医疗数据,而这些数据多达数千个;高度的移动性;许多并行且间断的工作活动;高度的协作;众多医疗应用程序和数字资料的使用;以及众多异构设备的使用。电子病历建立在现有计算机技术(操作系统和中间件层)之上,因此通常是根据这些当代技术所体现的桌面计算模型设计的。因此,当今的临床系统无法支持上述临床工作的核心方面,因此在日常使用中往往显得力不从心。

然而,我们确实相信,基于活动的计算在医院等医疗环境之外也是一种可行的计算原则。许多工作环境的特点是处理大量数字数据、移动性、并行和中断工作以及协作。即使在办公室环境中,基于活动的计算支持也可能非常有用,我们相信,即使移动性在办公室环境中并不总是那么普遍,它作为一种编程环境也会大有裨益。

理论上,我们主张,尽管基于活动的计算名为“基于活动的计算”,但它并非另一种工作流系统 恰恰相反。基于活动的计算中的“计算活动”是指在特定活动中收集、管理、分发和共享相互关联的材料和工具的一种手段。随着人类活动越来越多地涉及数字材料的处理,人们迫切需要能够帮助用户以反映活动本身的方式管理海量数字材料的计算工具。这种需求的一个基本部分包括支持协作人员之间行动的分配和集成。因此,在这种超越当今个人桌面计算模式的普适计算平台中,对基于活动的合作的支持至关重要。

致谢

我们衷心感谢所有为我们贡献宝贵时间的临床医生。Henrik B. Christensen 参与了基于活动的计算(当时称为“活动中心计算”的一些初步构想的概述,Claus Bossen 则参与了医院工作的实地研究。这项工作由丹麦中心资助。

信息技术研究 (CIT) 和 ISIS Katrinebjerg 能力中心。目前,ABC 项目由丹麦研究委员会在 NABIIT 项目下支持。

笔记

1. Norman 提到了对“共享活动空间”的支持。然而,由于这些概念从未实现,因此很难判断其确切含义。Rooms 系统不支持设备间的房间移动或任何形式的合作。
2. 有关活动理论、计划、情境行动和工作流系统之间的关系的详细讨论,请参见 Bardram 1997。
3. 这是我们原型解决方案中实现的。在实际部署场景中,我们显然需要与真实的目录服务器配合,并使用 LDAP 接口等实现互操作性。
4. 我们借用了 Windows 任务栏中的“开始”图标,以帮助用户识别其用途。

参考

Abowd, GD 和 Mynatt, ED (2000). 绘制普适计算的过去、现在和未来研究。ACM 计算机人机交互学报 (ToCHI) 7 (1): 29–58。

Bardram, JE (1997). 计划作为情境行动:一种活动理论方法应用于工作流系统。载于第五届欧洲计算机支持协同工作会议论文集,第17-32页。英国兰开斯特,9月7-11日。

Bardram, JE (1998). 协作、协调与计算机支持 一种基于活动理论的计算机支持协同工作设计方法。奥胡斯大学计算机科学系博士论文。Daimi PB-533。

Bardram, JE (2004). 基于活动的计算 原理、实施与评估。奥胡斯大学普适医疗中心技术报告。已于 2004 年 4 月提交至 ToCHI。

Bardram, JE (2005a). Java 上下文感知框架 (JCAF) 一种用于上下文感知应用程序的服务基础设施和编程框架。

刊载于《普适计算:第三届国际会议》(PERVA-SIVE 2005)论文集,第98-115页。德国慕尼黑,5月8-13日。

Bardram, JE (2005b). 基于活动的计算:支持移动性和协作。普适计算 个人与普适计算 9 (5): 312–322。

Bardram, JE (2005c). 登录的麻烦:论可用性和计算机安全性。

普适计算 个人与普适计算 9 (6): 357–367。

258 雅各布·E·巴德拉姆

Bardram, JE 和 Bossen, C. (2005). 移动办公:医院协作的空间维度。计算机支持协同工作14 (2): 131–160。

Bardram, JE, Kjær, TK, 和 Nielsen, C. (2003). 通过异构设备间应用漫游支持医疗保健领域的本地移动性。第五届移动设备与服务人机交互国际会议论文集,第161-176页。意大利乌迪内,9月8日至11日。

Bardram, JE, Kjær, RE, 和 Pedersen, M. Ø. (2003). 情境感知用户身份验证 支持普适计算中基于邻近度的登录。载于《Ubicomp 2003:普适计算》论文集,第107-123页。华盛顿州西雅图,10月12-15日。

Bellotti, V. 和 Bly, S. (1996)。《告别桌面电脑:产品设计团队中的分布式协作与移动性》。载于1996年 ACM 计算机支持协同工作会议论文集,第 209-218 页。

