

自适应软件过程的“服务-角色-规则”形式化模型设计^{*}

曹宝香 夏小娜

(曲阜师范大学计算机科学学院 日照 276826)

摘要 对自适应软件过程深入研究,在 Agent 搭建其模型基础上,引入过程资源和过程角色的概念,完成定义与原理论述。并对过程资源和过程角色的自适应调配过程实现,制定出相关的映射规则关系网,实现自适应角色、服务、规则的三位一体化控制,完成自适应的刚柔相济网络体系。

关键词 自适应,服务,角色,规则,代理,软件过程,规则库

Formal Model Desing about the “Service-Role-Rule” of Self-adaptive Software Process

CAO Bao-xiang XIA Xiao-na

(Computer Science College ,Qufu Normal University,Rizhao 276826,China)

Abstract Self-adaptive software process was deeply studied. Based on their models built by agent technology, introduced the definition of process resource and process role, then completed their concepts and principles. For process as well as resources and the role of adaptive process of the deployment process , achieved and formulated relevant rules of the mapping network. Movever, partially achieved the rule library and realization of self-adaptive role, service and rule's whole control, in order to complete self-adaptive network system, which is not tough but flexible.

Keywords Self-adaptive, Service, Role, Rules, Agent, Software process, Rule library

1 引言

软件过程是复杂与动态一体化交叉的过程,它与软件的质量保证、项目的进度和生产率等有直接关系,已成为软件一切人员在工程领域的重要研究方向之一。在第九届国际软件工程会议上,L. J. Osterweil 提出了“软件过程仍然是软件”的论断。

Agent 能够主动地根据所处的环境,自治地实施其行为,具有良好的自适应特性,对软件过程的“刚性”和“柔性”可以较好地自治调整以适应软件过程的变化,并确保在性能测试良好的情况下,极大地实现软件开发的目标。本文首先分析了软件过程的定义,在阐述完 Agent 技术的相关应用后,针对其特性,发挥特长,运用到软件过程的模型构建中,改良传统软件过程模型,引入过程资源与过程角色的概念,统筹提出基于“服务-角色-规则”的自适应软件过程模型理念,在面向对象的软件设计技术基础上,建立起服务、角色自主调配的映射规则关系网,并对过程实现进行了形式化的描述与定义。

2 软件过程的定义及相关的复杂性

图 1 为基于复用技术的软件过程基础框架,是在 CMM 模型驱动的自顶向下的处理过程,融入度量驱动的自下而上的逆向改进序列,在支持复用的基础上,以过程支持层为数据枢轴,由过程组织层的宏观迭代过程和过程运作层的微观动态调整优化构成,二者通过复用组织过程,实现了过程支持层数据的共享与实时处理。通过实时对过程运作进行统筹处理,不断更新过程能力的强度与韧度,发现过程关键目标的自适应改进,启动有效相适应的过程组织层,反复进行下去,达

到持续不紊的柔性过程。

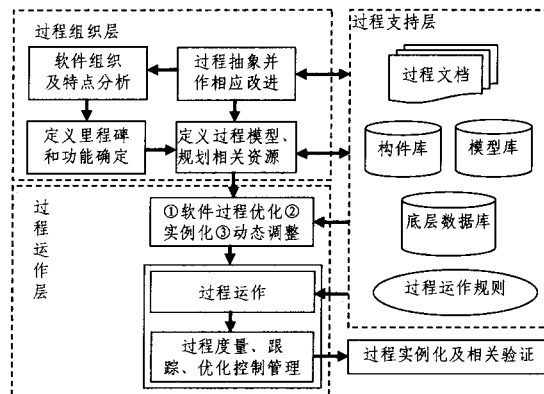


图 1 软件过程基础框架

图 1 粗粒度地给出了软件过程的组织与协调模型,但是对于相关的资源、人员的组织上只是模糊得给出了一个整体概念,至于其中的逻辑规则并未给出一个行之有效的解决方案,比如如何捕捉过程的改进、如何快速建立起处理的相应规则或者映射到相适应的规则库中,需要进一步作底层分析与论证。

软件过程涉及的因素很多,具有明显的不确定性,过程能力和人、环境、技术、资源、规则及用户需求的理解都有着实时的依赖性和相关性。运用静态的、机械的、被动的技术描述软件过程预期有可能发生的情况,又显式地定义解决方案,是很困难的。特别是当所处的环境发生变化,此种描述难以自适应地对这些变更作出实时的调整,只能依赖于物理干预去机械修改,在时间、工效上都是极端不利的,加大了软件成本。

