# 软件攻坚术：速成原型与知识工程的结合

Pamela W. Jordan, Karl S. Keller,
Richard W. Tucker, David Vogel

**摘要**

本文提出一种叫做两阶段软件攻坚术的软件开发方法，它是速成原型法与知识工程相结合的产物。使用这种方法能生产功能更多的原型，而所需时间为四个月，而不像传统原型法那样需要2年。文中论及该方法与现有原型法的差别，并以开发美国陆军的MSES为例，说明了它的要领和主要做法。最后对该方法提出了几点建议。

没有任何一项革新称得上对软件开发过程的效率和方便性做出了重大改进。[1,2]软件开发方法分为两大门派：（1）从与机器打交道着手的软件开发方法，即高级程序语言和软件开发环境[3]；（2）着手软件的设计和规格说明的方法。[1,4,5]速成原型是这两大门派的混合。它是一种评价活动，试图概括一个系统解决目标系统中涉及的某些问题的能力，[5]同时引导对新暴露的问题做进一步研究。

在Mitre-Washington人工智能中心，我们试验了一种快速生产高功能原型的方法。我们对这一方法杜撰了软件攻坚术(software storming)这一名词，它涉及专家在紧张的开发工作中参与系统的初期设计与实现，把知识工程和软件开发技术与工作站硬件的最新成果相结合。这里所描述的试验是对人工智能领域中所发展的工具和技术的一种测试，也是软件攻坚术形成的第一步。使用软件攻坚术，我们花了不到四个月的时间，开发了一个比标准原型功能性强得多的软件原型。

---

言值，那么一般$l^2$不在$X$的术语集合中。

寻找$X$的一个语言值的问题（它的意义近似于$U$的一个给定的模糊子集）被称为语言近似的问题（Zadeh,1975c, Wenstop,1975; Procyk,1976）。在这篇文章中，我们将不讨论如何处理这一相当重要的问题，但是将假定，语言近似是隐含在一个可能性分布重新翻译成用自然语言表示的一个命题的过程之中。(见(2.10))

（未完待续）

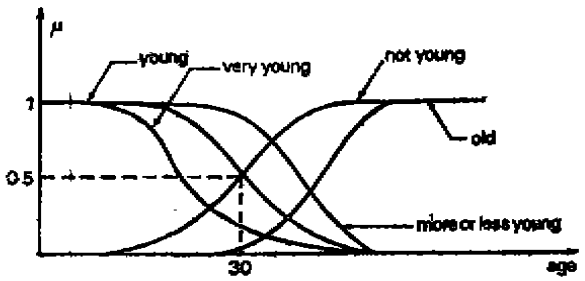



图1 Age的语言值的图形表示

〔金雅芬 译自《Machine Intelligence》(J. E. Hayes, D. Michie, and L. I. Mikulich, eds) Vol. 9, Elsevier, New York. pp 149—194, 1979, 高 钟扬〕

## 一、现有的原型法

原型法的主要目的是减少建造优质系统的时间和开销。原型法遵从系统增量开发原则，它包括端点用户评价新生系统的能力。端点用户的不断参与，导致了系统的更快开发和最终导致更有用的系统。[1]尽管原型法的一些最新革新[1,4,8]暗示着端点用户参与的重要性，但远未充分地让端点用户卷入软件开发中去。

与原型法不同，知识工程技术通常利用领域专家（所论问题方面的专家）作为端点用户的代表。[7]知识工程期中的原型开发得通过领域模型的增量求精，一般要拖延2年以上。

Waterman[8]对专家系统的演化给出了以下五个阶段：

·演示性原型，这种系统可以解决部分问题，并用来向管理部门和预期的主办者显示专家系统的可能价值。

·研究性原型，这种系统能处理问题的大多数方面，尽管某些方面还需要进一步开展研究工作。

·现场原型，这种系统支持问题的所有功能方面并由端点用户不断测试其效率和实用性。

·生产性模型，这是一种完整的系统，它已经过选中的端点用户的全面测试，但不作为商品销售。

·商品性系统，这是一种准备上市销售的系统。

我们的试验使用了软件攻坚术来开发演示性原型和研究性原型。其它大多数原型法旨在用于原型系统开发的后几个阶段。这些方法一般是些高结构式方法，类似于软件的结构式设计和开发。软件攻坚术很不同于这些方法，它依赖于知识工程期间的智力攻坚，也就是一种固有的非结构活动。

Buchanan等人描述了开发专家系统的五个主要阶段。[4]软件攻坚术类似于专家系统开发过程，在紧张的为期四周的工作中，使用了专家系统开发的变通阶段。图1给出了专家系统开发(ESD)阶段和软件攻坚术(SS)阶段的比较。软件攻坚术和专家系统开发与传统原型法两者的主要差别在于软件攻坚术受到了时间限制。软件攻坚阶段是整个原型开发周期中的一个被高度压缩的步骤。

