计算机科学1995 Vol. 22 No. 4

自适应人机界面的评价*)

李成辉 谭 浩 刘锦德
(电子科技大学微机所 成都610054)

摘 要 Briefly discribed the approaches and methods used for evaluating human-computer interfaces, and discussed several special problems in evaluating adaptive human-computer interfaces.

关键词 Adaptivity, Human-computer interface, Evaluation.

1. 引 言

评价是人机系统设计中的一个重要组成部分。所谓评价是把软件系统按其性能、功能、可使用性等方面与某种预定的标准进行比较，验证系统设计是否满足用户需求，检查所开发的系统在功能和性能等方面是否满足设计者的意图，诊断系统的潜在缺陷等。

通常，对界面进行评价是在界面已经开发完成后才进行，但如果在系统开发的早期阶段就进行评价，就可以及早发现一些设计缺陷；如果在系统设计的后期或者在系统已开发完成以后才发现这些缺陷，校正工作就要复杂得多，造成大量时间、人力和物力的浪费。

评价分为"构成评价"和"总结评价"。构成评价是指在系统的开发过程中进行评价，目的在于找出系统中有待修改或增强之处。总结评价是在系统开发完成后，对其功能、可用性、有效性等总体性能进行总结性的评价。

对人机界面的评价已由直觉的、凭经验的方法，转为逐渐采用科学的、系统的方法。最新的研究包括为一些评价准则提供理论支撑，以及开发可对用户界面进行评价的概念框架等。

本文将首先讨论人机界面评价方面的一般方法，这些方法同样适用于自适应人机界面的评价；随后，再讨论自适应人机界面评价方面特有的问题。

*)本文工作得到电子部电子科学研究院基金资助。

2. 评价的目标

对人机界面的评价包括三个一般目标，而与界面类型、硬件/软件、设计阶段、所使用的方法以及所收集到的数据的类型无关。这三个目标为：

1)设计功能的评价。将某个设计所具有的功能与用户需求进行对比评价。这有两种实现途径，即面向用户的方法或面向系统的方法。前者是根据用户使用系统和界面所能完成的任务以及这些任务满足用户任务需求的程度来对设计能力进行评价；后者则是对系统和界面的特色评价并将它们与用户的特色需求进行对比。

2)设计决择影响的评价。找出某个界面设计对用户以及用户与系统的交互作用会产生什么样的结果，对有意的和意外的影响都进行评价是重要的。所谓有意的影响是那些界面设计所期望出现的结果；而意外的影响则是伴随某个设计特色而产生的结果。在对界面设计进行评价时，最引人注目的影响是为了使用系统所具有的能力，用户需要付出多大的努力。

3)设计的潜在问题的诊断。诊断评价目的在于找到并弄清界面设计存在的问题及其影响范围。通常是通过对人机交互作用过程进行检查，从而找出潜在的缺陷。

3. 评价途径

3.1 设计—评价循环

一个普遍接受的观点是，评价应与界面的设计同步进行，以保证所开发的界面不会远离设计目标和用户需求，并且可以随时对所作的设计进行必要

的修改。

传统的设计过程，首先是根据设计目标和用户需求编制出设计规范，然后据此进行设计并加以实现；在实现完成后，将对目标系统进行评价，并将结果返回到原始规范，接着对设计进行修改并重新加以实现，重新进行评价；如此反复直至获得满意的目标系统为止。

对于小项目而言，这种设计—评价循环的方法是可行的，但其缺点十分明显：若评价结果建议对设计应进行修改时，需对源代码进行修改，对于较大的或复杂的项目，到最后阶段才进行评价并不实用，往往只能对系统进行一些小修改。为了解决这一问题，需要采用其它方法。

3.2 模拟式评价

模拟是一种重要的评价途径，可以在界面实现以前就对它进行评价，从而可以快速地对界面设计进行修改，省时省力。模拟有多种形式，例如，可从只是简单地描画出所设计的屏幕布局，到动态地、交互地进行显示。模拟可能需要很大的工作量，而所作的努力很可能与最后的实现工作无关。所以，一般只进行简单的模拟；实际上，通过很简单的模拟，就可以进行许多有用的评价。

