第33卷 第16期 计 算 机 工 程 2007年8月
Vol.33 No.16 Computer Engineering August 2007
·博士论文· 文章编号：1000—3428(2007)16—0017—03 文献标识码：A 中图分类号：TP311

基于求精的软件体系结构设计方法

赵 恒[1,2]，叶俊民[3]，王振宇[2]
(1. 哈尔滨工程大学计算机科学与技术学院，哈尔滨 150001；2. 武汉数字工程研究所，武汉 430074；
3. 华中师范大学计算机科学系，武汉 430079)

摘 要：分析了当前软件体系结构研究对软件开发支持存在的不足，将软件体系结构求精技术引入大型复杂系统软件体系结构设计与开发过程中，结合抽象层次和层次视图，提出了一个基于求精的软件体系结构设计方法——ARSADM，给出了其关键步骤和过程，用于指导软件体系结构的正确设计。

关键词：软件体系结构；软件体系结构求精；软件过程

Refinement-based Software Architecture Design Method

ZHAO Heng[1,2], YE Jun-min[3], WANG Zhen-yu[2]
(1. College of Computer Science and Technology, Harbin Engineering University, Harbin 150001;
2. Wuhan Digital Engineering Institute, Wuhan 430074; 3. Department of Computer Science, Central China Normal University, Wuhan 430079)

【Abstract】 On the basis of reviewing current researches on software architecture supporting for software development and analyzing their inadequacies, this paper advances a new refinement-based software architecture design method(ARSADM). The method combining architectural hiberarchy with views applies the software refinement technica in the process of software architecture design in order to guide the correct software architecture design.

【Key words】 software architecture (SA); software architecture refinement (SAR); software process

1 概述

软件体系结构是软件开发方法学和模型的综合，是一种控制软件性的有效方法和技术的集合体。其研究和实践旨在将一个系统的高层软件结构显式化，以在高抽象层处理诸如全局组织和控制结构、功能到计算元素的分配、计算元素间的高层交互等设计问题，从而达到降低软件开发成本、增加相关系列产品中不同成员之间共性的潜力、控制软件复杂性、提高软件重用度的目的[1]。它在软件需求与软件设计之间架起一座桥梁，着重解决软件需求向实现平滑过渡的问题。

分析目前软件体系结构的研究，可以发现目前的研究大致可分为两类：一类侧重于软件体系结构形式化理论研究；另一类则将软件体系结构的设计、描述和表示同软件系统建模相结合，旨在建立软件需求与软件设计之间的平滑过渡。前者着重考虑体系结构强大的分析能力，而后者则强调体系结构对系统实现的直接支持。目前大多数的软件体系结构研究集中于前者，侧重于对体系结构的描述和高层性质的验证上，对后者的研究相对薄弱，对软件体系结构的求精和实现的支持能力明显不足[1]，还没有出现成熟的方法和技术，从而在一定程度上限制了研究成果的应用。软件开发实际上是一个从问题域向解决方案域逐步映射和转换的过程，其最终目的是获得高效的可执行系统。基于软件体系结构的复杂软件系统开发方法强调通过建立构件以及构件之间的相互关系来对问题域系统进行建模，通过对SA模型的分析、设计、实现与维护完成整个软件生命周期。因此，SA的设计质量直接影响着整个系统的软件质量。目前人们对基于SA的软件开发方法还没有明确统一的认识。在大多数的工程实践中，体系结构设计是建立在直觉和经验之上的，随机性很大，缺乏严格的理论基础和工程原则，影响了SA的设计质量，妨碍了体系作用结构在软件开发过程中的指导作用。因此，有必要为SA的设计建立一个良好的软件过程。

2 软件体系结构求精

2.1 基本概念

软件体系结构求精概念(software architecture refinement, SAR)是随着SA研究的发展而提出的，它是SA研究的重要研究内容。SAR最早出现于20世纪90年代[2]，是程序求精思想在SA领域的应用和发展。

大规模复杂软件系统的开发是一项复杂而艰巨的任务，是不可能一蹴而就的。为让问题变得简单和易于控制，体系结构师通常需要采用层次化、分步的方法，在不同的抽象层次对复杂系统的体系结构进行建模。高层抽象的体系结构能够简化系统整体分析、帮助开发人员理解系统和相互之间的沟通和交流，而低层具体的体系结构因为提供更多信息，能够进行更加严格的检测和分析，或者能够进行仿真执行和指导生成系统实现代码。

一个体系结构设计最初可能是一高度抽象的基于构件－连接件的视图，对于这种抽象视图，必须进一步求精为更具体的视图，具体视图比抽象视图包含更多的需求和设计信息，如构件接口说明、实现策略等。这样，才可以逐步转化一个可以用编程语言实现的模块视图。

