

# 软件生长过程模型与 Web 软件开发<sup>+</sup>

朱 鸿

( 国防科技大学计算机科学系 , 长沙 410073 )  
Email: zhuhong6129@263.sina.net

**摘要 :**本文提出了一个软件开发的生长式过程模型 , 指出软件演变的基本动力是用户需求的变化 , 软件生命过程的多样性是必然的 , 软件生命过程是可规划的 , 引入了软件生命周期策略的新概念。文中分析了 Web 软件的性质 , 以及当前软件过程模型和开发工具对 Web 软件开发支持的不足之处 , 据此提出了适合于 Web 的生命周期策略 - 生长策略 , 并根据支持在生长策略下开发 Web 软件对软件工具的需求 , 设计了软件生长环境模型。

**关键词 :**Web 软件 ; 软件工程 ; 软件过程模型 ; 软件进化与生长 , 生命周期策略 ; 软件环境与工具 , Agent 技术。

**中图法分类号:** TP311.5      **文献标识码:** A

进入 Web 时代以来 , 软件研究与实践出现了一系列新动向。因特网和 Web 技术带来了一个具有分布式、超媒体、自治、协作等性质的应用软件实现平台 , 刺激了许多新的计算机应用领域的发展 [1] 。然而 , Web 软件的开发和维护面临着巨大的挑战 [2] 。当前 Web 软件开发实践与传统软件开发在许多方面有显著区别 [3] 。例如 , 与传统软件相比 , Web 软件开发队伍规模小 , 开发周期短 , 力求尽快地把产品推向市场 , 而不像传统软件开发那样要求以最低的开销开发软件产品。开发过程混乱、产品质量低下是当前 Web 软件开发的普遍现象 [4] , 传统软件生命周期和过程模型在 Web 软件的开发中遭到了抛弃 , 传统的软件开发方法和工具显得软弱无力。 Web 软件的开发急需方法学指导和工具支持 , 人们开始重新思考软件开发的基本原理和基础 , 极端程序设计 [5] 和轻快软件开发 [6] 就是其中典型的代表。

本文将应用软件生命周期理论 , 探讨 Web 软件的特点 , 寻求适合 Web 软件开发的过程模型和工具支持。

## 1. 软件生命周期的生长模型

软件过程模型具有两大作用 : 一是对软件生命过程的认识理论 , 它在一定的抽象层次上刻划软件生命过程的客观规律 ; 二是指导软件开发 , 例如 , 提供软件工具设计的依据。在文 [2, 7] 中我们提出了软件生命过程的生长模型。这一节我们从认识理论和指导作用两个角度概述并分析该模型的特点。

### 1.1. 软件生命过程

如图 1 所示 , 该模型把软件生命周期分为三个阶段 : 萌芽期、成长期、和衰亡期。萌芽期包括了软件系统从概念形成到第一个可使用的系统开发完成的过程。这个过程可以通过各种软件开发过程模型来完成。成长期包括了软件在使用过程中不断发现新需求、不断开发并集成新构件并淘汰旧构件的成长过程。其主要活动包括新需求的发现与识别、新需求的重要性分析、新需求与现系统的相容性分析 ( 即分析新需求在现系统中的实现是否能行及其开销与收益的比例等问题 ) 、新构件的开发及其与现系统的集成、无用构件的淘汰、过时构件的更新 , 等等。值得指出的是 , 多个需求可能以并行的形式被识别、分析、实现和集成到现系统中去。在这种软件并发成长的情况下 , 还必须分析新需求之间的相容性。衰亡期是一个软件系统在不能满足新需求的情况下 , 被淘汰而停止使用的过程。

软件生长模型把用户需求的变化看作是软件演变的基本动力 , 也是确定新软件的诞生和已有软件的衰亡的决定性因素之一。而用户需求的变化成为对软件进行修改的开发行为 , 还必须满足可行性和必要性两个基本条件。我们认为 , 无论是商业还是非商业、企业还是个人的软件开发活动 , 都应该满足行为心理学的基本规律 , 例如 , “能够做

---

<sup>+</sup> 收稿日期: 2002-09-16; 修改日期: 2002-10-8

基金项目: 国家 863 高科技发展计划资助项目 2002AA116070;

作者简介: 朱鸿 (1961-), 男, 江苏人, 博士, 教授, 博士生导师, 主要研究领域为计算机软件方法学, 软件工程.