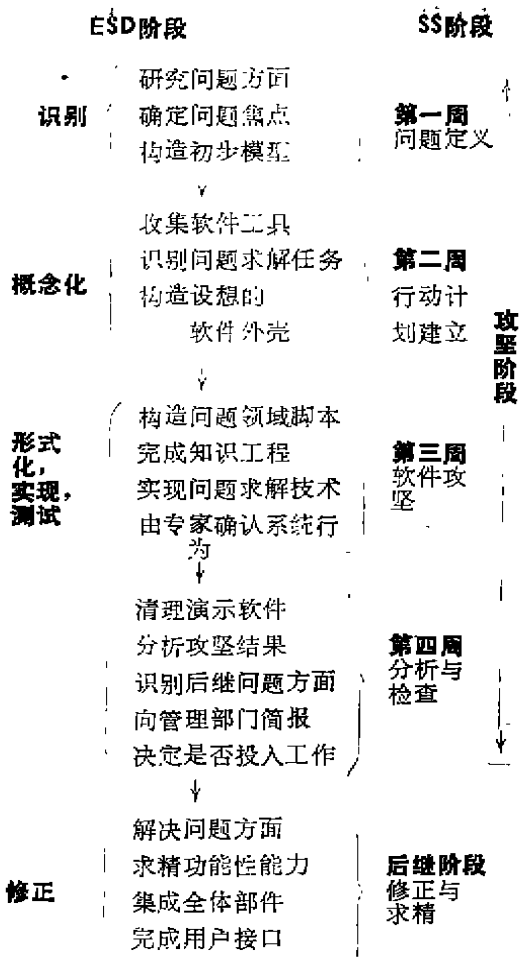



图1 ESD阶段与SS阶段的比较[4]

## 二、何谓软件攻坚术

除了其时间限制外，软件攻坚术与其它原型法的差别还有以下三个主要方面。第一，在软件攻坚术中，领域专家直接汇集到软件开发过程中。他们和知识工程师携手共同工作，把知识利用到系统中并评价系统行为。软件攻坚术的基本要领是领域专家和知识工程师结合在一起，为智力攻坚，问题求解技术而努力，一组成员自发地贡献自己的思想。

照Osborn的看法，智力攻坚是"使用人的智慧来攻克富有想象力的问题，并采用突击队的方式，每个攻坚队员大胆地朝着同一目标进取。"[8]通过智力攻坚，使用人工智能技术和灵活、有表达能力的支持工具(如AI工作站）可以迅速完成知识工程。

软件攻坚术与其它方法的第二个差别是，攻坚活动被录制在录象带上，因此可以将人们对问题领域的新发现存档并为改进攻坚技术提供反馈。第三，攻坚过程由两个不同的阶段组成：紧张的开发攻坚阶段和把初始原型扩充为可以演示的原型的后继阶段。

为有效地把领域专家汇集到软件小组中，得要求尽量缩短软件改进的周转时间，以便给专家迅速显示他们的知识实现情况。一种通用的知识工程技术是在专家执行类似任务的时候分析他们的表现行为。软件攻坚术通过综合表现行为分析和领域专业知识的软件实现，改进了这一技术。在基于知识的系统开发中，系统一般描述了领域的一个模型和一种问题求解策略。在软件攻坚术中，基于知识的领域模型起着增量求精问题求解策略的一个实验室作用。问题是由领域专家从考察的新生模型中抽出的，这些专家给开发者提供直接反馈。

软件攻坚中的基本活动是在电视上捕获攻坚事件，不必进行信息存档和编制资料，这些对于软件开发过程常常不是统一的。录像带是录音带的直接拓广，是为知识工程师熟知的方法。[7]视频副本包含攻坚期间所保存的知识工程期的全部历史。更重要的是，电视提供了工作人员相互联系和进展的成败都显示的公正记录。因此，参与者可以改进他们的软件攻坚方法。

攻坚阶段和后继阶段两者的差别有几个方面。其主要差别是涉及的人数和模型构造的进展速度。一个软件项目要开始类似于惯性物理学：项目开头难。根据这种现象，攻坚阶段需要问题领域和软件领域中大量重要的专业知识。后继阶段以降低人员配置级别进行。这个阶段包括清理攻坚阶段中所开发的软件，增加由攻坚结果所产生的功能，并集成所有部件。

## 三、攻坚的问题

我们的试验已把软件攻坚术应用到了作战通信问题上。美国陆军最近开始把价值数十亿美元的所谓流动电话用户设备系统(MSES)的数字通信系统装备到战场，这是陆军使用的最大系统。[10]MSES满足派遣给部队的流动用户和固定用户的通信需要，这些部队包括军、师作战区到营部和独立连这样的部队。

固定电话用户在他们的指挥所使用硬布线电话。固定电话用户大约是流动用户的四倍。流动用户使用流动电话用户无线电话终端(MSRT)与其他流动用户和固定用户进行通信。MSRT的工作方式与商业上的细胞式电话相同，但不同于工业上的细胞式电话，陆军是在作战的动态变化环境中使用的，因此不能奢望在经验上确定把设备放在何处最好。

