基于移动嵌入式系统硬/软件协同设计的 EHSC 算法*)

张江陵[1] 郑世珏[1] 胡金柱[2]

(华中科技大学计算机科学与技术学院 武汉 430074)[1] (华中师范大学计算机科学系 武汉 430079)[2]

摘 要 嵌入式系统设计的一个重要任务就是寻找硬/软件最佳搭配方案。随着系统复杂性的不断提高,采用嵌入式系统硬/软件协同设计是提高设计质量的有效手段。本文在讨论嵌入式系统设计一般方法的基础上,阐述了系统的硬/软件协同设计技术和硬件/软件划分方法,提出了以系统硬/软件划分策略为基础,系统组件的权重值为参考,组成元素划分为依据的设计理念,构造了基于移动环境的系统的硬/软件协同设计的 EHSC(Embedded Hardware/Software Codesign)模型。并依照此模型,实现了一种移动嵌入式系统"电子书包"阅读器的设计和开发。

关键词 嵌入式系统,硬件/软件协同设计,EHSC 算法

The EHSC Algorithm Based on Mobile Embedded System

ZHANG Jiang-Ling[1] ZHENG Shi-Jue[1] HU Jin-Zhu[2]

(School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074)[1]

(Department of Computer Science ,Central China Normal University, Wuhan 430074)[2]

Abstract An important task of embedded system design is to seek an optimal hardware/software codesign. With the growing complexity of the systems, embedded system hardware/software codesign becomes an effective way to improve design quality. On the discussion of general methods used in embedded system designing, this paper illustrates hardware/software codesign technology and their partitioning, proposes a design idea based on hardware/software partitioning, the weight of components and their partitioning, which leads to the construction of EHSC. Based on the module, a mobile embedded system-"Electronic Satchelis" developed.

Keywords Embedded system, Hardware/software codesign, EHSC algorithm

1 引言

嵌入式系统的多样性和复杂性,使硬件软件的功能划分、资源调度与分配、系统优化、系统综合、模拟仿真等方面存在许多尚待研究解决的问题[1]。传统的嵌入式系统设计方法将硬件和软件分为两个独立的部分,由硬件工程师和软件工程师按照拟定的设计流程分别完成。这种设计方法只能改善硬件或软件各自的性能,独立的设计空间不可能对系统做出较好的性能综合优化。基于部件功能的协调设计,实际应用中比较难掌握设计尺度,容易产生两种不利结果,即:所谓 Over Design(最终目标平台的某些性能指标超过系统设计目标的相应要求,导致硬件投入较高而性能/价格比较低);Under Design(最终目标平台的某些性能指标不能满足规格说明的要求)。从理论上来说,每一个应用系统,都存在一个适合于该系统的硬件、软件功能的最佳组合。如何从应用系统需求出发,依据一定的指导原则和分配算法对硬/软件功能进行分析及合理的划分,从而使系统的整体性能、运行时间、电耗、存储容量…达到最佳状态,已成为硬/软件协同设计(Hardware/Software Codesign)[2]重要研究内容之一。

嵌入式系统的硬/软件协同设计问题实际上是硬/软件划分问题在约束条件下的优化问题,如在满足系统执行速度及吞吐率等约束条件下,使系统目标实现的物理尺寸最小。本文在讨论嵌入式系统设计一般方法的基础上,阐述了构造基于移动环境的系统的硬/软件协同设计的 EHSC 模型。并依照此模型,实现了一种移动嵌入式系统"电子书包"阅读器的设计和开发。

2 嵌入式系统的体系结构

嵌入式系统体系结构或采用的算法和调度策略都与通用计算机系统相似,主要分为硬件和软件两个部分。系统硬件包括微处理器、SCO(System On Chip)、ASIC (Application Specific Integrated Circuits)、FPGA(Field Programmable Gate Arrays)、SOM(System On Module)、A /D(Analog/ Digital)、D/A(Digital / Analog)转换器、辅助功能芯片和外部设备。嵌入式系统软件包括操作系统和应用程序。