马萨诸塞州波士顿,11月 16 日至 20 日。

Bødker, S. (1991). 通过界面:基于人机交互的用户界面设计方法。Hillsdale , NJ: Lawrence Erlbaum。

Bødker, S. 和 Christiansen, E. (1997). 场景作为设计的跳板。载于 Bowker, G.,Gasser, L.,Star, L. 和 Turner, W. 编,《社会科学研究、技术系统与合作工作》,第 217-234 页。新泽西州希尔斯代尔:劳伦斯·埃尔鲍姆出版社。

Bødker, S.,Ehn, P.,Kammersgaard, J.,Kyng, M. 和 Sundblad, Y. (1987)。乌托邦式体验:为熟练的图形工作者设计强大的计算机工具。收录于 Bjørknes, G.,Ehn, P. 和 Kyng, M. 编,《计算机与民主:斯堪的纳维亚的挑战》,第 251-278 页。奥尔德肖特:阿弗伯里出版社。

Christensen, HB (2002). 利用逻辑编程检测普适医疗中的活动。载于国际逻辑编程会议论文集。丹麦哥本哈根,7月29日至8月1日。

Christensen, HB, and Bardram, JE (2002). 支持人类活动

探索以活动为中心的计算。载于《Ubicomp 2002:普适计算》论文集,第107-116页。瑞典哥德堡,9月29日至10月1日。

Carroll, J. (ed.) (1995). 基于场景的设计:系统开发中的工作与技术展望。纽约:John Wiley and Sons。

Engeström, Y. (1987). 拓展学习:发展研究的活动理论方法。赫尔辛基:Orienta-Konsultit Oy。

Grønbæk, K.,Gundersen, K.,Mogensen, P. 和 Ørbæk, P. (2001)。交互式房间支持复杂和分布式设计项目。在 Interact 01 会议记录中,第 407-414 页。日本东京,9 月 1 日至 5 日。

Henderson, JA 和 Card, S. (1986)。房间:使用多个虚拟工作空间来减少基于窗口的图形用户界面中的空间争用。

ACM 图形学学报 (TOG) 5 (3) :211-243。

Kaptelinin, V. (1996). 计算机介导的活动:社会与发展情境中的功能器官。载于 Nardi, B. (主编), 《情境与意识:活动-

人机交互理论与人机交互,第 45-68 页。马萨诸塞州剑桥:麻省理工学院出版社。

Leont'ev, AN (1978).活动、意识与人格。新泽西州恩格尔伍德克利夫斯:普伦蒂斯霍尔出版社。

MacIntyre, B., Mynatt, ED., Voids, S., Hansen, KM., Tullio, J. 和 Corso, GM (2001)。利用交互式外设显示器支持多任务处理和背景感知。载于 ACM 用户界面软件与技术 2001 会议论文集 (UIST01),第 11-14 页。佛罗里达州奥兰多,11 月 11-14 日。

Munoz, M., Rodriguez, M., Favela, J., Gonzalez, V. 和 Martinez-Garcia, A. (2003)。医院中的情境感知移动通信。IEEE 计算机 36 (8): 60-67。

Norman, DA (2000)。《隐形计算机:好产品为何会失败?个人电脑为何如此复杂?信息设备才是解决方案》。马萨诸塞州剑桥:麻省理工学院出版社。

Norman, DA, and Draper, S. (eds.) (1986)。以用户为中心的系统设计。新泽西州希尔斯代尔:劳伦斯·埃尔鲍姆。

O'Conaill, B. 和 Frohlich, D. (1995)。工作场所的时间空间:如何应对干扰。载于 SIGCHI 计算机系统人为因素会议论文集,第 262-263 页。科罗拉多州丹佛市,5 月 7 日至 11 日。

Pinelle, D. 和 Gutwin, C. (2002)。群件演练:为群件可用性评估添加上下文。载于 SIGCHI 计算机系统人为因素会议论文集,第 455-462 页。明尼苏达州明尼阿波利斯,4 月 20-25 日。

Robertson, G., Dantzig, M., van Robbins, D., Czerwinski, M., Hinckley, K., Risden, K., Thiel, D. 和 Gorokhovsky, V. (2000)。任务库:一个 3D 窗口管理器。载于 SIGCHI 计算机系统人为因素会议 (CHI 00) 论文集,第 494-501 页。荷兰海牙,4 月 1 日至 6 日。

Rouncefield, M., Viller, S., Hughes, J. 和 Rodden, T. (1995)。“持续中断下的工作:CSCW 与小型办公室”。信息社会 11 (4): 173-188。

Smith, G., Baudisch, P., Robertson, G. G., Czerwinski, M., Meyers, B., Robbins, D., Horvitz, E. 和 Andrews, D. (2003)。Groupbar:任务栏的演变。OZCHI 2003 论文集。澳大利亚布里斯班,11 月 26-28 日。

Sousa, JP 和 Garlan, D. (2002)。Aura:普适计算环境中用户移动性的架构框架。第三届 IEEE/IFIP 软件架构工作会议论文集。加拿大蒙特利尔,8 月 25 日至 30 日。