然而，模拟的复杂程度通常确定了所作的评价的实用程度，因为模拟越精确，就越能确信评价的结果可适用于真正的界面，许多简单的模拟尽管很有用，但却是静态的，不能真实地模拟人机交互作用的动态过程。采用静态模拟，系统不能向用户提供任何反馈，因而就很难模拟大的动作单位，例如命令序列、整个用户任务以及错误校正过程等。

要解决上述问题，更实用的方法是采用动态模拟，在终端上向用户呈现一个模拟的界面。它涉及实际界面的某些方面；用户可向这一模拟品发送命令并接收响应。"系统"好象真在执行用户发布的命令，但这实际上是评价者事先编程定死了的。

采用这种方法的指导原则是，进行模拟所需要投入的努力应远小于实际实现系统所需的努力，因为为模拟而编制的代码最终都会被丢弃，它们本身不会形成目标系统。

3.3 原型式评价

利用形式化的对话规范说明方法，以及将规范转化为具体的实现结果的软件工具，可以快速地建立用户界面的原型。这种方法与动态模拟很相似，但利用特别开发的软件工具，可以提供更复杂的、与实际设计更接近的界面。所以，它已受到评价者的重视。

3.4 模型式评价

在设计的早期阶段，许多研究者都尽力想采用一种形式化的方式来描述人机交互作用的不同方面，现已开发出了多种模型，并尝试对它们进行了分类，对这些模型进行分类是沿着两个轴进行的：(1)谁或什么被造型；(2)谁或什么负责进行造型。有的分类法还包括第三轴，即模型是用于谁的。

建立界面的形式模型的过程分为下述几步。首先，从系统预期应完成的任务中选出有代表性的一种，这可以通过任务分析方法来完成。然后，找出用户为了完成这一任务而需要执行的那些动作("方法")。这些方法通常被组织为由各个层次构成的层次组织形式，最顶层是任务层、向下还包括对话层等层次，直到输入/输出层。表1是一种模型的层次结构。

表1 Moran的层次结构模型

组 件	层 次
概念部件	任务层
	语义层
通讯部件	语法层
	对话层
物理部件	空间布局层
	输入/输出层

概念部件含有系统赖以组织的抽象概念；通讯部件含有命令语言和通讯对话；物理部件包括用户见到和使用的物理设备。这些部件明确地分成不同层次，每个层次都根据自己的抽象级别对系统进行了完整描述，这些描述中含有利用在相应层次可使用的动作(方法)来实现系统所涉及的任务的例程。

最上面是任务层，负责分析用户希望执行的任务集。语义层对概念实体以及用以实现用户任务的概念操作进行安排，这一层次代表了系统在功能方面的能力。语法层对命令语言(命令、参数等)进行安排。对话层对对话进行了定义，即对与语法层的每个元素相关联的动作以及控制对话的规则作了定义；规定了用户的按键序列以及操作其它设备的动作序

列，还有系统的显示动作。输入/输出层描述了 I/O 设备和显示结果的布局。设备层则负责描述物理设备的特色。

表示交互作用的通常方法是采用一种有代表性的表达语法来对其加以编码，例如可采用任务动作语法(TAG，Payne，1984)；或者采用任务造型语言(TAKD，Johnson，Diaper & Long，1984)。

对界面模型的评价可以基于这种形式化的描述法来进行，也可以采用经验方法。对界面的概念模型进行评价的好处之一，是可以在设计过程的早期进行评价，即在对界面进行更详细的(低层)描述之前进行。采用自上而下或自下而上的设计和评价方法，在把这种形式化的描述模型的各个层次合并到设计中之前，可以对各个层次单独进行评价，也可以相对于其它层次进行评价。

3.5 整个设计过程的评价

采用上述各种手段，就可以在整个设计过程中进行评价。为了有效地进行评价，需要有将设计和评价完全集成起来的手段，而且设计者和评价者要预先考虑好怎样把评价结果反馈到设计中。这可以这样来完成，即在一开始就找出在开发过程中进行设计时所要经历的各个阶段，并指定几个评价点。集成可以这样来完成，即在每个阶段中指定一个或多个设计—评价循环。对于复杂系统而言，在设计的各个阶段均进行评价更加有用和有效。