图2示出了MSES的一般配置。任何两个用户间的通信是由（为找到一条最优网络路由而设计的）直视搜索(flood search)通

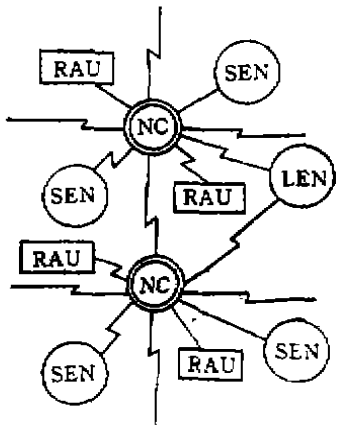



图2 MSES的一般配置。NC—电话总机结点；LEN—大型电话分机结点；RAU—无线电访问机；SEN—小型电话分机结点。

过视线(LOS)汇接成的骨架栅格网络的电话总机结点交换机而确立的。另外一些类型的设备把局部用户小组与电话总机结点连接起来：电话分机结点代表固定电话用户，无线电访问机(RAU)代表流动电话用户。

**多约束的处理** 部署MSES的关键问题是，在给定的军队情况，设备特征和地形条件下，如何在战场上布置许许多多的设备。这个问题可以陈述如下：假定约有40个电话总机结点交换机和350台外围设备，在整个35000平方公里的军、师区域内，将提供什么样的配置能最好地覆盖大约10000个流动的和固定的电话用户呢？确定什么"最好"是困难的，因为必须考虑多种常常相互冲突的约束，其中包括：

·必须支持所有固定电话用户。

·在预期流动电话用户出现的区域内，应使无线电覆盖面达到最大。

·为了尽量少用中继设备，应使LOS汇接达到最大。

·通过达到最大的连接性，应使网络的脆弱性降至最小。

大型和小型电话分机结点(LEN和SEN)为固定电话用户服务，并按照它们所支持的固定电话用户数目分配给军队。通常，每个LEN被连接到两个电话总机结点，而一个SEN只连接到一个电话总机结点。一个LEN可以支持的固定电话用户比SEN支持的多。电话分机结点被簇集成一个个小组（每组四到六个结点），并且被连接到同一电话总机结点，就象围绕一个活动中心讲话一样。对这些连接，使用电缆或LOS汇接更好。若有必要，可以使用附加的LOS设备来中继信号。

无线电访问机为流动电话用户服务。通常，两个RAU各连接到一个电话总机结点，并且至少一个使用电缆汇接。一个RAU使用一种全向信号来延伸到可能的流动电话用户。

**问题的难点** 区域无线电覆盖子问题就是一个复杂问题，因为通信信号受传播地形的影响。信号损失的算法计算离不开一个必不可少的传播路径损失的数学模型。为了理解这方面的覆盖问题，需要进一步描述一下MSES。

区域无线电覆盖与RAU的位置有关。RAU覆盖的区域叫做它的足迹。图3示出了RAU的足迹。RAU足迹是其中MSRT可以进行通信的一种区域。诸如森林密度、崎岖和都市化等地形特征会严重影响RAU或MSRT接收的传播信号的损失。还没有任何分析方法能完成所需要的计算。因此，必须获得信号传播方面的领域专业知识以提供启发式知识，使系统能对信号在地形上造成损失进行推理。

**解决问题的方法** 在攻坚阶段期间，我们对MSES设计提出了以下两组计算子任务：

数据要求：

（1）形式化脚本，包括军队情况和地形数据。

（2）在给定的决议案下对RAU的各种可能位置，预先计算RAU的足迹。

（3）在给定的决议案下预先计算相距一规定距离地点间的LOS覆盖。

功能要求：

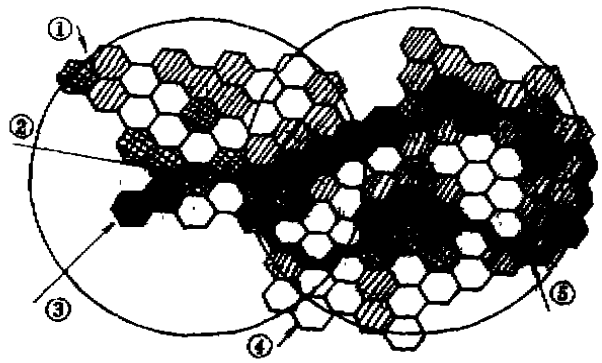
（4）为给固定电话用户提供服务，设置LEN和SEN。

（5）为形成骨架栅格网络并连接到LEN和SEN，设置电话总机结点。

（6）针对最好覆盖设置RAU，并把它们连接到电话总机结点。

上述一系列子任务规定了软件周（即攻坚阶段的第三周）期间解决问题的次序。子任务的次序是由它们间的相互约束所确定的。特别要注意的是，第一组子任务包括数据收集或数据生成任务，第二组子任务包括领域专家的初步目标。这三个子任务的次序是按这样的问题求解准则定下的：首先着手问题的约束最多部分。电话分机结点受部队指挥所在战场上的位置的约束。电话总机结