Suchman, L. (1987)。计划与情境行动:人机沟通问题。剑桥:剑桥大学出版社。

Wertsch, J. (1985)。《维果茨基与心智的社会形成》。马萨诸塞州剑桥:哈佛大学出版社。

四

关于桌面隐喻的思考和 一体化

第四部分简介

第四部分,即本书的最后一部分,重点探讨了与桌面隐喻和集成数字工作环境设计相关的两个普遍性问题。其中一个问题,即用户如何理解和解读桌面隐喻,由 Ravasio 和 Tscherter 在本章中讨论。

该隐喻面向参与计算机系统分析、设计和使用的不同类别的人员,包括研究人员、设计人员和用户。在所有这些类别中,用户可以说是该隐喻最重要的目标受众,但令人惊讶的是,人们对他们(而非其他类别)如何理解和使用这个隐喻知之甚少。第四部分讨论的另一个问题是工作环境集成的通用方法。集成是否可以通过扩展某个特定应用程序,直到它允许用户执行所有类型的任务来实现?Kaptelinin 和 Boardman 在本章中讨论了这个问题。

Ravasio 和 Tscherter 在本章中的讨论基于作者开展的实证研究。这些研究旨在了解用户在基于桌面隐喻的环境中日常工作中形成的“理论”。通过观察和访谈,研究人员发现,在现代计算机技术的日常使用及其与物理办公环境的整合中,出现了一些概念性问题。这些问题的根源可以追溯到人们难以理解桌面系统工作原理的一些基本概念。作者提供了大量示例,说明桌面隐喻如何失效,并阻碍人们使用计算机技术的一些可用功能。特别是,研究发现,经验不足的用户往往没有意识到

桌面是文件系统的一部分,可用于存储文件和文件夹。

本章阐述了将基于设计的研究与现有技术使用情况的实证研究相结合的重要性。对使用这些技术的人的实际问题和工作实践进行实证分析,可以揭示针对新技术和高级用户进行的研究的盲点问题。例如,正如本章所述,“普通用户”(不一定是技术专家)对关键系统功能的看法可能与研究人员或设计师截然不同。因此,根据目标最终用户的关注点来检验新颖的设计理念至关重要。

在第四部分的第二章中,Kaptelinin 和 Boardman 区分了两种集成工作环境的方法:(a) 以应用程序为中心的集成,即创建一个允许用户处理不同类型信息对象的“大型应用程序”; (b) 工作空间级集成,支持在统一工作空间内协调使用多个应用程序。以应用程序为中心的集成的一个例子是为电子邮件添加功能,使其转变为通用的任务管理环境。本章讨论了这种电子邮件开发方式的优缺点,并提出了支持应用程序中心集成的替代方案的论据,即在工作空间级支持多种工具的集成。作者开发的两个系统 UMEA 和 WorkspaceMirror 旨在阐明工作空间级集成的概念。

9

用户对桌面隐喻的理论,或者为什么我们应该寻求无隐喻接口

帕梅拉·拉瓦西奥和文森特·切特

桌面电脑的兴衰

桌面隐喻 一种促进访问集成数字工作环境的卓越隐喻 发明三十年后,

仍然是当前主流系统的标准入口。至少在商业上,替代方案几乎不存在 或许除了用于PDA的PalmOS。桌面隐喻最初旨在简化办公室工作人员执行的结构不良但常见的任务和操作 (Johnson等人,1989)。然而,多年来,个人电脑被引入到与传统办公室工作无关的领域,需要支持的例程甚至更加不明显。为了跟上这些发展的步伐,桌面也进行了相应的调整。

因此,如今商业数字工作环境用户遇到的许多问题,都源于一种过于具体的隐喻,这种隐喻不再符合 (甚至可能从未符合)其起源的现实世界规则。尽管 (或许正是因为)桌面版本更新,如今的个人电脑仍然无法达到 Dynabook (Kay 和 Goldberg 1977)所定义的灵活性、强大功能和无缝集成的工作程序,而这些功能据称已经实现。¹

本章的动机和目的

虽然我们已经拥有扎实的知识基础,能够满足用户对桌面系统的需求,但仍存在一些重要的问题有待解决。例如,悬而未决的问题包括:(1) 对屏幕空间的使用知之甚少;(2) 迄今为止,工作环境一直被视为由多个独立组件组成的集合,而非一个完整的整体。

其目的是支持工作活动；(3)因此，在新系统的设计和开发中，只解决了部分问题，而且这些问题几乎完全被相互排斥地进行。因此，本章的目标如下：

首先，提供关于在工作实践中如何使用实际桌面（即屏幕空间）的知识。由于屏幕空间是当前系统的入口，因此它是系统中最显眼的代表性设施，并且显然是用户拥有的区域。

然而，迄今为止，人们对这一领域的了解甚少，因此，从我们的角度来看，有必要填补这一空白。

第二，提供关于用户在物理环境中体验的见解
工作环境“办公室”和电子工作环境“办公桌”