4. 评价方法

评价方法分为两大类，即形式化的方法和经验方法。

4.1 形式化的方法

形式评价方法的一般是对系统的任务模型进行分析，从中了解系统应该如何完成任务和应该达到何种目标，并由此测定系统的功能、系统的设计效果等。它和经验方法的区别在于，形式方法不需要直接去测试或观察用户的实际操作。对于形式方法而言，最重要的是它假设了任务模型中的组成必须能够充分代表任务的复杂性以及用户操作过程。只有基于比较准确的任务模型基础上的形式评价才是比较可靠的。

形式化的评价方法源于建立用户与界面的交互作用模型。这方面的例子有"命令语言语法"(CLG，Moran，1981)以及 GOMS(Card & Moran，1980)等。CLG 与 GOMS 有许多相似之处，例如，它们都含有一组操作符和方法，以及一组称为任务的、按功能组织的目标；CLG 的符号表示法也与 GOMS 相似。CLG 与 GOMS 是最近的一些模型的先驱(Kieras & Polson，1985；Payne，1984)。

使用形式评价方法的一个好处是，可以在界面的细节描述被具体实现之前就进行评价，从而允许在设计的较早期进行。不过，无论设计者认为该界面如何清晰、简单，仍不能完全预知实际用户所反映的情况。因此，目前仍旧大量依赖于比较简单而可靠的经验方法来对系统作出最终评价。

4.2 经验方法

无论人机界面的形式方法还是评价理论都还很不成熟，而且人机界面的交互过程极为复杂，单纯采用形式方式进行评价，很难考虑用户的认知特性。因此，仍不能抛弃经验方法。实际上，形式方法的开发和改进也需要用到经验数据。目前，较为可靠并且切实可行的仍是采用各类经验方法进行评价。下面就对几种经验方法作一简要介绍。

1)实验方法。有多种，这里只简单给出这些方法的重要特色。实验方法可用于前面描述过的各种评价手段，它们最适于对界面的各种实现法或所作的各种模拟进行对比。实验方法的步骤为：

①确定实验的总目标以及要验证的假设。例如，对于文字处理器而言，某个实验可能是要比较不同输入设备的文字选择性能。在这里所作的假设，可能是假定这些输入设备(例如光标移动键和鼠标)产生的光标的移动速度和精度是不同的。

③设计实验。这是最困难的一步。最可靠的实验方法是随机和重复测度法，即将受试者随机分为两组，一组按实验要求进行操作，另一组用作进行比较的控制组；更复杂的方法是使用多个实验组。所谓重复，是指让每个受试者均经历每种实验条件。实验设计还应确定应测取哪些值。例如，可以是完成某个任务所需的时间，以及完成该任务时所犯的错误等。

③进行实验，然后对结果进行分析总结。受试者的多少将直接影响实验结果的有效性。很多情况下仅使用了很少量的用户来进行研究，由此而得的结论显然不能代表实际的最终用户，评价结果的有效性当然也就大受限制。

2)观察方法。观察用户的行为是一种广为采用的方法。首先给予用户应在某个系统上完成的任务，

然后观察他们执行任务的过程，以对人机交互进行评价。共有三种观察方法：直接观察、录像以及系统监视。

各观察方法最主要的差别在于对用户的干扰程度各不相同，所能提供的数据也略有差异。直接观察方法是指评价者直接对用户在终端前的行为进行观察。这一方法的好处是每当有疑问时可以随时向用户提出；缺点是会打扰用户的操作，且很费时间。录像方法可以提供全面而长久的人机交互作用记录，随后可以进行各种详细分析。但缺点也会干扰用户（除非悄悄进行），而且分析工作很费时间。系统监视法是采用系统本身来对用户会话的某些方面加以记录，可以自动而可靠地记录多种数据，且精度高；当需要长期而大量地收集数据时，此法就更为适用，而且不会打扰用户。这一方法的缺点是不能记录某些类型的数据，所以常与其它方法联合使用。

3）调查方法（Survey）。可为评价研究提供重要数据，在设计的任一阶段均可使用。开始时，它可提供有关用户需求方面的信息，从而可与设计目标进行对比；随后，可以收集到用户关于界面的意见、测试设计的影响并检测用户需求的满足程度。