受LOS连接到LEN和SEN以及连接到其它电话总机结点的约束，而RAU受它们到电话总机结点的电缆或LOS连接的约束。计算RAU足跡(第2步)是一项主要的知识工程工作，因为如上所述，求解这一问题，还没有分析方法。




①圆：理想条件下的RAU覆盖；②圆点：无线电访问机；③深色：高障碍物；④浅色：低障碍物；⑤粗黑线：RAU足跡边界。

图3 RAU足跡

## 四、软件攻坚要领

正如我们所叙述的，速成原型的软件攻坚方法分为两个阶段。我们建议，攻坚阶段要短，约为一个月，因为这个阶段十分紧张，疲劳和进度约束使人们难以维持更长的时期。投入后继阶段的时间量取决于所解决的问题，在最终系统功能方面所希望的细节，当然还有系统急需程度。这一节，我们将向想要使用软件攻坚术来建造原型的人们，介绍一下我们的试验的资源和进度要求。

**人员配置** 在开发Mitre-Washington AI技术中心的基于知识的系统软件时，有五位领域专家参与了我们项目的攻坚阶段。他们中有一位也是战场脚本专家。此外，还涉及四位Mitre军事通信专家。软件专家无论在民间通信规划或在军事通信规划方面以前都没有什么经验。技术管理人员鼓励公开讨论攻坚周期间的问题和成就(下节细谈)，并把这方面的资料全都录制在录象带上。

基于知识的系统的软件专家负责知识工程工作，他们拜访通信领域专家，学习他们如何执行任务并将这种知识编码成软件。挑选有丰富经验并且能合作而不带个人情绪的知识工程师来实现基于知识的系统，我们认为，重要的是挑选那些从事过不同课题的人员，以便在试验中发挥各自的长处。

通信专家中有两位置身于该项目的整个攻坚阶段。他们的专业知识包括所求解问题的两个主要成分：通信的一般物理问题和军用通信设备的特征。另外两位通信专家在攻坚周期间协助关键问题的知识工程工作并确认中间结果。

在给后继阶段配备人员时，我们必须考虑谁应继续进行原型设计工作。我们推想，对后继阶段新建小组或介绍任何新的小组成员都是不明智的，因为培养新成员需要时间，体现不出软件攻坚"跳级起动"的好处。后继阶段，虽然周期较长，但需要的人不多。知识工程小组中有三个成员被调去工作大约两个月，并指派一位领域专家去支持后继阶段工作。

**进度表** 攻坚阶段包括以下四个不同部分：

第一周——问题定义：背景调查，目标定义和构造初步领域模型。需要两位领域专家和两位知识工程师。

第二周——行动计划的建立：初步收集软件工具和数据，构造设想的系统外壳，和建立领域问题求解模型。

第三周——软件攻坚：建立整个系统的体系结构，实现和知识工程化脚本数据，提出军队布署方案，初步实现问题求解模型，构造初步的用户接口，和完成子任务的综合。这需要五位知识工程师和两位领域专家（此外，还需要增加两位领域专家参与某些关键问题的商议）。

第四周——攻坚分析：清理软件并向全体管理部门作简要介绍和演示。这需要两位

知识工程师。

我们将攻坚阶段限于四周时间，以便使其费用最低，并最大限度地缩小其对正在开展的别的工作的影响。时间短也便于说服参与者往往因工作延长占据所有空闲时间带来的情绪影响。对所有这四个攻坚部分，实行五日制工作周，既方便又使持续时间有限。我们的目标是到星期五停营关门时完成各周的活动，对软件攻坚不要求加班工作。

我们把后继阶段看成是一个扩展的清理和求精周期，其间我们可能把更多的细节加入初始原型中。我们不能象攻坚阶段那样详细地定义后继阶段。直到攻坚阶段之后，我们才能估计我们在后继阶段期间要做的工作。因为，只有此时我们对问题才有了更好的理解，才能估计原型需要多少额外的功能细节。

**硬件与软件工具** 在行动计划建立（攻坚阶段的第2周）期间，我们决定需要四台Symbolics Lisp计算机。这四个工作站允许同时攻坚多任务。所有软件专家都十分熟悉Symbolics Genera软件开发环境和Common Lisp。这四台计算机在我们的实验室中留着攻坚小组在软件攻坚周中专用。

对于知识表示和推理，我们选用了ART（自动推理工具），即Inference公司的专家系统外壳。尽管其他一些工具也用得上，但是我们认为，这种工具的资料具有较好的指导性且更快更成熟。在以前的行动计划建立周中，五个攻坚队员还没有一个用过ART，但他们却在商业和内部使用过其他专家系统外壳。假如发现ART太难于学习和使用，作为后备我们选中IRIS（集成表示和推理系统），它是Mitre开发和使用过的。其有利之处是有两个攻坚队员具有使用它的经验，并且源代码经过修改是可用的。