且愿意做才有行为”的规律。

软件生长模型的基本原理之一是承认软件过程的多样性的存在性，强调过程模型必须能够包含软件生命过程的各种具体表现形态，而且应该能够说明软件开发过程出现某个具体表现形态的原因。例如，对于 Lehman S - 类型的软件，“程序要求满足预先指定的规约，其正确性是程序与规约之间的绝对一致性” [8] 因此，S - 类型的软件的唯一可能的‘新需求’是改正程序中的错误，软件演变的过程也就是纠正错误的维护过程，其中主导性的变化是性能和质量的提高。再例如，对于 Lehman E - 类型软件，“它所解决的问题和实现的应用属于现实世界的领域，其正确性是由程序在一定操作条件下的行为所决定的，并制约于人们的经历、认识、决策水平、以及种种由此而蕴含的人为因素” [8]。

随着人们的认识和决策水平在开发和使用计算机软件的过程中不断提高，人们的需求必然发生不断的变化，用户的需求变化又必然引起软件演化。因此，E - 类型软件演变的过程是不断地发现新需求并满足新需求的开拓性过程，其中必然包含大量功能上的增加，而不仅仅是性能和质量的提高。

## 1.2. 软件开发中的指导作用

我们认为，软件生命周期不仅具有多样性，而且特定软件的生命过程是可以规划的、可以控制的。软件过程的生长模型对软件开发过程的指导作用集中表现为它提供了针对不同类型软件的特征，进行生命周期策略性规划和分析的手段。

例如，对于 S - 类型软件，由于其规约的稳定性，软件的变化即改正错误，变化越多，开销就越大。因此，规划整个软件生命周期的策略应该以尽量减少改变为重点。从成长模型来看，达到这一目的的唯一途径是所开发的种子系统应该尽量满足规约。也就是说，对于 S - 类型软件，较好的生命周期策略是着力于种子系统的开发，力求减少软件的变更。我们称这一策略为静止策略。

对于 E - 类型软件而言，受用户使用计算机的经历的变化和外在条件的变化等因素的影响，其用户需求具有变化的持续性等 7 条规律 [8]，因此，为满足用户的新需求而产生的软件演变是不可避免的。适用于 S - 类型软件的静止策略不再适用于 E - 类型。而适用于 E - 类型的较好的生命周期策略则是：(1) 在种子系统的开发时，不必追求满足所有的需求，而应该区分需求的轻重缓急，首先实现关键的核心需求，并在系统结构设计中，充分估计今后对系统进行扩充、修改等变更的要求。(2) 在系统使用过程中，充分听取用户的反馈意见，从而识别用户在使用中产生的新需求，并根据其轻重缓急，选择核心需求加以实现，而不必追求满足用户所有的需求。每一次更新都应该看作是其漫长的生命周期中的一小步，因此，每一次更新都必须为今后的更新作好准备。也就是说，对于 E - 类型软件，较好的生命周期策略是时刻准备着变更，不必追求完美，从而使变更的开销降低，软件的生命周期延长。我们称这一策略为生长策略。

值得指出的是，除了软件的 Lehman 类型之外，特定软件的生命周期规划还必须考虑许多其它因素，例如，开发资源、开发能力、功能与非功能需求、进入市场的时机，等等，我们将另文探讨。