基金项目：国防预研基金资助项目
作者简介：赵 恒(1966－)，女，博士研究生，主研方向：软件体系结构，软件工程；叶俊民，博士；王振宇，研究员、博士生导师
收稿日期：2006-08-20 **E-mail：**hellezhao@tom.com

文献[3]指出，系统的实现实质上就是一种体系结构的求精。运用求精技术，能够给出体系结构的实现方案，支持在各种不同的抽象级别对系统进行抽象和思考，能够方便大规模、复杂问题的体系结构建模，帮助推广体系结构形式化建模技术的应用。

所谓的软件体系结构求精是指上层抽象体系结构与下层具体体系结构之间存在一种形式上的抽象映射，下层通过这种映射应保留上层的属性。所以，求精实质上是一种代换技术，用于将软件系统的上层“抽象模型”(其规范）变换为另一种更具体的下层“数学模型”(其求精结果)。下层模型在以下两方面比前者更具体：(1)它可能包含了与原来模型有关的更多细节；(2)它可能更接近于实现。

2.2 程序与求精的比较

SAR与早期的程序逐步求精不仅在实现层次级别上不同，而且在求精方法上也不同。早期的程序逐步求精起源于Dijkstra 和Hoare的程序证明工作。程序求精方法大多是采用谓词逻辑和证明义务来说明系统的规格说明，并进行逻辑推理而得到程序的。这种方法侧重于程序的推导，主要缺点是没有满足非功能属性设计的概念，规格说明必须详细到可推理，因而对于大型系统往往显得力不从心。而SAR强调的是实际实现与其规格说明的一致性。一般采用行为替换(behavioural substitutability)和守恒扩展(conservation extension)方法。因此，SAR是“需求工程与体系结构建模相互结合、相互依赖的开发活动，以保证需求（功能需求和非功能需求）与系统实现的完整性和一致性”[3]。SAR重点及难点在于如何保证不同抽象体系结构层之间求精过程的语义一致性，以及对体系结构非功能特性的分析和验证。

3 基于求精的体系结构设计方法

3.1 基于软件体系结构的软件过程

软件过程又称为软件生命周期过程，是在软件生命周期内为达到一定目标而必须实施的一系列相关过程的集合。它定义了用于创建复杂软件系统的方法论。软件过程必须科学、合理，才能开发出高质量的软件产品。软件体系结构是软件生命周期中的重要产物，它影响到软件开发的各个阶段。因此，软件过程和软件体系结构成为现代软件开发中最为关注的两大问题。

文献[1]提出了 ABC(architecture based component composition)方法，即将 SA 与 CBSD 相结合，以 SA 模型作为系统蓝图指导系统的开发全过程，把分布式构件技术作为构件组装的实现框架和运行时的支撑，使用工具支持的映射规则缩小设计和实现间的距离，自动地组装验证所需要的系统。ABC 方法的根本思想是在构件组装的基础上，使用 SA 的理论与概念来指导软件开发，以提高系统生成的效率和可靠性。

文献[4]针对大型复杂系统的开发，提出了基于交互的复杂软件系统合成演化方法(ICE-CSS)，强调了通过建立部件之间的相互联系结构来描述它们之间的分类交互(interaction)、促进它们的显式合成(composition)和实现复杂软件系统的演化(evolution)设计过程。

上述方法都为大型复杂系统基于 SA 的软件开发提供了方法学上的指导。但二者的缺点是对如何正确设计一个大型复杂系统的软件体系结构的设计方法与过程没有予以说明。

本文在上述方法的基础上，将软件体系结构求精技术引入大型复杂系统软件体系结构设计与开发过程中，结合抽象层次和层次视图两方面，提出了一个基于求精的软件体系结构设计方法(architectural refinement-based software architecture design method, ARSADM)。

3.2 ARSADM 的基本思想

ARSADM 是以体系结构为核心、采用迭代的方法从软件需求导出正确体系结构设计所要经历的阶段和步骤的集合。这里所谓的正确体系结构设计是指将系统的体系结构建模为一组由连接件连接的构件集，它们在一定的约束条件下，能正确地映射系统需求。ARSADM 方法的基本思想是把整个 SA 设计过程划分为不同的抽象层次：需求规格说明层，抽象 SA 层，具体 SA 层和对象层。每个层次都支持特定的功能需求和非功能需求。为了实现每一层次的平滑过渡，每一抽象层从结构、行为和交互 3 个方面建立相应的层次视图。层次视图从不同侧面反映系统特性；用 SAR 的方法对每个层次视图进行形式化求精，从而实现不同抽象层次间的平滑迁移，通过采用形式方法来控制整个迁移过程，以提高设计质量。