主要的调查工具有三种，即问卷（Questionnaire）、分级表（Rating Scale）和会晤（Interview）。

与前面描述的用于收集客观数据的实验方法不同，调查工具是用来收集主观数据。它们可能会请求用户描述使用界面的经历，以及他们关于界面的态度和意见等。有的研究者认为主观数据是最重要的信息；例如，他们认为用户对系统响应时间的看法，就远比实际测取的系统响应时间更有意义。采用调查方法，可对用户界面的任何方面进行评价。

调查方法的缺点在于所提的问题必须是用户能明确回答的，该法所得到的是主观数据，其可靠性和有效性均不如前面描述的实验或观察方法。

5. 自适应人机界面的评价：特殊问题

对自适应人机界面的评价亦可沿用上述方法。但是，由于自适应界面本身固有的特点，在如何选择适合的评价方法方面，还存在一些特殊的问题。

5.1 静态对比对象

为了对自适应系统进行对比评价，需要找到一个合适的非自适应系统（或称静态系统）来进行性能对比。如果在几个方面都存在着自适应变化，则对每一方面都应提供静态对比对象。

然而，并非总是能找到合适的静态对比对象，尤其对于新的应用，不存在相应的静态对等版本。对于自适应变化的每一个方面，自适应系统的行为都是在一组可能的状态中变化，评价者可选其中一个状态来作为静态控制对象，最合适的选择应基于实践经历。如果存在一个最佳状态，即在此状态下有范围广泛的用户可获得工作效率的提高，则可将此状态作为静态对比对象。然而，采用最佳状态来作为静态对比对象将是非常复杂的，因为除去简单情形而外，往往存在着大量的可能状态，而且其中的一些还不一定为设计者和评价者所知晓。

另一个问题是，必须确定自适应系统何时是"自适应"的，然后再进行相应的比较。然而，由于交互作用的双方均因相互影响或外部限制而处于流动状态，很难确定这一"自适应"状态。解决方法可以是确定一定的间隔，然后每隔一定的间隔来进行采样，并加以对比。但是，由于自适应系统开始时总需要一定的时间来收集其环境和用户信息，这一期间将是低效的；进行对比时是否包括这段期间，应加以仔细考虑。

5.2 评价的起止位置

评价的开始和终止点也是要考虑的问题。以自适应拼写校正器为例，它根据用户的出错频率来确定校正清单的顺序，使得最可能被用户选中的单词出现在清单的开头，从而减少选择时间。经过一段时间后，完全可以合理地假定用户将学会而不再犯这些错误，用户的出错模式（用户模型）将会改变，所犯错误的总数也将减少，从理论上讲，当用户不再出错时，自适应机制就成为冗余的了。非自适应的校正器也可以减少用户错误，当然其显示出的校正清单并没有特定顺序。当错误趋于零时，自适应的拼写校正器通过按特别顺序显示校正清单而获得的选择时间的减少值也趋于零。所以，评价者应清楚地认识到，在某一点处自适应机制将不再有效。

5.3 自适应行为的动态性

自适应系统除了能通过逐渐适应用户而处于一个优于非自适应系统的状态而外，还应满足更高的要求。评价者的任务不仅是要表明系统能够提高性能，而且还应表明系统能找到环境中存在着的不同最佳状态，即是说系统能够适应具有不同特点的用

户、团体或任务。自适应并不是瞬时的，它是外界环境和先前状态的复合函数。正是由于自适应性的顺序依赖性，导致了自适应行为的复杂性。一个自适应系统通过适应环境而变得令人满意；但由于环境的改变，又可能变得令人不满意。

自适应行为有三种模型，潜在自适应，即系统潜在地适应多个任务，但一旦适应某个任务，就不能再适应另一个任务；兼容自适应，即系统已经适应某一个任务，但还能潜在地适应其它任务；共同自适应，即系统适应某一个任务的同时也适应其它任务。兼容自适应可用于评价系统是否满足上面所说的更高要求。但是，为了保证系统不是简单地拥有一个适合两种情况的静态控制策略，在评价中还应加上条件：人机交互作用是系统逐渐适应环境而变得令人满意的，而不是立即变得令人满意的。兼容自适应和适应的过程性即是自适应行为的动态性，这是自适应系统设计中所追寻的目标。