ART通过模式（亦叫框架）来支持知识表示，其中对象具有层次关系，并且对象的槽（属性）可以给定值。适当时，可以从该层次结构中较高层对象继承值。例如，LEN的每一个实例可以支持一些固定电话用户，这些电话用户数是所有LEN的类，而不是每一实例的特征。ART还提供了规则，其中前件（if句子）按模式的槽值使用模式匹配以选择确定发生什么动作的条件。这些动作被编码在规则的后件（then子句）中。后件可以增加或撤消知识库中的知识，生成图形并控制新规则的应用。

除动态窗口软件（为Symbolics Genena环境的组成部分）之外，对图形特征和图像的表示，我们还选择了MMI（即内部图形软件包）。为了使不熟悉的工具数量最少，我们决定不使用ART的图形功能。

对项目的这两个阶段，我们使用了同一组软件和硬件，但对于后继阶段，我们只需要两台Symbolics Lisp机。

## 五、攻坚阶段

攻坚阶段在两阶段软件攻坚术中是非常重要的，因为在很短时间内要完成一定量的工作，十分紧张

**背景定义与问题定义** 在攻坚阶段的第一周，我们进行背景研究以确定在问题方面已完成什么工作。以前所发表的大多数著作都局限于网络管理[11]或骨架配置，[12]而很少有关于流动电话用户网络配置方面的。

我们搞了一种结构式访问，第一周拜访军用通信专家三、四次，以便学习更多的问题方面和明确具体问题的焦点。领域专家主要对确定无线电访问机（RAU）的足迹感兴趣。软件专家认为，确定RAU足迹可能容易，并决定尝试这样一个系统，先设计一个由电话总机结点和电话分机结点组成的骨架栅格来最佳地放置RAU，并从这点出发，配置整个MSES。他们估计，RAU足迹计算约占四分之一工作量。回想起来，这个攻坚周计划太雄心勃勃了，

**行动计划的建立** 我们花上攻坚阶段的第二周来生成领域问题求解模型，并挑选前

面所述的硬件与软件工具。我们也开发了一个设想的系统(即表示用户接口与系统功能第一次分解的软件框架结构)。我们现在才知道,我们应花更多的时间,用含有地形和军队信息的脚本数据来给设想的系统添上血肉。这一失误的影响将在后面叙述。

在第二周结束时,我们从领域问题求解模型中抽取了问题的模块子任务。我们把这些子任务分配给各个小组成员,让他们在第三周根据自己以前的经验和表现出来的兴趣去加以实现。我们对子任务排定了优先次序,选择了可能简化的方面,并弄清了在问题实例中寻求退却办法。我们认为RAU布局是最重要的任务,而完成用户接口就不重要了。这些退却办法包括:如果ART证明是一个障碍,就转换到IRIS;如果脚本构造很麻烦,就编制地形与部队数据而不是试图坚持就事论事。

**软件攻坚周** 随着模块子任务被弄清和排定优先次序,我们就准备好了开始实现周(即软件攻坚)。由于刚好只分配一周时间,所以我们强调完成给定的任务,而不要求软件清晰和资料完整。在第四周和后继阶段,我们的工作是清理软件。

在软件攻坚周,用一台电视摄像机记录小组成员间的相互联系。记录约14小时的攻坚工作,包括一小时报告和我们认为重要的会议,如象小组成员间富于洞察力的相互交往。前三天,小组认为磁带录像是一种干扰,因为它中断攻坚队员的工作,要求他们面向摄像机讲述和演示工作进展。然而,以后几天,攻坚队员对摄像机变得更适应,当他们开始更好理解它的价值时,就自然把它作为攻坚过程的一个组成部分。

在攻坚之后,电视录像带提供了软件小组的小组动态和软件工程师与领域专家相互联系的一种公正记录。

小组中的相互联系:电视录像带记载了小组成员交流思想所涉及的困难。同一想法在听众实际记住之前,在几小时或几天内可能要表达若干次。有时为了理解一种想法的全部影响,听众并不完全知道所表达的是什么;而有时听众有完全不同的想法,并立刻拒绝考虑某一具体问题的其它想法。

电视录像带揭示了软件攻坚周第一天的某些混乱情况。即使上星期五已经拟定了任务分配,但某些小组成员仍有这样的问题:"我从哪里开始干?"对于每一个安排的任务,在我们没有弄清几个短期子任务之前,进展是缓慢的。小组成员只有完成了一些短期子任务,然后弄清了一些新的子任务,这才走上正规的工作节奏。我们定期集成几个子任务的软件以便确定进一步需要做什么工作。