ARSADM 体现了迭代的软件开发思想，可以比较好地解决沟通困难、需求变更等问题，同时保证 SA 设计演化的正确性。图 1 给出了抽象层与视图之间的关系。

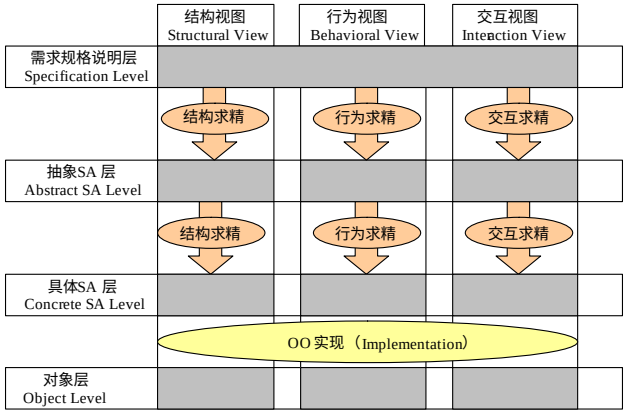

图 1 抽象层与视图之间的关系

(1)需求规格说明层：用软件术语描述问题空间的需求，定义其基本词汇、领域信息和应用需求。该层模型的主要特点是领域相关性，即描述的是领域特定的词汇，使用用户能理解的语言表达系统需求。其模型与目标应用没有直接的关系，具有领域通用性。由于在需求层描述的是问题域系统的概念层，因此这一层采用统一视图。

(2)抽象 SA 层：由系统规格说明明确定义系统与环境之间的界限，并由此确定系统的外部和内部服务。用一组可组合的构件及其接口来表示系统。从系统的结构、构件行为和交互行为 3 方面建立相应的抽象视图。

(3)具体 SA 层：通过对系统层抽象的结构、行为和交互视图的细化，将系统层模型逐步求精为包含更多设计信息的构件－连接件模型。由于构件粒度多种多样，因此对于大粒度构件需要继续从结构、行为和交互进行细化，直至可用 OO 方法予以实现。

(4)对象层：用一组相关对象实现构件及构件之间的相互关系。

3.3 实施步骤

下面给出 ARSADM 用到一些基本概念表示。

定义 1 系统目标(goal)：客户对所希望的系统提出的要求，包括功能目标和非功能目标。功能目标(Gf)是指待开发系统应提供的服务功能；非功能目标(Gnf)是指对服务功能的质量要求(QoS)、开发目的和体系结构约束等。

定义 2　代理(Agent)：待开发系统中所涉及的各种角色，如用户、设备、已有软件系统等。

定义 3　系统(System)：待开发的系统及其相关环境。

定义 4　体系结构模型(A-Model)：对系统体系结构的形式化描述。其基本元素包括构件(Component)、连接件(Connector)和约束(Constrains)等。

定义 5　体系结构求精(A-Refine)：对 A-Model 中基本元素的增加或限制以使 A-Model 能更大限度地满足系统目标。

A-Refine过程实质上对A-Model的一个细化过程。基本方法有两种：水平求精和垂直求精。水平求精是在同一抽象层上对A-Model$_i$的细化，使之减少不确定性。其主要目的是识别A-Model$_i$中需要保持的系统目标。垂直求精是从抽象A-Mode$_{li}$到具体A-Model$_{i+1}$的转换，即在不同抽象之间，通过对高层抽象模型进行精化而得到具体模型的过程。其目的是使A-Model$_i$更接近于实现。