## 2. 适合 Web 软件开发的生命周期策略

这一节首先分析 Web 软件的特性，然后应用软件生长过程模型理论分析适合 Web 软件的开发过程。

### 2.1. Web 软件的性质

Web 软件在许多方面与传统软件有重大区别，认识 Web 软件的性质是解决 Web 软件开发中存在的问题的关键。下面讨论 Web 软件的性质。

(1) 宏观需求：由于电子商务、网上教育、电子政务等应用领域的出现，人们对 Web 软件所能够带来的经济价值和社会价值给予很高的估计，尤其是在企业界，人们认为 Web 软件是开拓市场、保持和提高竞争力的必要工具。人们对软件价值的这一认识不仅造成了对 Web 软件需求的迅速膨胀，而且还带来了许多更深刻的变化。

(2) 决策因素：由于人们对 Web 软件价值的认识，在抢占市场的动力驱动下，Web 软件开发往往力求尽快地把产品推向市场，而不像传统软件开发那样要求以最低的开销开发功能全面、质量良好的软件。在功能、质量、开销和时间这些影响软件开发决策的因素中，时间成为了 Web 软件开发的第一要素。

(3) 开发人员：由于对 Web 软件需求的迅速膨胀，软件开发人员短缺、开销增长，这一方面迫使 Web 软件开发队伍的规模缩小，一方面使得大量缺少经验和训练的人员加入软件开发行列，如何使软件人员在开发过程中边工作边学习已成为软件研究不可忽视的课题。

(4) 技术与工具：Web 软件技术本身处于一个快速发展的阶段。在短短的几年内，出现了许多 Web 软件的运行平台、信息表示形式以及开发语言和工具等等，新技术的出现总使得 Web 软件能够实现新的功能、达到新的质量水平。因此，在谋求更高回报的动力驱动下，从新技术的出现到新技术的使用的时间延迟缩短了。这不仅要求软件开发人员必须不断地学习新技术和新方法，而且，它对 Web 软件的结构也产生了深刻的影响。

(5) 内部结构：在 Web 软件系统中，由于采用最新技术的策略，先后开发的各个构件往往采用不同的方法与技术，从而在同一个系统中也常常呈现出多样性。而传统软件设计时，往往在系统范围内采用统一的、标准的信息表示形式，针对选定的运行平台，使用一种开发方法和实现技术，以简化其实现，保障一致性。因此，传统的以支持一种方法和技术为主的软件开发工具对多种开发方法和实现技术的混合失去了效用。

(6) 运行环境：Web 软件的运行环境具有开放性和动态性，其正确运行常常依赖于万维网上具有自主性的信息资源，甚至硬件资源，而传统软件开发方法则往往假设系统所依赖的资源在系统的有效控制之下，并通过修改控制和配置管理等手段，以保障其一致性等质量要求，从而保障系统的正确和高效运行，但这些手段对于具有自主性的第三方资源无能为力。

(7) 用户：由于运行环境的开放性，Web 软件还具有用户类型复杂、差异巨大等特点。传统软件开发中行之有效的用户分析和人机交互设计以及用户培训等方式方法一部分失去了可行性。

(8) 应用领域：许多基于 Web 的信息系统，如电子商务、网上教育、电子科技，等等，它们属于 Lehman 软件分类体系中的 E - 类，人们对这些应用领域的认识还处于开始阶段。

### 2.2. Web 软件的生长式开发

如前所述，由于许多 Web 应用软件是 E - 类型的，其演变过程通常满足 Lehman 的七条软件演变规律：变化的持续性、复杂度的递增性、演化过程的自我规范性、组织机构的稳定性、变化规模的均衡性、功能的持续增长性 [8]。较适合 Web 软件的生命周期策略是生长策略。在当前万维网时代有意识地采取生长策略开发 Web 应用软件，不仅是 E - 类型软件特征的要求，它还具有如下优越性。

(1) 降低时间压力：软件的价值在很大的程度上依赖于软件投入使用的时间或投入市场的时间。这已经被软件工业界所广泛认识，也有一些软件研究新思维的重要出发点 [5]。但是，这一事实尚未得到软件过程研究主流（如过程改进的研究）的充分注意。生长策略使得大规模软件的开发过程成为逐步成长的过程，因此，可以综合分析开发时间、功能与质量、开发能力、开发开销等众多因素，合理安排开发过程，让最迫切需要的功能与性质在市场允许的时间内得到实现，从而适当降低时间压力。