“顶层系统”相互关联、相互依赖。

这两种环境常常被视为同一

它们在用户处理和组织信息的过程中相互补充和支持，共同构成一个整体。

然而，与典型的硬币不同，两种环境的权重会有所不同。通常情况下，实体办公室仍然是主要环境，因为它的依赖程度更高，因此被视为另一种环境的参考基础。毕竟，台式电脑的很大一部分目标就是成为实体办公室的成熟电子版本。虽然这种定位在“台式”电脑诞生之初可能很有用，但如今的电脑已经不仅仅是复制实体办公室的目标，还能为信息处理和管理信息所需的工具开辟新的途径。因此，了解这两种环境如何相互比较和影响似乎很有意义。例如，随着可靠索引机制的出现，一些用户开始将信息存储在“池”中，而不是按层次结构存储。此外，相当一部分用户在收集和存储信息时遵循“保留所有内容 你永远不知道”的策略。这些方法在过去的实体办公环境中是不可想象的。

这两种方法必然会引出一个问题：为什么迄今为止的研究成果对商业系统几乎没有影响。

虽然这个问题没有明确的答案，但我们将尝试分析一些可能成为答案的方面。因此，本章将围绕以下核心问题展开：

为什么我们应该寻求无隐喻的界面 267

•从用户使用现有商业系统的经验中,我们能学到什么?这些知识将如何影响未来新型系统的开发? (“实践、问题和桌面系统”部分)

•鉴于从用户的角度来看,集成的数字工作环境是物理办公室的电子对应物,这种关系是否会导致物理和电子“办公室”之间产生不必要的相互影响?(题为“物理和电子办公室的共存”一节)

•关于集成数字工作环境未来发展的设计问题,一般来说,可以说什么?(“设计方法”部分)

我们通过报告我们的两项不同研究(Ravasio、Guttormsen-Schär 和 Krueger 2004; Ravasio 2004)的结果,并讨论和总结对未来桌面系统正在进行的工作的后续影响来解决这些问题。

背景:桌面系统使用简史

为了确定电子办公信息系统的相关需求,Malone (1983)对基于纸张的实体办公室进行了研究。他指出,除了常见的文件和文件夹外,还有一种快速且非正式地组织信息的方式“堆”。Malone还指出了四个在随后几十年的系统中反复出现的问题,例如本书中所讨论的那些:

用户更倾向于空间分类而非逻辑分类。Vicente、Hayes 和 Williges (1987)证明,空间能力较弱的人在层次化文件系统中容易出现定向问题。Kidd (1994)探讨了空间布局(以及由此产生的分类)对于知识型工作(即知识的开发和获取)的价值。

信息获取通常依赖于多种属性。Kwasnik (1991)指出,(物理)文档分类不仅取决于文档属性(例如作者、标题等),更重要的是,取决于情境因素(上下文)。根据她的研究成果,她还编制了一份基于上下文的类别列表,她的研究对象根据这些类别对个人文档进行分类。Barreau (1995)重复了这项研究,研究对象为

个人电脑（而非实体办公室）的订购策略相似。结果发现，电子办公室和实体办公室的订购策略相似。关于本地系统搜索的成果数量相当有限。迄今为止，只有 Barreau 和 Nardi (Barreau 和 Nardi 1995;Nardi 和 Barreau 1997) 触及了该主题，其余研究则集中在传统的信息检索类型上（例如，Bates 1979;Sutcliffe、Ennis 和 Watkinson 2000）。

信息很少能被明确地归入单一类别

Kaptelinin (1996) 更详细地研究了用户在文件系统中应用的组织策略，尤其是用于项目工作的策略。他指出了以下问题：(1) 缺乏用户支持来追踪和规划个人活动；(2) 缺乏支持临时文件配置的功能；(3) 系统文件信息无法重现其上下文。当个人电脑最终普及，未经培训的“普通”用户也能使用互联网时，Barreau (1995) 以及 Nardi、Anderson 和 Erickson (1995) 在一项关于计算机工作实践的研究中采访了共 22 位用户。他们得出结论 (Barreau 和 Nardi 1995;Nardi 和 Barreau 1997)，信息分为三种类型：临时信息、工作信息和归档信息。不仅在实体办公室，而且在电子办公室中，文件放置都具有重要的提醒功能。

信息收集总体上没有得到很好的维护；人们更倾向于使用手动搜索程序而不是勉强使用的内置搜索工具。

他们还能够证明，分层文件系统及其命名机制可用于铭刻提醒，以供日后“定位”。

分类是一项艰巨的任务信息获取和分类行为在某种程度上分别是个人信息空间中许多活动的开始和结束 (Landsdale, 1988)。然而，迄今为止，对它们的研究还很有限。在同一背景下，Abrahams 等人 (Abrahams 和 Baeker, 1997;Abrahams、Baeker 和 Chignell, 1998) 分析了书签的使用情况。他们的结果与 Malone 的结果相似，因为他们注意到了标记或管理语义组织时所涉及的问题，并且书签还可以作为记忆浏览会话序列的助记符。此外，他们还指出，大多数用户认为网络上可用的信息分为“我的书签”和“未映射源云”。