ARSADM 过程示意图如图 2 所示。

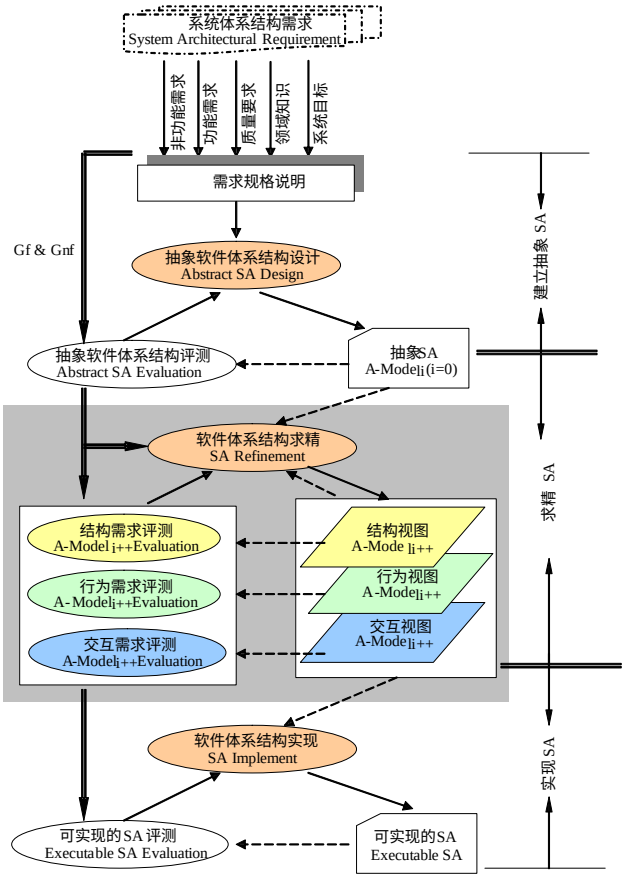

图 2　ARSADM 过程示意图

在图 2 中，开始体系结构设计之前的一个预备工作是根据软件需求获取软件规格说明，它是进行体系结构设计的基础。根据软件需求规格说明进行抽象SA设计，考虑系统的总体结构和基本构成成分。这一步完成SA设计的基础性工作。抽象SA利用抽象性体现系统的关键特性，但不包含任何实现细节。接下来对抽象SA进行求精，把一个抽象的SA逐步求精为一个可实现的SA。该过程是一个增量迭代过程，根据系统的复杂性，可能需要多个中间层A-Model$_i$，抽象与具体是相对的。

在ARSADM中，每个A-Model$_{i+1}$都是由A-Model$_i$从结构、行为和交互 3 方面的求精与验证而生成，从而保证了A-Model$_{i+1}$的正确性。

当一个体系结构求精到具有足够的信息以支持实现时，称该体系结构设计是可实现的。这通常不存在一个统一的标准，它取决于程序员的开发水平、所使用的工具等多种因素。在 ARSADM 中，所关注的主要如何运用求精技术把抽象的 SA 精化为一个能正确体现需求的、更为具体的 SA，用于指导复杂系统软件体系结构的设计。

4　抽象 SA 模型的求精

本节主要对 ARSADM 中的核心过程－抽象 SA 模型求精的方法进行说明。

抽象SA模型的定义提供了进行体系结构求精的空间。体系结构的求精是以A-model$_i$为输入，运用求精规则，迭代输出下一个A-model$_{i+1}$的过程。为此必须解决两个基本问题：(1)对SA的形式化精确描述；(2)提供相应的机制和规则保证低层具体的体系结构能够实现高层抽象体系结构的功能，并遵循它规定的约束和其所有属性。这也是ARSADM的核心。对于基本问题(1)，ADL提供了有力的工具，因此，基本问题(2)是本文研究的重点。

文献[5]分析了需求与求精之间的密切关系，指出需求工程与体系结构建模是密切相关、相互依赖的，因此，诸如体系结构求精等体系结构的相关处理活动必须保持需求的完整性和一致性。因此，在ARSADM中SAR的求精规则的定义取决于设计目标。依据A-model的不同的抽象层次，对Gn和Gnf也进行分层处理，针对相应的抽象层建立相应的Gn$_i$和Gnf$_i$。

具体求精方法是：

(1)基准状态的建立：在对A-model$_i$求精之前，确定A-model$_i$的基准状态，这是求精的起点。在基准状态中必须明确A-model$_i$的所有体系结构需求及约束（即Gn$_i$和Gnf$_i$），它们既是确定求精规则的基础，又是评测A-model$_{i+1}$的标准。

具体分为以下步骤：

1)确定本层需要满足的体系结构约束，并对之精确定义。在上一抽象层中，这些需求约束可能仅说明了“做什么”或“怎么样”等问题，在这一层中需要将它们准确地定义为“在哪做、做些什么”或“有多快、多准确”等；

2)将每个 Agent 映射为一个独立的子构件，构件的接口由 Agent 接口所定义的变量和操作集合变换形成；

3)对每一对构件 C1 和 C2 之间定义一个通道连接件用于管理两个构件的数据流和控制流；

4)从构件结构、行为和交互关系 3 个方面建立系统的结构视图、行为视图和交互视图；

5)确定视图与约束之间的关系，指派视图中每个构件、连接件的设计与实现约束；

6)测评视图的正确性，判断能否正确反映本层体系结构约束。

(2)对A-model$_i$进行求精，生成新的A-model$_{i+1}$。

1)根据A-model$_i$的结构视图，对A-model$_i$中的抽象构件进行分解求精，确定A-model$_{i+1}$的合适组成构件。

2)根据A-model$_i$抽象构件的行为视图，定义A-model$_{i+1}$构件之间的连接关系，形成A-model$_{i+1}$的交互视图，通过构件组合生成新的A-model$_{i+1}$的体系结构。

3)对A-model$_{i+1}$进行评测与证明：根据Gn和Gnf对A-model$_{i+1}$进行模型检查，证明A-model$_{i+1}$与A-model$_i$之间的一致性。测评的标准有：构件的完整性和正确性，构件接

(下转第 22 页)