(2) 减少开发风险：软件开发的风险有多种。对于 E - 类型软件来说，其中最大的风险之一是所实现的软件非用户所需求。传统软件工程理论针对这一风险的基本策略是强调需求分析在软件开发过程中的作用，要求充分分析用户需求并完整准确地以需求定义文档的形式表示需求，然后开始软件开发的后继工作。然而，如第 1 节中所讨

论的，在 E - 类型软件（尤其是 Web 软件）的开发实践中，人们往往发现用户的需求模糊不清且不断地变化，难以形成完整准确的需求定义。生长策略也强调需求分析在软件开发中的作用，但对待需求变化的风险有别于传统理论，其策略不是一次实现所有的需求，而是区分需求的轻重缓急，首先选择实现清楚的、关键的、变化可能性较小的核心需求。通过把大规模软件开发化为一系列较小开发规模的逐步生长的过程，一旦发现开发方向错误，可以及时纠正，从而较大的投资风险可以化为较小的投资风险。而且，用户在使用软件的过程中，需求也会变得明朗和确定，不必要的需求可以避免在形势不明朗的情况下错误实现，先前变化可能性较大的高风险的需求也就可能随之成为变化可能性较小的低风险的需求，成为进一步开发的核心需求，新的原先未认识到的需求也可能浮现，成为关注的焦点。这样，忽略重大需求的风险也可以在很大程度上得到避免，或者得到及时纠正。

**( 3 ) 提供学习机会：**软件开发中的另一类重要风险是技术风险。如第一节所述，由于 Web 技术本身处于快速发展的阶段，新技术层出不穷，软件开发人员需要不断地学习新技术，以实现原有技术不能或难以实现的功能和性质与性能。软件人才的严重缺乏、开发时间的短促等因素使得难以按照传统的‘先培训、后上岗’方式来获得新技术，而往往不得不‘边学习、边使用’。如何降低软件人员不得不在开发实践中学习、掌握新技术所带来的风险是当前软件研究不能忽视的重要问题 [ 6 ]。生长式开发使软件开发人员能够在生长过程中的某个阶段的较小的开发活动中学习和尝试新技术，并在阶段性产品的成功中较早地尝到成功的滋味，从而激励学习和掌握新技术的动力，即使新技术的应用受到挫折，也便于及早发现问题，总结经验教训，这样降低了风险的代价。总之，生长策略使软件人员能够较好在开发实践中学习掌握和使用新技术，降低技术风险。

**( 4 ) 改善信息交流：**开发者和使用者之间信息交流的障碍是软件开发困难的主要原因之一。软件的生长式开发使软件的功能能够逐步地递交给用户，而用户在对系统的使用中所产生的反馈也可以较早地传递给开发人员，从而使开发者与使用者之间的信息交流得到加速。如( 1 )所述，从使用者得到的反馈在生长式开发中成为确定生长方向的决定性因素，从而，软件开发者和使用者之间的交流在生长式开发中可以更为有效。这种以用户的反馈来指导软件生长方向的过程可以使软件的开发真正成为企业成长过程的一个有机组成部分，实现软件与企业共同成长。

总之，生长策略是适合 Web 软件开发的策略。事实上，许多 Web 软件的开发过程都是生长过程。如果把整个万维网看作是一个大信息系统，其演变过程充分反映了这一生命周期的特点 [ 2 ]。

### 3. 支持生长模型的 Web 软件开发工具

在生长策略下开发软件，对软件工具的功能和结构都提出了新的需求。其中主要功能需求有：( 1 )支持新需求的识别，使用户的反馈和环境的变化能够更准确、更及时地反映给开发人员；( 2 )支持系统对新需求与现有系统的分析，例如对实现一个新需求可能产生的影响的分析；( 3 )支持的各种变更操作，尤其是新构件与现有系统的集成，以及过时构件与系统的脱离。