通过对电子邮件的深入研究,Pliskin (1989)、Whittaker 和 Sidner (1996) 以及 Bälter (1997) 成功定义了一系列不同的电子邮件用户类别:优先排序者、归档者、不归档者、春季清理者、频繁过滤者和不清理文件夹者。近期,Ducheneaut 和 Bellotti (2001) 指出,电子邮件是一种所谓的“栖息地”,即用于完成和组织各种专业和私人活动的设施。

在我们自己的调查中,我们研究了桌面本身(即屏幕空间)的使用情况,并试图识别用户在日常使用计算机过程中遇到的各种问题(Ravasio、Guttormsen-Schär 和 Krueger,2004)。我们将获得的洞察融入到界面原型的开发中(Ravasio 等人,2003),并开展了一项研究,分析了在文档分类和检索过程中,实体办公和电子办公之间的相互影响(Ravasio,2004)。以下章节将呈现从这些研究中得出的一些见解、经验和反思。

综合以上信息,我们可以观察到,计算机确实在各个方面改变了办公室的工作流程。然而,办公室仍然以“纸质”为主(Whittaker 和 Hirschberg,2001)。

实践、问题和桌面系统

桌面系统支持各种活动,从信息获取到信息分类,再到新信息的使用和汇编,以及最终的信息分类。虽然实际工作任务的支持由各个应用程序负责,但有些基本活动需要由环境本身(而非第三方软件)支持。这些活动包括组织和检索本地存储的信息、注释和评论、审阅以及版本控制。我们在此背景下研究了以下两个相关问题(Ravasio、Guttormsen-Schär 和 Krueger,2004):(1) 所谓的电子桌面实际上是如何被感知的?它的用途是什么?(2)为什么用户认为对自身信息进行分类和检索很困难?

为了回答这些问题,我们按照指导方针进行了16次半结构化访谈。所有访谈均在受访者熟悉的工作环境中进行。访谈问题在受访者坐在电脑前时提出,并配有

摄像机位于受访者右侧,以便同时捕捉受访者和采访者的声音。受访者由12名Windows电脑用户和4名Macintosh电脑用户组成:5名研究人员(来自职业健康、视觉感知、电子工程和增强现实领域)、2名兼任讲课的研究经理、2名业务经理、2名秘书、2名学生、1名全职讲师、1名教师和1名程序员;受访者由7名男性和9名女性组成。后续章节将讨论我们对上述两个问题的答案。

屏幕空间的用途

按照最初的设想,实际的桌面代表着一个用户专属的区域。但从技术上讲,它只是文件系统中的另一个文件夹,具有一些特殊属性,包括通过空间布局在屏幕空间中的显示。使用桌面这样一个强有力的隐喻,不仅提供了根据用户需求和喜好进行调整的机会,更重要的是,它允许中低技能用户快速掌握自己的系统。然而,这种初衷与现状并不相符,因为我们观察到,屏幕空间的使用很大程度上取决于用户的技能。

低技能用户并不知道屏幕平面可以用作数据存储位置,并且最终会成为文件系统的一部分。另一方面,中高级技能用户则有意识地、广泛地使用桌面,并根据自己的工作需求进行调整。因此,技能仍然是高效掌握此界面的重要标准。

非专业用户也经常因为文件夹层级结构和桌面功能上的相似性而感到恼火,他们惊讶地发现,一些简单的操作竟然能带来与预期不同的效果。例如,一位中等水平的用户将整个“我的文档”文件夹层级结构(Windows 资源管理器左侧)拖到桌面,以为这样就能概览树状结构的层次结构。然而,当他将层级树拖到桌面时,他最终注意到,这个操作导致整个层级结构(包括其内容)都移动到了桌面。造成这种恼火的一个原因是系统滥用屏幕空间来达到自己的目的。

无论是存储新创建的快捷方式

在没有用户明确要求的情况下,系统会自动安装程序或显示系统活动提醒(例如,托盘)。通常情况下,用户不敢主动干预,也不敢丢弃不需要的资源,因为他们认为原本需要的资源(例如,快捷方式指向的应用程序)会“神奇地”从电脑中消失。

中高级用户首先将屏幕空间用作临时存储位置。为了实现此目标,屏幕空间由每个用户按图案排列,旨在支持其快速视觉定位(图9.1和9.2)。这些图案本身采用简单的几何形状,例如正方形、圆形等,并按文档进行组织。



图9.1

在这位资深PC用户的屏幕上,各种几何形状清晰可见,文件组也按文件格式和用途进行了分类。例如,左下角显示一个程序快捷方式的正方形,右下角则显示批处理文件快捷方式的集合。Windows任务栏位于左侧,这样可以将(右手)鼠标“扔”到任务栏上,从而减少完成任务所需的体力。