系统组成元素主要有五个部分:① 嵌入式处理器。MCU、MPU、DSP、SOC、ASIC 等;② 其它部件。协处理芯片、存储器、输入输出部件、网络接口和电源等;③嵌入式编程软件。C/C++、Visual C++、GNU C++、嵌入式 Java、Ada、Modula-2 等; ④ 嵌入式操作系统。VRTX、PSOS、VXWORK、WINODWS CE、EPOC、LINUX、μc/OS、POWER PC 等;⑤ 调试工具。实时在线仿真器(ICE)、逻辑分析仪、ROM 仿真器、源程序模拟器等。

目前嵌入式系统的三层结构(图 1)逐步演化成为一种四层结构。这个新增加的中间层次位于操作系统和硬件之间,包含了系统中与硬件相关的大部分功能。

*)本文得到国家自然科学基金资助(60174043)。张江陵 教授,博士生导师,主要研究方向为计算机系统结构与海量存储技术。郑世珏 博士研究生,主要研究方向为计算机系统结构。胡金柱 教授,博士生导师,主要研究方向为软件工程。

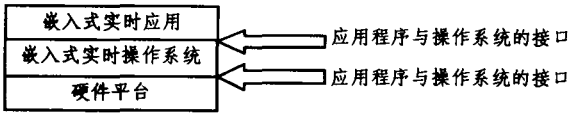

图 1　嵌入式系统结构的基本结构

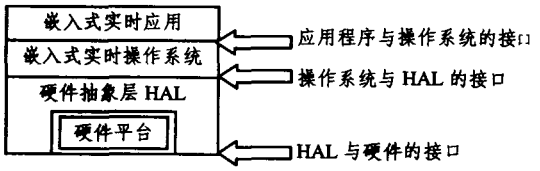

图 2　引入 HAL 以后的嵌入式系统结构

由于引入了一个中层次，屏蔽了底层硬件的多样性，操作系统不再直面对具体的硬件环境。因此，这个中间层次叫做硬件抽象层 HAL(Hardware Abstraction Layer)。这种统一的接口形式得到广泛的认可和应用，是形成板级支持包 BSP (Board Support Package)的原因。目前在嵌入式领域通常把 HAL 称为 BSP。图 2 显示了引入 HAL 以后的嵌入式系统结构。BSP 的引入大大推动了嵌入式实时操作系统的通用化，但却使采用传统方法对硬/软件划分更加模糊。

3　硬/软件协同设计方法

一种典型的硬/软件协同设计过程如图 3 所示。首先，应用独立于任何硬件和软件的功能性规格方法对系统进行描述，采用的方法包括有限态自动机(FSM)、统一化的规格语言(CSP, HDLs, C…)或其它基于图形的表示工具，其作用是对硬/软件统一表示，便于功能的划分和综合。

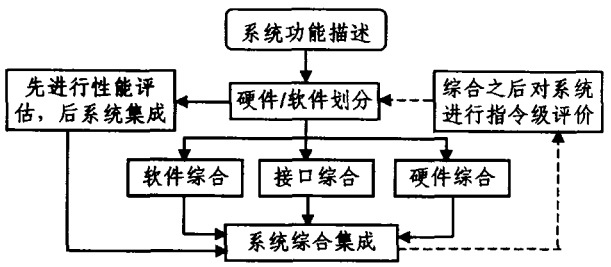

图 3　一种硬件/软件协同设计过程

硬件/软件划分的实质就是对硬件、软件功能模块的分配。这种功能分配从系统功能要求和限制条件出发，依据算法进行。完成硬/软件功能划分之后，要对划分结果做出评估。评估方法之一是性能评估，另一种方法是硬件、软件综合之后对系统指令级评价参数做出评估。