电视录像带能使我们观察小组合作行动的心理状态。我们认为,在某些情况下,任务被分配错了;也许有另一个人比所指定的人对一个问题更感兴趣且工作能力更强。我们还认识到,有时单独一个人工作比同另一个人合作的效率更高;向另一个小组成员解释自己解决某问题的方法可能只会增加困难。较好的做法是,在完成任务后,才向其他人解释。不过,当谁都不能确切知道在困难的技术场合怎么办而又有必要进行探讨时,两个小组成员一起工作,可能效率更高。

当用ART实现一些事情而似乎又无法编制文档时,小组成员使用了复杂的Symbolics软件排错工具来查明什么是必须完成的。当我们发现了Mitre MMI软件包中的错误时,我们除了排除这些错误之外别无它法。当我们的分析似乎证明我们计划使用的地形数据与我们感兴趣的区域不符合时,我们编制了数据来代替之。小组深刻意识到建立一个原型所花时间有限,而且有强烈的动机来完成这件事。

知识工程工作:领域专家是软件攻坚小组必不可少的成员,并且对知识工程工作会议起主要作用。但是,象大多数专家系统外壳一样,ART也需要有关高效开发规则的软件

专业知识，因此由我们的领域专家去使用而没有软件专家的帮助是不合适的。因而，软件专家应对开发过程给与指导。

如果有效工作时间不多，极为重要的是，领域专家以及开发者应集中在攻坚上，而不要分散它。我们的知识工程技术是访问领域专家，弄懂他们的问题求解方法并且在递增生长模型范围内实现这些方法。这种途径允许领域专家立即看到他们的知识操作在系统中的结果。快速反馈促进了专家进一步研究问题。这方面一个好的例子是RAU足迹计算模型的推导。

计算RAU足跡这一任务并不象我们所预料的那样简单，结果在这一周导致了召开一些主要涉及知识工程的会议。其它一些计划中的任务可能需要复杂的知识工程，但时间不允许我们对其开展研究工作。计算RAU足跡的会议是从向专家提出下列问题开始的："你怎样计算RAU足跡？"领域专家描述影响信号衰减的一些环境因素，并且提出表示因这些因素产生的信号损失的图形和方程组。最初开发者对RAU全向信号的传播特性会产生某些错误概念，因此，他们会过分简化计算RAU足跡的高级描述。经过有耐心的领域专家反复解释之后，开发者能够认识到自己的错误。

在测试期间，有时开发者发现衰减损失高得可疑，并要求领域专家回头去检查结果。尽管领域专家断言这些衰减损失是正确的，但他们也为这些结果烦脑，并与一个没参与攻坚的专家进行商量。专家们发现衰减损失中有些因素是计数了多次，它们返回给能解释错误的新的专家，而且我们迅速纠正了这个问题。

最初的知识工程会议持续三到四个小时。以后的会议每次持续约一小时。会议遵循这样一个模式：在与领域专家讨论了一个问题之后，软件工程师就根据讨论建造软件；他们给领域专家演示所得到的软件，提供反馈并提问新的问题。最后我们得到令领域专家和开发者都满意的解决办法。

问题：在攻坚后，工具、脚本数据和规程等的问题大大减慢了我们的进展。然而，如果有了适当准备，我们可以在今后的项目中避免这些问题。

两天中，我们使用ART系统受到了大的挫折。我们认为应转换到我们的内部推理系统(IRIS)上来。对ART我们没有人以前有过经验或培训过，所以毫不奇怪，我们使用它会有困难。尽管ART的资料质量优良，但我们不得不尝试解决某些有关其行为方面的问题。因此，迅速完成工作是困难的。尽管如此，我们决定继续使用ART，因为我们已经解决了许多由于我们的错误理解而不是软件限制所产生的问题。到攻坚周的星期四，我们在ART方面遇到的大多数麻烦已被克服。显然，我们完全应在以前几周中花更多的时间来学习ART。

在整个攻坚周中，我们发现，我们需要的数据并不熟悉。首先，地形数据就有问题。我们从数据库中检索的地形数据与区域的地图不一致。在攻坚周中我们不可能纠正这个问题并且不得不编制数据来测试地形信息要求的成分。后来在后继阶段，我们发现，当小的地形单元归组成大的单元时，则大的单元就可以相对于小的单元旋转，而我们的地形解法，并没有使用适当的旋转。如果在前一周我们不如此匆忙考察地形数据，我们也许能发现这个问题及其原因。

另一个问题是获取军队数据来导出电话分机结点的布局以支持固定电话用户。我们花了大量时间，从不大详细的脚本出发，开发了一个详细程度满足要求的脚本。在第二周最好是搞更多的设想开发，以便为脚本定义和输入军队信息。

最后，有些小组成员关心，由于使该项目局限在某种特殊技术或方法学，那么在攻坚阶段所做的快速决策可能对后继阶段工作造成不利影响。例如，知识表示专家在攻坚中建议，应改变推理规则，使得它们不涉