圖9.2

这位 Mac 资深用户通过添加文件夹快捷方式自定义了“Dock”。他还采用了文档和文件夹的空间布局，以便高效访问常用资源。

类型、主题等等。接近度代表了主题或类型的关系。由于桌面充当临时存储空间，随着时间的推移，它也变得拥挤，需要偶尔清理或重新整理。重新整理桌面后，其内容将按照以下三个标准进行整理：(1) 临时信息保留在上面；(2) 长期有用的信息被归档到“存档”文件夹中；(3) 工作信息要么归档到包含正在进行的工作的文件夹中，要么在桌面上的特殊位置重新整理。同样，这些标准符合 Barreau 和 Nardi (1995) 已经发现的三种通用信息类型。

即使在文件系统的层级结构文件夹中,有时也会依赖空间布局。特别是,当存储的信息总量以及每个文件夹中存储的信息量从个人角度来看不太大时,就会依赖空间布局。例如,有些用户知道某个文件夹

在打开的资源管理器窗口中,它将是从顶部数第二个,或从底部数第二行左侧数第三个。

总体而言,我们同意 Halasz 和 Moran (1982) 提出的批评,即具体的隐喻 在本例中是桌面 并不能帮助新手理解计算机。尽管如此,我们认为这种批评适用于所有计算机系统专家用户,而非专家用户。隐喻所引发的心理模型 桌面 很难与现实世界中对应的实体办公室相提并论。普通用户无法判断技术和其他需求如何影响隐喻与其对应物的对应性。因此,对于新手来说,除了计算机本身的功能之外,学习这种兼具一致性和不一致性特征的混合体中的惯例至少同样困难。

更高级的用户虽然记住了这些差异,但这并不能帮助他们真正“理解”他们正在使用的系统。对他们来说,学习与不断记住更多约定是并行的。

分类和检索

虽然实体办公室允许多种方式完成信息分类和检索任务,但层级文件夹结构以及文件和文件夹的命名方式将这种多样性降至最低。这种缺乏灵活性的现状使得用户很难留下已习得和隐性知识的痕迹,而这些痕迹将有助于日后追踪特定的信息。因此,用户必须投入大量精力来定义、组织和维护这些层级结构,并尽可能多地传递知识。因此,他们将精力投入到一个他们认为知识不会丢失且易于检索的地方:文件夹层级结构及其命名模式。

然而,组织方面的难题早在实体办公空间或计算机出现之前就已存在。这些问题在教育环境中尤为突出;例如,对于向新手介绍计算机使用的教师来说,最难克服的障碍之一 (Reichmuth 2004)是,对一些人来说,办公室和计算机都无法帮助他们组织信息或文档。这些用户对信息或文档的理解截然不同,以至于这两种环境中所需的结构化组织对他们来说毫无意义。

组织工作与重要资源的归档工作相辅相成。其目标是确保即使在相关网络服务中断的情况下,也能“持续”访问相关资源(网页、文章、报告,以及图片、音频和视频资料等)。

因此,用户尝试将信息存储在本地,以便用户自己访问。同样,他们的档案对他们来说也体现了图书馆的角色:它被视为成果和产品的集合,可以直接重复使用而无需修改,或者作为参考资料。

然而,档案只有在包含纯粹的必需信息,且不散布过时、废弃或无关材料的情况下才有用。因此,维护已成为中高级用户定期进行的一项重要活动。维护确保只保留有价值的提醒和尚未失去相关性的信息,而其余信息一旦过时就会被整理出来。通常,维护是在项目“里程碑”处进行的,此时有用且重要的资源(例如,实际结果和相关文档)被保留,而其余资源则被丢弃。存档文件(即非工作文件)的年限大约为六个月到八年,档案中较旧部分的数量逐渐减少。后者支持了以下发现:不仅要进行维护,而且多年来档案中较旧的部分也会被反复清理,直到它们被发现只包含“纯粹的必需信息”。有时,我们会看到档案中较旧或非常庞大的部分被外包给DVD等外部存储介质,然后“存储”在物理货架空间上。

分类的目的是将知识铭刻在文件系统中,以便创建和留下线索,使信息能够再次被找到;而与之对应的检索则涉及制定策略,试图解读这些线索,以便重新获取先前存储的信息。需要注意的是,在本地系统内进行检索始终是尝试查找已知存在的信息,因为这些信息在过去的某个时间点被处理过;这与网络搜索工具或数据库检索工具不同,后者旨在根据某些特定条件检索最接近的、先前未知的匹配信息。

这种对比或许可以合理地解释为什么内置搜索工具很少被使用 用户寻求的不是“最接近的匹配”,而是100%的匹配。

此外,使用搜索工具似乎需要一定的认知能力,因为搜索条件(“字段”)并非人们在工作中思考、行动和记忆信息时惯常使用的条件。因此,搜索工具的结果往往不尽如人意。基于这些经验,用户最终倾向于手动搜索。同时,对他们来说,手动搜索也是一种温习个人雕刻组织知识的方式,而且无需比使用工具付出更多(认知)努力。