系统硬件/软件划分协同设计与传统设计相比有两个显著的区别：第一，描述硬件和软件使用统一的表示形式；第二，硬件软件划分可以选择多种方案，直到满足要求。显然，这种设计方法对于具体的应用系统而言，容易得到综合性能指标的最佳解决方案。下面对已有的协同设计方法进行简述。

3.1　基于图形的方法[3]

该方法采用有向图(数据流图)、Petri 网统一描述软件和硬件。在数据流图中，用节点表示软件操作或硬件功能单元。ADAS(Architecture Design and Assessment System)使用软件图和硬件图来描述软件和硬件，而软件对硬件的操作采用有限制条件的软件图表示。在 ADAS 中，软件对硬件资源的竞争由用户指定的优先权解决，这与操作系统的处理过程类似，软件使用“准备好”、“阻止”等若干状态竞争硬件资源。

ADAS 使用随机 Petri 网分析确定各种性能的度量标准，如硬件利用率。仿真具有粗粒度和功能仿真两种。粗粒度仿真不考虑节点功能度的细节或标记(token)内容，主要做性能分析，而功能仿真时，其软件操作与“C”语言描述对应。此时，软件图中传输的 token 内容为实际数值的数据结构，因此，这种仿真主要用于检验功能度。一旦确定设计的功能度和性能，将产生硬件的 VHDL(Very High Speed Integrated Circuit Hardware Description Language)。

3.2　基于 VHDL 的方法

该方法利用 VHDL 特性，将系统功能描述成 VHDL 相互影响的并发进程，系统功能被映射到单独的物理单元，其次是设计软件进程。逐步分解操作以及对每种系统功能进行评估是这种方法的主要内容。协同设计过程采用 VHDL 的显著优点是比其它硬件描述语言更加方便。已有实例证明，建立一个计算机体系结构的可执行模型与实现属性无关[4]，相反，VHDL 描述软件的方法可以很方便地转换成要研究的软件开发内容。

3.3　有限态自动机(FSM)方法

FSM (Finite State Machine)可以表示系统行为、硬件模型和软件模型。其通信机制适合描述具有反馈特性的实时系统。FSM 的数学理论可用于正确性形式验证、仿真、硬件中软件的划分和综合，除了形式方法学以外，这种方法的另一个特点就是从规格说明到具体实现的转换过程非常规范。由于状态空间有限，这种方法仅适合小规模嵌入式控制系统。

3.4　基于遗传算法的划分方法[5]

采用遗传操作非重叠的遗传算法，可以从解空间编码、适应函数及个体选择策略、交叉、变异策略、群体局部优化等几个方面对划分算法问题进行分析，实现嵌入式系统的硬件/软件划分。该算法首先确定用于遗传搜索及群体局部优化所需的参数，然后随机生成用于遗传操作满足约束条件的初始解。每次遗传操作前，均采用 SA 算法(模拟退火算法)对其进行处理，获得用于遗传操作的局优解集，遗传操作在局优解基础上进行。利用遗传算法解决硬件/软件划分问题的一个重要方面是确定评价各预选硬件/软件方案性能的适应函数，因为适应函数体现了个体在遗传搜索过程中的繁殖能力，是选择操作的依据。

4　EHSC 模型

虽然系统硬/软件协同设计采用了不同的数学模型，但总体来讲，主要是对设计方法的宏观定性分析。针对一类移动嵌入式系统的专门设计而言，这些方法还可以进一步抽象和简化。本文依据上面讨论的研究方法和技术路线，阐述 EHSC 模型的设计方法。

根据对移动嵌入式系统产品的文献描述和市场调查，本文认为制造方、经销商和用户三方最关心产品的因素依次为价格、功能、体积(可含重量因素)与速度(因毫秒级的运算速度很难用眼分辨，所以人们一般将其排在后面)。显然，在建构移动类硬/软件协同设计模型时，设计目标应追求价格越低越好、功能越全越好、体积越小越好和速度越快越好。这些因素是描述元素的属性重要参考源。如前所述，移动嵌入式系统主要分为两大部分，硬件部分和软件部分。其各部分的组