除了这些需求之外，Web 软件的特性还要求软件工具能够支持多样性和组合性，即：( 1 )支持多种开发方法和实现技术的有效混合使用；( 2 )支持多种信息表示标准的混合使用；( 3 )支持较方便地对软件工具进行扩展，使之在新标准、新技术和新方法出现时，能够便捷地将新工具集成于其中并支持其与现有工具的混合使用。

由上述需求可知，软件集成机制在 Web 软件工具中起着十分重要的作用。

分析当前各种软件集成技术，我们可以区分三种软件集成机制：( 1 )静态集成机制通过将各个软件构件的源代码在一起编译和连接形成一个完整的系统；( 2 )动态集成机制将目标代码通过连接形成完整的系统，并在系统运行开始之前完成集成；( 3 )活态集成机制在各个集成成分都在运行状态下通过通讯和交互形成一个协调工作的系统。当前 Web 软件技术正在朝着愈来愈多地采用活态集成机制的方向发展 [ 2 ]。例如，Web Service [ 9 ] 和多 Agent 技术 [ 10 ] 都是活态集成机制的典型例子。针对这类系统的软件环境也应该采用活态集成机制。Agent 技术在 Web 软件中正在得到越来越广泛的应用。多 Agent 系统所提供的方便灵活的活态集成机制十分适合万维网上资源的开放性、动态性和自治性。而且，万维网服务器上运行的软件在文 [ 11 ] 的弱 Agent 概念下均可视为 Agent。我们认为，对以 Agent 技术为主体的 Web 软件的开发、维护和管理也必须应用 Agent 技术。而以其它集成机制构造的 Web 软件，Agent 技术也提供了一种实现满足上述需求的软件工具环境的较好的途径。此外，Agent 技术还具有其它诸多优点，在此不一一细述。

图 2 给出了一个具有如上功能的信息系统支撑工具环境的概念模型 MAISGE。

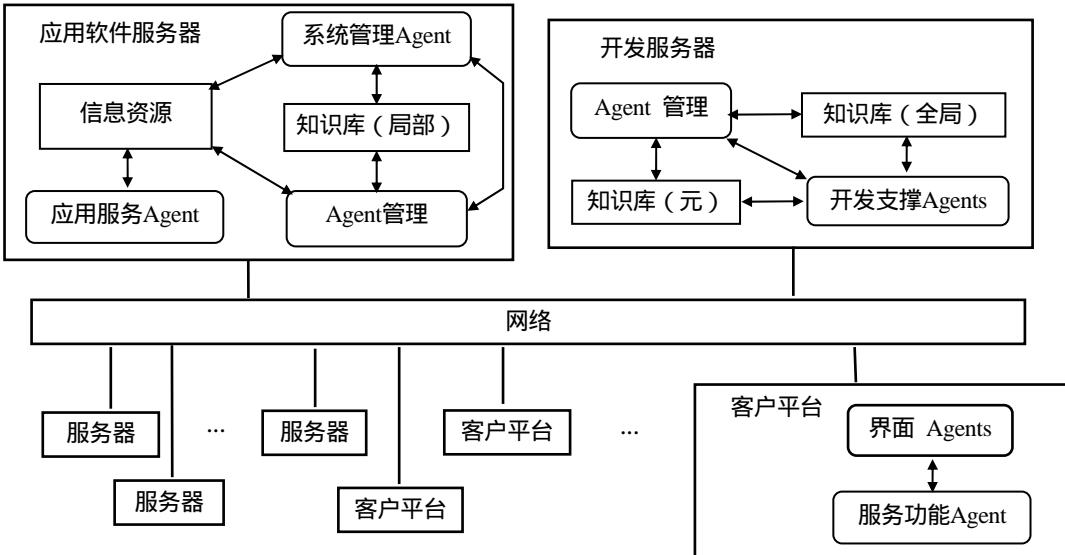


图 2 多 Agent 软件生长环境 MAISGE

MAISGE 模型基于多服务器和多客户平台的网络计算环境，是一个针对 E - 类型分布式信息系统生长式开发的多 Agent 软件系统。应用系统中的软件构件以 Agents 的形式出现，软件构件的加入和删除成为 Agent 的动态加盟和退出。支持软件开发活动的各种工具也以 Agent 的形式出现，并与应用系统中的作为软件构件的 Agents 同时共存，在系统管理 Agents 的协调下，相互协助，共同完成应用系统边开发边使用、稳步成长的过程。因此，整个系统中包括两类 Agents：应用服务 Agents 和开发服务 Agents。