及我们的六边形栅格地形表示，因为这些推理可能使我们的通信领域专家产生混乱。他们很可能更熟悉正方形栅格，也因为在后继阶段工作中，我们想抛弃六边形表示，而赞同另一种表示法。由于在攻坚周软件小组高度集中，这种建议不会得到十分认真的考虑。小组认为这种建议会分散注意力，可能在前一周或在后继阶段中更有用。

**攻坚阶段成果** 到攻坚阶段结束时，我们获得了电话分机结点布局、RAU足跡计算和基于高度的简单视线计算的规则和方法学。攻坚阶段另一些成果包括形成了战场脚本，对六边形地形表示法的支持和完成了初始用户接口。我们估计上述成果约占我们计划达到的功能的20%。

在攻坚阶段结束时，我们大大缩短了到达目标的距离，即实现电话分机结点和电话总机结点的初始布局以及RAU布置方法接近专家的方法。然而，当我们演示我们所达到的功能时，领域专家给出了两条生动的评语：他们的印象是，我们能通过软件理解问题和提供反馈速度相当快，即使它生成的功能不多，但是就该通信规划问题而言要比当前现有的软件帮助更大。

## 六、 后继阶段

与攻坚阶段对照，后继阶段使用了传统的软件原型法。然而，它的速度和成功由于攻坚阶段的跳级起动作用而得到大大增强。

在后继阶段，我们求精和完善了ART这样一些规则：计算足跡，将设备分配给固定电话用户和计算LOS。然后我们解决为流动电话用户服务而设置电话总机结点设备和RAU设备的问题。这时，我们需要LOS和足跡计算的结果。我们已准备预先计算足跡和LOS数据，但是当我们试图对所有数据集这样做时，我们遇到了所需存储器容量和运行时间的问题。由于我们不能预先计算这些数据，所以我们需要改变我们的方法。我们决定把ART规则翻译成Lisp过程，其理由有二：第一，我们决定LOS计算不需要规则，因为它们都是些简单计算。第二，我们预料，计算足跡和为固定电话用户分配设备都不需要进一步求精。

我们能够如此容易地把规则翻译成过程，说明了专家系统方法在开发原型软件中的价值。我们顺利地修改和增强了这些规则，直到所有细节都包括进去并且正确的行为得以实现。由于所需的所有信息都在规则中，所以它是一个简单的翻译过程，也许比一种高级语言到另一种高级语言的翻译更简单，因为控制机制对知识的使用不会有不利的影响。

直到我们进行了放置电话总机结点和RAU之后，对原型法使用专家系统规则的价值才明朗起来。我们明白，当我们预先计算足迹和LOS数据时，我们遇到了这一任务的运行时间和存储容量的问题，因此我们在开发功能时应从过程性方法着手。但是随着开发的深入，我们不得不继续给软件打补钉以进行某种求精。Symbolics Genera软件开发环境具有增量开发风格，对于求精代码十分容易，不管代码在规则中或是在过程中。然而，编码规则中的知识由于规则的说明性质而使维护的清晰性和容易性受到限制。回想起来，我们应使用规则来开发新的软件，并且在我们进行攻坚阶段的足迹和LOS计算时，使用很少量的测试数据实例来避免存储容量和时间问题。在完成求精后，我们能够把规则翻译成过程以处理整个数据集，这时我们就能完成了足跡和LOS计算。

**知识工程工作** 在后继阶段，不象在攻坚阶段那样，我们不要求与领域专家进行十分紧张的相互联系，因为我们在攻坚阶段获得了绝大部分关于MSES问题的知识。但现在我们认为，我们对后继阶段与领域专家的相互联系做了某些改进。例如，在选择和实现一种方法之前，我们可以讨论我们放置电话总机结点和RAU的方法，并且可以经常短时间地演示我们的进展。有了这些步骤，我

们可以听取到领域专家的更多有价值的见解。

**问题** 在后继阶段我们遇到的唯一重要问题是ART。如前所述，为装载LOS和足跡计算的地形数据所需的存储器和实际计算前为重置规则所需的时间都高得来不可接受。在我们的整个地形数据库中使用ART规则来预先计算有1.000单元的LOS和足跡，消耗了大量的虚拟存储器。通过增加虚拟存储器，我们最终也能完成将规则应用到数据上。然而，对于规则的另一种应用而重置ART事实库得花一个小时以上。在Inference公司，我们向人们谈了我们的问题，人们确认ART能够处理大的数据库，最大到其知识包括约10,000,000个事实。我们的数据库接近40,000个事实，代表了地形单元、军队和流动电话用户设备。我们的结论是，我们对ART比较缺乏经验，加之我们的数据库规模有限，有可能对性能问题产生影响，致使我们的速成原型工作放慢。

在后继阶段，我们对ART进行了几次试验，试图确定性能问题产生的原因。我们对一条与数据库中每一项匹配的规则使用了所有地形数据集。我们发现，ART用约为1到3分钟加上分页时间(1至4分钟)可以重置。有了这些结果，我们假定在我们的一条或多条规则中一定存在某种过失。然而，我们的进度表不允许我们避开这个问题，仍继续迅速推进原型设计，我们把ART规则翻译成了过程软件。