^{*} 本文由国家自然科学基金项目(项目编号:60072014)和省自然科学基金项目(项目编号:Y2003G01)资助。曹宝香 教授,硕士生导师,研究方向为计算机图形学、CAD;夏小娜 助教,硕士研究生,研究方向为管理信息系统、软件工程。

3 代理技术在软件过程中的应用现状

将 Agent 技术引入软件过程的模型化描述日益受到关注,目前主要集中于两种模型的研究:一是 Agent 增强过程模型(Agent-enhanced Process Model),二是基于 Agent 的过程模型(Agent-based Process Model)。

其中,Agent 增强过程模型偏重于代理技术与基于 workflow 处理过程的柔性集成,这套方案的实现技术,是借助于形式化的 workflow 形式。对于管理,则由 Agent 实现控制。在此两项技术结合的过程模型实现了过程和系统的重用,能在保留已有应用过程及系统基础上,通过 Agent 较好地实现了过程的自适应。但它过多保留了传统模型因素,对于自适应的具体实现总具片面性,再就是异种的 Agent 来对过程各个方面实施管理,复杂程度颇高,因此这种技术主要集中于分布式的过程协同中。

基于 Agent 的过程模型借助 Agent 对过程相关元素及相互的联系直接予以描述,从根本上吸取了 Agent 的自适应特性,实现对服务的整合以及任务的分布式管理。ABWFMS 模型引入了 3 种 Agent:过程 Agent、监控 Agent 和服务 Agent,分别完成任务的动态分配和分布式协同,负责本地监控任务的实施、封装任务实现的方法。通过此三类的协同,实现分布条件下动态的工作流整合和服务整合。ABMEs 模型引入两种 Agent,实现过程活动动态整合的活动 Agent,以及封装了活动实现的角色和方法的资源 Agent,它们通过合同网络协议实现协同。但上述的模型并非针对软件过程领域,忽略了很多软件过程中的关键要素,比如过程活动的执行角色、资源分配人员组织的规则建立等。

4 自适应过程中的服务、角色、规则关系

软件过程涉及的要素很多,比如活动、资源、角色、过程部件、半成品、产品等,它们相关了整个自适应过程。当把这些要素进行分时段分性质的划分时,就要在仔细分析它们所处的过程状态与流程里程碑基础上,合理有效地规划到系统剖面的三个层次上。

图 2 中,规则的匹配、角色、服务的调度过程是与中间层密不可分的,规则库的创建很重要,本文把它一分为二,分别为基于服务的规则库和基于角色的规则库,在中间层的业务逻辑处理环境下,实现了规则请求、规则调度等双向过程。中间层一方面实现了服务、角色的适时调配,也实现了服务间的过渡、柔性接口的实现及最终软件过程的柔性集成。

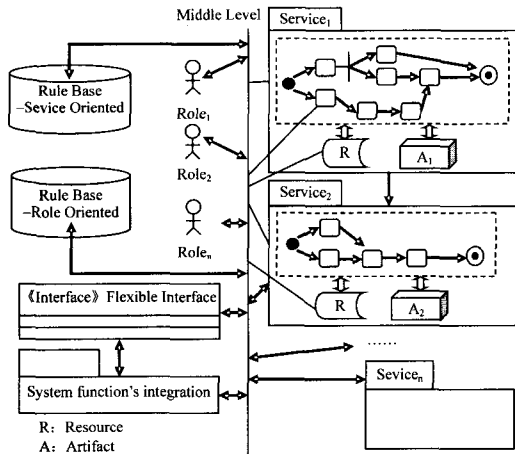


图 2 服务、角色、规则映射关系图

5 基于 Agent 技术的自适应软件过程模型创建

本节在图 2 的基础上,给出规则的识别与建立及中间层的业务处理流程。把图 2 中涉及的关系定义为关联类,各种服务定义为一个过程 Agent 单元,对于内部的资源调配在自身 Agent 的柔性处理基础上,加上中间层与规则库的适时关系捕捉与自适应处理过程,达到度量的统一与到位。对于服务的过程 Agent 单元间的关系建立与拆分同样依赖于中间层的业务中介处理,因此在服务内服务间的过程中间层的作用至关重要,这里采用增强 Agent 过程模型,实现整个系统的网络化多层化管理。图 3 是用 UML 抽象类表示的关系图,每个类对应相应的 Agent 类型。