5.4 自我评价

复杂自适应系统的一个主要特征是系统应能监测和评价自己的行为。在最简单形式下，这仅仅是通过简单的反馈循环来表现的。但随着系统的不断复杂，就要求向系统中引入诸如能对自己的动作过程预先进行评价，甚至对自己的评价功能进行评价等能力。

将评价的专门知识嵌入到系统中，就能使系统有能力评估自己的性能并相应改变自己的行为；这类系统称为自我调节系统。例如，在电话号码检索的自我调节系统中(Trevellyan, Browne, 1987)，系统使用了两个策略：一个是动态策略，它适应于用户使用数据呈 Zipf 分布时的状态；另一个是静态策略，它适应于用户使用数据呈随机分布时的状态。当现行策略的性能达不到基本要求时，系统就使用另一个策略。为了使系统按照这个方法进行自我调整，系统必须具有：一些测试系统相关方面性能的方法、性能的目标档次、一些测试目标性能是否满足的方法以及系统根据结果来改变自己的响应能力。

但是，系统进行自我评价存在两方面的问题。首先，系统仅能有限地访问环境信息。例如，很容易构造这样一个系统，该系统能监测并使用出错率来改变自己的行为，但要做到系统能够评价出错率与用户感受的关系却很困难。其次，向系统提供一个用于进行策略切换的基准值是相当困难的。一种解决方法是在静态系统上做实验，根据给定条件得到标准响应时间或出错率，从而确定基准值。例如，在电话目录检索系统中，在静态系统上使用一百次查号来测得这个基准值—基本平均查找长度。当用户使用数据呈平均分布时，动态策略的平均查找长度将大于或等于这个值，这时，由于静态策略的一致性，使它优于动态策略，因而系统转换到使用静态策略。

进行策略切换并不是一个简单的问题，必须对策略和用户间潜在的相互作用加以考虑。

5.5 稳定性和变化性

自适应系统设计的一个关键问题在于如何设计系统的自适应控制策略，一个难点是如何解决控制策略稳定性和变化性间的平衡。在用户看来，一个在策略间进行迅速切换的系统是不连惯的，而系统的不连惯性将会损害系统的性能。解决好稳定性和变化性间的平衡是增强系统性能的决定因素之一。

6. 评价与设计

前面讨论了自适应人机界面评价的问题，现在从总体上来看看评价技术怎样融入设计过程。

6.1 构成评价

自适应系统的设计也应采用反复设计、建造和评价的设计方法。整个设计中，对系统运行的假设基础的有效性进行检查将有助于了解系统是否达到目标。

用户需求分析可认为是对系统的潜在用户的需求进行评价。对用户需求和可获得功能的检验能说明系统需要自适应的方面。

自适应性必须基于用户或任务的可变性的某些方面，而对这些特性的特点必须作出初步评估。在某些情况下，这些变化特征可在文献中查到，而在另一些情况下，则必须对此进行实验研究。

接着，设计者必须找到能满足这些目标的自适应策略，然后说明自适应所依赖的触发信号和根据这些信号所作出的界面变体推荐。在此阶段，设计者应为自适应机制建立文档。此时，还能预测到在具体实施中存在的潜在问题。

6.2 总结评价

一旦系统构造好后，就需要对系统进行总结性评价。总结评价的主要目标是对于提供了自适应特

性的那些任务，要确定受试者在自适应系统下的效率是否高于类似的非自适应系统。在下结论之前，应首先回答下述问题：

1)系统是否按预定方案运行?给定任何用户输入后，设计者应能预测系统的响应。

2)自适应系统运行的假设基础是否有效?尽管在开发初期验证假设基础是正确的，但还是有必要在系统构造好后对所作的假设的合理性进行重新验证。

3)系统是否是自适应的?需要收集来自用户的数据，以确定系统是否根据用户的变化而改变响应。

4)系统是否达到设计要求?尽管在某些情况下，如果系统具有自适应性，它就已达到设计要求，但在其它情况下，则并不一定能达到要求。因而，要通过实验来评测系统是否满足了设计要求。