以下是此背景下需要回答的最后一个问题:从不同用户的角度来看,信息访问流程是什么样的?在我们的研究中,所有受访者都首先选择直接访问其类别进行搜索。接下来,受访者按逻辑顺序逐一查看所有文件夹,优先考虑可能包含所需文件的文件夹,但不会查看其中包含的每个文件。如果此时尚未找到所需的特定信息,受访者最终将手动逐个检查相关文件夹中的所有文件。如果搜索到此为止仍未成功,则从第一步重新开始。

问题信息的分类和检索需要巨大的认知努力。虽然这个问题的直接解决方案还看不到(尽管有一些有希望的进展;例如参见 Copernic Technologies 2004),但显然需要做出努力来减轻这种负担。加剧认知负荷的一个原因是系统对各种信息类别(如书签、电子邮件和文件)的分离。用户将他们的数据视为一个单一的信息体。现有的分离增加了归档信息和再次查找它们的难度。有必要使用一种简单的存储机制将所有用户拥有的数据重新统一到环境中的单个存储位置,该机制允许并支持用户根据自己的思维过程创建和雕刻信息之间的关系。然而,这种链接机制不能简单地由“超链接”信息组成(这意味着将所有事物与其他事物潜在地联系起来这种方法会对人的定向能力产生毁灭性的影响[Conklin 1987]),而是旨在从用户的角度将属于一起的项目连接起来。

对个人信息集合和档案的自动压缩或汇总支持更是少之又少,更不用说共享信息资源或协同工作了。然而,这也带来了一个核心优势:维护活动(通常是重新定位和丢弃信息片段)极其容易执行,几乎没有任何内在问题。如果信息通过共享或协作与系统内(甚至更糟的是,在远程位置)的其他信息相关联,“删除”将不再是一件小事。如果存在不同版本的

在定义关系时考虑了文档

各个信息之间的关联。

虽然目前的商用系统主要面向内容(即关注特定文档中包含的实际信息),但使用PC的工作也可以面向任务(关注待完成的任务)或面向情境(同时关注各种文档、程序和手头的任务)。由于用户不断在工作模式之间切换,因此必须为每种模式提供同等的支持。然而,无论PC系统上的活动是从任务、情境还是内容导向的角度来看待的,它们都可以被分配到三个交替的阶段之一,即:(1)从本地或远程来源获取信息;(2)实际工作,包括处理和转换先前获取的信息;以及(3)对原始资源和工作成果进行重新分类(Landsdale 1988)。虽然信息分类行为由环境本身支持,但对信息获取以及正在进行的(“转换”)工作的支持主要外包给应用程序。因此,系统端和应用程序端需要以几乎完全不同的方式看待。

由于桌面侧重于文档管理,因此几乎不支持正在进行的工作活动。例如,该环境几乎不提供任何评论、批注、版本控制或用户友好的全局搜索功能。内置搜索工具通常基于元数据,至今仍未受到普通用户的重视。由于技术上无法绕过检索工具中元数据的使用,我们需要了解“用户友好型”元数据的外观和构成,才能真正朝着……的方向发展。

为什么我们应该寻求无隐喻的界面 277

系统端支持自动检索以及链接或分类。

用户在其系统内完成的工作通常与即时归档或访问信息的情境（上下文）直接相关。正因如此，无论系统端对内容或任务导向型工作的支持策略设计得多么完善，一旦情境问题介入，这些策略都会面临严峻挑战。

情境是难以预测或构建的，即便可以实现也相当困难。因此，我们的目标必须是设计便捷、易用、“即时”的程序，使实际用户能够并支持他们“揭示”自己对情境的感知。

实践与问题·结论

桌面最初的设计是用户专属的区域。它应该是一个只有用户才能拥有个人喜好和品味的地方，系统绝不应该滥用这块区域——包括安装程序在其上存储快捷方式！图 9.1 清晰地展现了中高级用户如何拥有屏幕空间的含义。图 9.1 中描绘的这位用户不仅根据最适合手头工作的标准订购了图标和文档，还用一张个人照片来装饰桌面背景。

当前的商用桌面系统仍然主要为文档管理而设计。它的目标并非支持思考行为，即新信息的汇编或生成。因此，核心系统本身对这一行为的支持还很初级，通常委托给应用程序。每个用户都可以从自己使用的单个文档集合和一系列第三方应用程序中获取所需信息。虽然从面向任务的角度来说，这可能合情合理，但

从这个角度来看，它几乎无助于解决一个事实：如果没有独立于特定应用程序的、合适的、全面的支持，单个信息最终将沦为孤立的知识孤岛。具体而言，这意味着真正以上下文为导向的用户信息视图是一项“不可能完成的任务”。

重要的是要记住，用户知道一组信息的价值大于每条信息价值的总和。然而，他们也意识到额外的

以隐性关系和隐性知识形式存在的好处仅存在于“他们的大脑中”，也就是说，它既不包含在单独的信息中，也不包含在整个集合中，也不包含在系统中。

经验告诉他们，在某些时候，该系统对于获取信息以产生结果并没有太大的帮助。

最终目标是让信息处理“更贴近”用户。

例如，不应将用户信息分散到整个系统的不同部分，而应将它们整合起来，无论是存储位置还是整体价值的呈现方式。然而，系统操作的资源和用户操作的资源必须相互分离，以确保前者不会干扰后者（对于普通用户而言，反之亦然）。