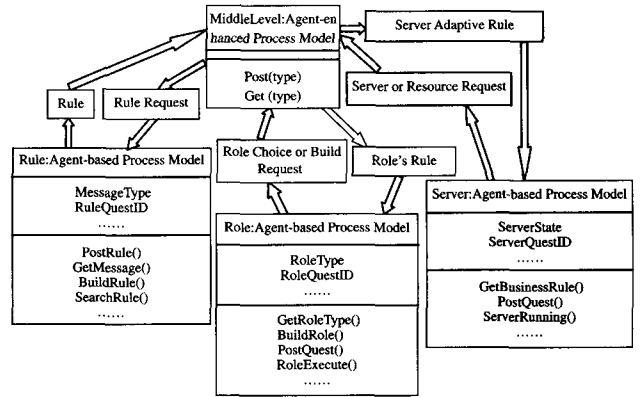


图 3 基于 Agent 的服务、角色、规则调配关系图

图 3 中,服务、角色、规则通过相应的 Post, Get 处理完成请求的提交、规则的选择与传递。三者均与中间层发生双向的业务过程。由于分别定义成不同的 Agent 过程模型,不管是局部还是整体,在模型上实现了数据的双向自适应过程,使系统模型整体上是透明的业务自治模式,并完成了局部的柔性过渡与适时调节,是合理的,加上抽象类的定位方式,从根本上保证了模型的健壮性和刚性。

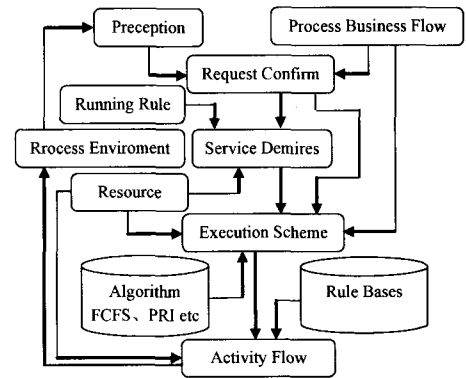


图 4 Agent 的自适应底层逻辑处理模型

为了进一步说明基于 Agent 技术模型的自适应处理的优越性,图 4 给出 Agent 对于信息、数据的自适应底层逻辑模型。首先,过程 Agent 在对环境的感知及其过程业务处理流程的前提下,确立请求信息及所需资源规则等情况,从请求中确定一组能够实现的目标生成其愿景。然后,过程 Agent 在其请求、过程运作规则以及资源情况基础上,通过逆向推理过程针对其愿景访问规则库,并按照“先来先服务+优先级高低”算法,形成相应的规则请求序列,映射出本次请求的服务活动流,过程 Agent 选择其当前的活动予以实施,同时并行地

完成特定角色的定位与选取,确保活动的有效实施。与此同时,过程 Agent 模型随时感知新的反馈与请求,借助于增强 Agent 模型,建立起彼此间的无缝关联。

6 规则库的形式化逻辑定义与底层支持

自适应的软件过程就是由一组过程 Agent 组成,并借助于增强型 Agent 技术实现了过程 Agent 间的交互。这些 Agent 在特定的环境、适时的阶段中能够主动自治地实施行为。

定义 1 自适应软件过程(SASP; Self-adaptive Software Process)形式上是一个四元组: $SASP = (PES, PM, PA, AEP)$,其中每个元素的含义如下:

(1) PES 表示过程环境状态(PES; Process Environment State), $PES = \{PES_1, PES_2, \dots, PES_n\}$,括号内为可能的过程状态选择项。

(2) PM 是过程里程碑(PM; Process Millstone),可以有多个。 $PM = \{PM_1, PM_2, \dots, PM_n\}$ 。每个过程里程碑又是一个四元组 $PM_i = (PMS, PMSeC, PMRoC, PMRuC)$,其中:

PMS 是一组过程里程碑描述(PMS; Process Millstone Statement)。 $PMS = \{PMS_1, PMS_2, \dots, PMS_n\}$ 。

PMSeC 是与过程里程碑相关的服务约束序列(PMSeC; Process Millstone Service Constraint)。 $PMSeC = \{PMSeC_1, PMSeC_2, \dots, PMSeC_n\}$,每项还涉及资源的定义与参与。

PMRo 是过程里程碑相关角色约束序列(PMRoC; Process Millstone Role Constraint)。 $PMRoC = \{PMRoC_1, PMRoC_2, \dots, PMRoC_n\}$ 。

PMRuC 是过程里程碑相关的规则序列。它是个三元组(PMRuC; Process Millstone Rule Constraint)。 $PMRuC = (PMRuCS, PMRuC-Se, PMRuC-Ro)$,每项都是一个序列,PMRuCS 表征规则变化选择的状态序列,PMRuC-Se 表征服务运作的规则序列,PMRuC-Ro 表征角色选择创建的规则序列。不再详述。