应用服务 Agents 提供对用户的直接的服务并完成系统的各种功能。应用服务 Agent 又可以进一步分为两类，界面 Agents 和服务功能 Agents，前者提供友善的人机交互界面与工具，并担负部分接受用户反馈的责任。后者完成各种系统功能。

开发服务 Agents 提供对软件开发人员的各种开发活动的支持，它们可进一步分为系统管理、开发支持和 Agent 管理三类。系统管理 Agents 负责对应用系统的监视、管理和日常维护，并通过与应用服务 Agents 中的部分界面 Agents 的通信，获得用户的反馈。开发支持 Agents 提供软件开发工具以及对软件开发人员的各种开发活动的服务，除各种较传统的开发工具外，它们还通过对系统知识的采集、维护和使用，帮助或独立完成各种修改影响分析、一致性和相容性分析等工作。Agent 管理 Agents 负责对系统中的各种 Agents 进行管理，其主要职责有三：(1) 注册：在 Agent 加入系统时，对 Agent 的功能、职责、能力、运行环境需求等信息进行注册登记，在 Agent 退出系统时，对 Agent 进行注销；(2) 调度：当开发、维护和使用任务出现时，将任务分配给适当的 Agent 去完成；(3) 记录：并对任务的状态、进展过程、结果以及各个 Agent 的表现进行记录和分析，为将来更有效地进行任务调度积累信息。

MAISGE 模型不同于现有软件开发环境，它具有如下特征：(1) 开发工具与信息系统共存于同一环境之中，从而使生长策略下信息系统在漫长的开发与使用同步进行的生命周期得到全面的支持；(2) 为了支持信息系统中日益增加的智能性服务以及信息系统开发本身所必须具备的智能性，采用分布式数据库和知识库，全面记录并使用有关系统及其软件开发的知识与信息，如 (A) 有关信息系统本身的结构、内容、版本、配置、演化过程等信息；(B) 系统中各个 Agent 的能力、需求和业绩，以及相互之间的关系；(C) 各种软件开发任务和开发过程的知识，如开发任务如何分解为子任务、开发任务之间的逻辑和时序关系，等等。这些通常散布于多个不同种类的软件开发工具的知识与信息被集成于同一系统中，可以提供信息系统稳步成长的良好环境。(3) 采用便于系统扩展的软件体系结构，即多 Agent 系统，使环境系统与信息系统本身都具有良好的可生长性。由于这些特点，我们称该模型为信息系统生长环境。

基于 MAISGE 模型，我们正在以生长策略开发 MAISGE - AquIS 系统，这是一个针对 Web 软件特点的软件生长环境元系统，其中只包含一组开发服务 Agent 和信息系统知识库框架。给定一个 Web 信息系统，该元系统可以通过一组知识抽取 Agent，自动抽取其结构信息并存放于系统知识库中，而从嵌入到生长环境中，成为一个特定系统的生长环境。该系统当前的主要开发重点是 Web 软件的质量管理。在对信息系统的开发、测试和维护中产生的元信息被

存放在知识库中，为自动进行回归测试提供必要的信息。它还对信息系统本身及其环境的变化进行监控，当变化发生时，分析变化的影响，自动产生由此而必须进行的测试工作任务，并由 Agent 管理与调度 Agent 产生测试计划、分配测试工作，由测试助手 Agent 协助测试人员完成质量保障工作。MAISGE - AquIS 的实现参见 [ 7 ]