最后，任何有助于将隐性和习得性知识铭刻到用户信息组织永久结构中的支持，在需要时可被应用程序和服务重用，都是有益且值得赞赏的。小型实用的功能可能产生巨大的影响：注释、版本控制、格式转换、易于使用的搜索等等，都是我们之前提到的例子。此外，还可以设想出更多功能，尤其是如果将协作问题也考虑进去的话，而我们在本章中完全忽略了这一点。然而，从现有功能中我们可以学到的是，只要它们的使用需要与手动完成相同任务一样多的认知或体力劳动，这些功能就不会被使用。

实体办公室与电子办公室的共存： 定性实验

随着计算机的使用日益普及，打印件、书籍、报告等耗费的纸张比以往任何时候都多（Sellen and Harper 2002）。这引出了一个问题：实体办公室是否仍然是指导人们如何使用其虚拟对应物——台式电脑的主要参考？以及实体办公室和虚拟办公室是否会相互影响？

在一项定性实验（Ravasio 2004）中，我们调查了“办公室”工作的基本程序，并过滤了个人的表现。

个人背景。两组十名参与者，每组分别被安排在一个陌生的办公室（即真实的他人办公室）或一个陌生的计算机组织（即真实使用的计算机的镜像副本）。参与者的背景如下：4名计算机科学家，1名PC支持者，7名学生（2名商科学生，1名语言学学生，1名心理学学生，3名环境科学学生），1名电气工程师，1名理疗师，1名化学家，1名经理，1名心理学家，1名律师，2名秘书。

实验之前，研究人员从每个环境中提取了两组文档。实验过程中，参与者首先被要求搜索第一组文档，然后在他们被分配到的环境中对第二组文档进行分类。

我们的目标是提取普通用户在信息导向和处理过程中所遵循的原则（文档描述符、策略和规则），并探究这些原则是否在不同办公环境中存在依赖或影响。每次用户会话都会被录像，录音会被转录并进行评估，之后还会由两名工作人员根据之前生成的代码本进行人工复核。

下一节将根据我们在本定性实验研究中获得的结果，更深入地探讨两种办公环境之间的相互影响问题。

办公室组织的艺术

在他人拥有的实体办公室中，用户几乎能够“一眼”就辨别方向，并快速找到所需文件。我们的参与者在实验过程中对此现象进行了独立评论。事实上，在陌生的（实体）办公室环境中，我们观察到参与者似乎在短短几分钟内就了解了办公室的布局。对于不同类型的信息和文件的典型存储位置，似乎存在着某种约定俗成的惯例，或者说“默契”。

同样的“静默理解”似乎在电子世界中并不存在。无论是在文件系统的组织上，还是在屏幕平面上。在这里，人们会持续不断地努力推断并得出关于信息逻辑位置的结论。

普遍可理解的线索或惯例要么完全不存在

或呈现方式不尽如人意,导致用户的需求、期望和目标无法得到满足。有时,我们会从实体办公室中汲取一些线索和提示,例如使用数字编码或颜色编码文件夹。然而,这些方法未能达到其预期目的,即刻画或“外化”个人的组织结构,使其易于他人以及“未来的自己”理解。

因此,在某种程度上,用户试图使用相似的、有时甚至是相同的惯例、结构和程序来组织他们的工作环境(即实体办公室和电子文件系统)。在我们的实验中,这一点在比较参与者在分类或检索文档时在每个环境中应用的策略和规则,并与用户自己对其流程的评论进行交叉核对时变得显而易见。然而,用户的意图往往没有按预期发挥作用,这给他们带来了明显的困惑,因为他们无法确定失败的原因;因此,他们既不清楚如何避免这种困惑,也不清楚究竟是什么导致了这种困惑。

或许还有其他方法。正是在这里,两种环境之间一系列根本的概念差异体现在参与者的行动和组织方式上。我们通过比较两种环境中的行为和陈述记录来证明这一事实。

我们的实验比较表明,许多组织概念和理念已从一种环境相互吸收到另一种环境中。我们得出结论,由于现实办公环境中的物理特性,这些“采用”的局限性和弊端,以及其带来的好处,对计算机新手和专家都显而易见。然而,在计算机的电子环境中,只有专家才能判断其影响。缺乏概念性知识导致新手和中等水平的用户迅速接受错误的假设,这反过来又导致他们在使用计算机的组织习惯和程序上出现根本性问题。

实体办公室的组织最终是介于搜索和归档活动与特定个人需求之间的一种实用性中间路线。主流计算机系统隐含地假设其用户存储文档和信息时所采用的概念和理念(标准)与检索时所采用的相同。然而,这种假设是不正确的。我们发现,