尽管我们对ART的经验有限，但是我们认为，当在少量数据下使用这种工具时，它对速成原型是特别有用的。在这种情况下，可以十分迅速地增量求精规则。但是，如果不是至少有一个小组成员很熟悉这个工具，那么使用它就是一个错误。

**后继阶段成果** 在后继阶段，主要收集攻坚阶段产生的完整解和部分解，并把它们集成为一个为陆军部队配置MSES的适当精加工过的系统。后继阶段产生我们计划达到的其余80%的功能。

## 七、 对攻坚术的几点建议

在很短的时间周期中，我们的软件攻坚试验取得了两个重要的成绩：一是知识工程师学习了大量问题领域知识(MSE)；二是实现了专家的问题求解技术。尽管我们对软件攻坚术只有点滴经验，但我们有以下几点建议：

**(1)了解你的问题** 软件攻坚术不大适合于病态问题或范围广泛的问题。攻坚术不允许有时间去研究，尽管研究问题在攻坚过程中可进行查明。攻坚方法要求有专家参加，因为他们知道如何解决他们领域中的问题。在第一周(问题定义周)，必须集中于攻坚期间尚待解决的问题。

**(2)了解你的小组** 在攻坚周中，小组成员间的合作对攻坚的成功有重大影响。在攻坚中，每一个攻坚队员必须迅速适应角色的变化。我们建议，小组成员能够：

- 认识到一项任务何时应由小组某成员独立完成，何时需要小组合作。
- 在问题没有取得进展时，能让给另一个成员去完成。
- 采纳其他成员的方法。
- 承认错误。

**(3)了解你的工具** 如果我们都有使用ART的经验，那么在攻坚周中，我们可能会取得大得多的进展。如果不熟悉的工具看来是适合的或需要的，那么必须分配时间去掌握使用它们的专业知识。

**(4)了解你的数据** 数据环境形成了软件开发各个方面的基础。必须充分理解数据量、数据表示和数据存取功能。例如，在攻坚周以后几周，我们发现地形数据库是正确的，但我们对它解释不正确。在攻坚周之前，细心检查数据，可能会发现这种错误解释。

**(5)资料录入电视** 我们建议，在攻坚

# 面向对象语言的谱系

Peter Wegner

本文深入讨论了面向对象的程序设计语言。根据对象、类、继承性、数据抽象、强类型、并发性与持续性等语言特征，作为语言空间的设计量纲，讨论了语言的分类和层次关系。着重阐述了基于对象的语言、基于类的语言和面向对象的语言之间的联系与区别。

由于六十年代后期的软件危机，使得结构和简明性而不是表达能力成为语言设计的基础。软件工程学科应运而生，提出了一些结构程序设计与结构化设计方法论作为应用程序设计的基础。人们认识的重点从程序当作语句序列而转到了程序当作相互作用模块的集合。美国国防部为解决软件危机而开发的Ada语言支持了包括函数、过程、程序包、任务与类属程序单元在内的各种模块。

第一个面向对象的语言是六十年代中期开发的Simula，其特征是类的概念。它的实例由操作，协同程序和子类的集合组成，其中操作带局部状态，协同程序通过"resume"操作模拟并行执行，子类继承父类的操作与状态。

本文是作者对自己在OOPLA '87会议上的论文[1]所描述的概念作了增改而写成的。阐明了面向对象的程序设计的基本概念，并分析了传统的面向对象的顺序式语言的设计方法和适合整个面向对象程序设计的并发持续式(concurrent and persistent)语言的设计方法。

---

周使用电视录像带来编制知识获取过程的资料、特别是捕获由小组发现的软件攻坚技术。我们使用了电视录像带来回顾知识工程会议，研究攻坚实验，和教育其他工作人员掌握攻坚技术。此外，在第四周结束时，我们用电视录像带录制原型的演示情况以完成我们的视频资料工作。

软件攻坚术是一种迅速抓住棘手问题的有用工具。这种工具使管理部门可以考察，在一个组织调配时间和资金的主要开销之前，预计如何能够解决问题，或问题是否确实可以解决。软件攻坚术与速成原型的其它方法之区别在于：其时间结构大为缩短，只需3到4个月而不是1至2年。在这个实验项目的两个阶段中，总共花费的工作量为10个人月。

速成原型的一般结果是一个主要包括用户接口的系统外壳，而攻坚术将产生功能多得多的系统。在后继阶段后不久，在一次陆军通信会议上，我们演示了我们的原型。出席这次会议的陆军MSES设计者认为，我们的系统已达到设计者所需要的约75%的功能。

总之，我们发现，软件攻坚术是一种跳级启动原型设计项目的廉价而高效的技术。

参考文献(略)

〔中　原译自《COMPUTER》VOL.22 NO.5 1989 P39—48，王　心校〕