(3) PA 是相互独立而对等的过程 Agent 组合(PA; Process Agent)。 $PA = \{PA_1, PA_2, \dots, PA_n\}$ 。每个 PA_i 将单独定义。

(4) AEP 是整个系统集成的一体化增强型 Agent 过程(AEP; Agent Enhanced Process),它是其他元素的递归定义,是三元组。 $AEP = (PES, PM, PA)$ 。

定义 2 PA_i 定义为一个五元组 $PA_i = (PAS_i, PARo_i, PARC_i, PAA_i, PAAF_i)$ 。其中:

(1) PAS_i 是相对于该过程 Agent 的局部过程环境状态(PAS_i ; Process Agent State),状态间存在着相互转换的边界哨。

(2) $PARo_i$ 是过程 Agent 的对应角色的描述,描述它参与该过程需满足的条件。

(3) $PARC_i$ 是过程 Agent 相关的资源约束,描述了 Agent 所拥有的资源性质和数量。是二元组($PARC_i$; Process Agent Resource Constraint)。 $PARC_i = (Character_i, Quantity_i)$ 。

(4) PAA_i 是一组过程 Agent 能够实施的活动序列(PAA_i ; Process Agent Activities), $PAA_i = \{PAA_{i1}, PAA_{i2}, \dots, PAA_{in}\}$ 。其中一个活动就是一个四元组定义 $PAA_{ij} = (CR, ARo, ARC, AM)$ 。其中:

CR 是活动的控制规则,描述了如何控制该活动实施的

整个过程(CR; Control Rule)。

ARo 是活动对应角色的描述,描述了角色参与该活动应具备的条件(ARo ; Activity Roles)。

ARC 是活动过程所需的资源约束,它描述了该活动实施过程中所需要的资源(ARC ; Activity Resource Constraint)。

AM 是活动的实现流程方法,它的过程依赖于人工的机械调配和软件过程的自适应调节与配给(AM ; Activity Method)。

(5) $PAAF_i$ 描述了过程流程相关的步骤序列($PAAF_i$; Process Agent Activity Flow)。 $PAAF_i = \{PAAF_{i1}, PAAF_{i2}, \dots, PAAF_{in}\}$ 。每个步骤是一个三元组递归定义。 $PAAF_{ij} = (CR, ARC, AM)$ 。

定义 1 和定义 2 给出了自适应软件过程模型的形式化逻辑定义。在该定义中,过程要素之间的关系并不是被显式地描述。借助于里程碑、过程流序列描述了在过程要素建立关系的自适应规则实现,而关系则是在实现过程中,针对特定里程碑和过程环境动态建立的,最终,通过增强型 Agent 过程实现了系统整体性能的统一,这也是上文中模型图所表达的思想。

结束语 本文把软件过程归结为服务、角色、规则的统一,为实现一体化的自适应过程,形式化地给出了基于 Agent 的模型,目的在于提供一个可以论证计算的模型来描述这种自适应过程的定义和实现,并给出了软件过程描述的形式化定义,通过这种描述,开发人员可以较方便地运用面向对象的建模技术建立过程模型。

下步工作将继续完善模型,开发出基于此模型的轻量级原型处理系统,体现出服务、角色、规则三者的柔性合作过程。并在过程建模的更加形象化方面继续探讨,与面向对象技术结合,提供一种简单、直观有效的模型,达到可视性、易证化的柔性统一。

参考文献

- [1] 赵欣培,李明树,王青,等.一种基于 Agent 的自适应软件过程模型[J].软件学报,2004,3(15)
- [2] 陈迎欣,刘群.基于递阶的软件过程模型重用方法[J].武汉大学学报(理学版),2005,82(51)
- [3] 陆勤,吴永明.一种基于 SC 的自适应软件设计方法[J].计算机应用与软件,2004,1
- [4] 郭成昊,赵严厉,王纪文,等.自适应系统的体系结构模型与分析[J].计算机工程.2006,19(32)
- [5] 倪晓峰.自适应型动态软件构架模型的研究与设计[D].复旦大学,2005
- [6] 姚松涛.自适应软件系统集成体系结构的研究[D].华南理工大学,2004
- [7] 徐如志,都艺兵,于华,等.基于复用的软件过程改进方法[J].计算机科学,2006,6(33)
- [8] 邹晓宇,王戟,周俊鹏.基于工作流的软件过程开发平台的研究[J].计算机应用研究,2006
- [9] 张友生,陈松乔.基于体系结构的软件过程 Petri 网模型[J].小型微型计算机系统,2005,1(26)
- [10] 张屿,李彤.一个基于构件的软件过程控制模型[J].计算机应用研究,2005,3