#### 4. 结语

软件生命周期的生长模型明确地提出了软件生命过程不同的具体表现形态的存在性和可规划性，指出用户需求变化是软件演变的基本动力，提出了生命周期应该与软件的 Lehman 类型等性质相适应。该模型与现有的瀑布模型、螺旋模型、RAD 模型、阶段进化模型等传统软件过程模型相比，较好地反映了软件维护、进化等变更活动在软件生命周期中的作用，由此引出了软件生命周期规划策略的新概念。该模型与目前正在形成的极端程序设计 [ 5 ] 轻快软件开发 [ 6 ] 等软件工程新思维相比，在应付 Web 时代软件开发所面临的困境的策略上具有许多共同的出发点，但生长模型更强调了继承传统软件过程模型的基本原理，而不是完全抛弃现有软件生命周期和过程模型的理论。本文应用生长过程模型的理论，在分析 Web 软件的特点的基础上，提出了适合于 Web 软件生命周期的生长策略，分析了当前软件工具对生长式 Web 软件开发支持的不足之处及其原因，并进一步提出了支持生长策略的软件工具模型 - 软件生长环境。

#### 参考文献

- [1] Crowder, R., Wills, G., Heath, I., and Hall, W. Hypermedia Information Management: a New Paradigm, in Proceedings of 3rd International Conference on Managing Innovation in Manufacture, University of Nottingham, 6-8 July 1998, pp329-334.
- [2] Zhu, H., Greenwood, S., Huo, Q. and Zhang, Y., Towards agent-oriented quality management of information systems, in Workshop Notes of 2nd International Bi-Conference Workshop on Agent-Oriented Information Systems at AAAI'2000, Austin, USA, July 30, 2000, pp57-64.
- [3] Reifer, D., Ten Deadly Risks in Internet and Intranet Software Development, IEEE Software, Vol. 19, No. 2, Mar/Apr 2002, pp12-14.
- [4] Taylor, M. J., McWilliam, J., Forsyth, H. and Wade, S., Methodologies and Website Development: A Survey of Practice, Information and Software Technology, Vol. 44, 2002, pp381-391.
- [5] Beck, K., Extreme Programming Explained, Addison Wesley, 2000.
- [6] Cockburn, A., Agile Software Development, Addison Wesley, 2002.
- [7] 朱鸿，Web 软件的生长式开发过程模型及其支撑工具，电子学报，已录用。
- [8] Lehman , M. M., Laws of Software Evolution Revisited, Proc. Of EWSPT96, Nancy, Oct. 1996.
- [9] Gottschalk, K., Graham, S., Kreger, H., and Snell, J., Introduction to Web services architecture, IBM Systems Journal, Vol. 41, No. 2, 2002, pp170~177.
- [10] Wooldridge, M., An introduction to Multi-Agent Systems, Wiley, 2002
- [11] Wooldridge M. J. and Jennings, N. R., Agent theories, architectures, and languages: a survey in Intelligent Agents: Theories, Architectures, and Languages, LNAI 890, Springer-Verlag, 1995, pp1-32

## Software Growth Model and Web Application Development\*

Zhu Hong

(Dept of Computer Science, National University of Defence Technology, Changsha 410073 )

**Abstract:** This paper proposes a software process model that regards software life-cycle as a process of growth. The model enables a diversity of different types of software processes to be modeled, which include traditional static process and more dynamic evolutionary processes. It is argued that the driving force of software evolution is the changes in users' requirements. The life-cycle of a specific software system can be planned and controlled, but it must be appropriate with regards to its features, such as its Lehman's type. Thus, the notion of life-cycle strategy is introduced. Based on an analysis of Web software's features, the paper argues that most Web applications should be developed in a growth strategy, which is characterized by the positive attitude towards modifications. The problems and weakness of existing software tools for developing Web applications are then analyzed. Finally, the notion of software growth environments is proposed for supporting developing Web applications in the growth strategy.

**Keywords:** Web-based application software; software engineering; software process models; life-cycle strategies; software environments and tools; software agent technology.

---

\* Received: Sept. 16, 2002; Accepted: Oct. 8, 2002.

Partly supported by the National High Technology Development Program of China under Grant No 2002AA116070;