5)用户是否喜欢自适应系统?即使自适应系统优于非自适应系统，但由于其它原因，用户也可能偏爱非自适应系统。例如，用户能预测到非自适应系统的响应，却不能预测自适应系统的响应，这就可能成为他们不喜欢自适应系统的一个原因。

7. 结束语

评价对于人机界面设计而言，是一关键的、不可缺少的重要环节；对于自适应系统的设计，也宜采用重复的"设计-建造-评价"循环过程，且应尽早进行评价。

建立自适应人机界面有几个潜在的不利方面，最明显的就是自适应系统会额外地增加系统的复杂性；自适应系统因易于使用而带来的好处，可能还不能补偿因监视用户交互作用并进行适当修改而需要的系统额外开销。另一个弱点是系统缺乏可预测性。传统的界面是静态的、一致的，而自适应界面将根据对用户需求的理解而改变界面以适应用户，此时界面的响应是不一致的；即使用户所作的响应与上次完全一样，界面响应也有可能改变，这就可能使用户感到困惑。

迄今为止，所开发的自适应界面均为实验性质，评价表明其性能尚不及类似的静态界面。很明显，自适应性是潜在的、非常有用的系统特色，但仍有许多问题有待研究。(参考文献共30篇略)

(上接第81页)

$|U_2|\leqslant t$，$U_1\subset U_2$且$U_2-U_1=\{u_i\}$。因为S是一步t-可诊断系统，由定理1可知：在S中存在测试边$\{i,j\}_k\in E$满足$\{u_j,u_k\}\in U-U_2$。再由G_1的定义可知：$\{j,k\}\in Y_1$。由于U_1是G_1的顶点覆盖集，u_j或$u_k\in U_1$，即u_j或$u_k\in U_2$，这导出一个矛盾。因此，S中每个顶点的阶至少为t。 □

引理2 如果系统中每个顶点的阶至少为2t，则该系统是一步t-可诊断的。

证明：(反证法)对于系统S中任二个相异的t-故障模式$(U-U_1,U_1)$和$(U-U_2,U_2)$，设$u_i\in(U_1-U_2)\cup(U_2-U_1)$。假设S不是一步t-可诊断的，由定理1可知：S中不存在测试边$\{i,j\}_k\in E$满足：$u_j,u_k\in U-U_1-U_2$，即只可能存在测试边$\{i,j\}_k\in E$满足：$\{u_j,u_k\}\cap(U_1\cup U_2)\neq\Phi$(空集合)。所以，$G_1$中的每条边至少有一个顶点在$U_1\cup U_2-\{u_i\}$中。由$|U_1|,|U_2|\leqslant t(|U_1\cup U_2|\leqslant 2t)$可知：$G_1$中存在一不超过2t-1的顶点覆盖集，即$u_i$的阶不超过2t-1，这导出一个矛盾。因此，S是一步t-可诊断的。 □

定理4 由n个顶点组成的最优一步t-可诊断系统的测试边数Y是$\theta(nt)$。

证明：由引理1和2可知，不等式$[nt/2]\leqslant Y\leqslant 2nt$成立，所以Y是$\theta(nt)$。 □

五、结束语

基于一般三元比较模型，本文给出了一步t-可诊断系统和顺序k/t-可诊断系统的特征，以及判定一个系统是否是一步t-可诊断的充分条件。并且，还给出了一个所需测试数为$2nt^2+nt$且时间复杂性为$O(n)$的分布式自诊断算法，以及最优一步t-可诊断系统的测试边数的界。

在本文模型下，下面三个问题还有待进一步研究：

(1)可诊断性问题的时间复杂性；

(2)适用于更一般的一步t-可诊断系统的多项式时间诊断算法；

(3)最优一步t-可诊断系统的设计方案。

从上述对一般三元比较模型的讨论中，我们可以看到三元比较模型中的问题的难度要大于二元比较模型。其实，在很多判定问题中也存在类似情况。例如，二元满足性和二维匹配问题均可在多项式时间内解决，而相应的三元满足性和三维匹配却是NP完全的[13]。 (参考文献略)