成元素大致如表1所示。

这些组成元素都各自有不同的功能属性如价格、主要功能、体积大小和运算速度等。

在建构协同设计模型时，首先要突出移动特点，明确设计的定位，对设计对象的性能、目的、用途和价格进行仔细分析。其次，以设计目标为标准，划分组成系统的硬/软件元素，并按照各元素的性能参考列举其属性，以此作为在建立模型时选

表1 移动嵌入式系统基本组成元素

硬件部件名称	型号和规格
嵌入式处理器	MCU、MPU、DSP、SOC、ASIC和SOM……
协处理芯片	CS5530A、CS9211、AC97 Codec、RTL8139C……
存储器	DRAM、SRAM、Flash Memory、SDRAM、MINI FDD……
输入设备	键盘、鼠标、触摸屏、麦克风……
输出设备	LCD、打印机、发音器……
电源	各种型号和功率的镍、锌、锂离子电池
……	……
软件部件名称	
嵌入式操作系统	VRTX、PSOS、VXWORK、WINODWS CE、EPOC、LINUX、μc/OS、POWER PC……
嵌入式编程软件	C/C++、Visual C++、GNU C++、eVC、嵌入式Java和Modula-2……
应用软件	Pocket Office 2000、通讯录、电子词典、计算器、电子日历、电子游戏……
……	……

取组成元素的根据。接下来对照设计目标，将各元素的属性，按统一标准转化为相应的权重值。下面在定义组成元素属性的基础上，给出基于移动条件下的硬件/软件协同设计优化模型。

定义1 为了确定系统各组成元素的具体实现方式，本文对系统硬、软件组成元素作如下划分(式1)：

$$E\rightarrow\{HW,SW\} \tag{1}$$

其中E代表组成系统的所有组成元素的集合，HW,SW分别代表硬件、软件组成元素集合。

为获取最佳的系统性能，采用组成元素划分来满足约束条件，以实现优化方案的过程，同时挑选合适的组成元素实现优化方案。

定义2 定义1中的硬、软件组成元素的集合可定义为：

$$HW=\{H^k\mid k=1,\cdots,p\} \tag{2}$$

$$SW=\{S^k\mid k=1,\cdots,q\} \tag{3}$$

其中权重矩阵H^k,S^k分别表示系统硬、软件组成元素，并可表示如下：

$$H^k=(H_{ij}^k)_{m\times n}=\begin{bmatrix}H_{11}^k & H_{12}^k & \cdots & H_{1n}^k\\ H_{21}^k & H_{22}^k & \cdots & H_{2n}^k\\ \cdots & \cdots & \cdots & \cdots\\ H_{m1}^k & H_{m2}^k & \cdots & H_{mn}^k\end{bmatrix}, \tag{4}$$

$$(i=1,\cdots,m;j=1,\cdots,n)$$

由于每种硬件的型号和属性的个数不同，这里m,n取不确定的正整数。

$$S^k=(S_{rt}^k)_{u\times v}=\begin{bmatrix}S_{11}^k & S_{12}^k & \cdots & S_{1v}^k\\ S_{21}^k & S_{22}^k & \cdots & S_{2v}^k\\ \cdots & \cdots & \cdots & \cdots\\ S_{u1}^k & S_{u2}^k & \cdots & S_{uv}^k\end{bmatrix}, \tag{5}$$

$$(r=1,\cdots,u;t=1,\cdots,v)$$

同理，由于每种软件的类型和属性功能的个数不同，这里u,v也取不确定的正整数。

在式(2-4)，(2-5)中，H^k,S^k分别代表组成移动嵌入式系统的第k类硬件、第k类软件元素。比如，H^1代表CPU，H^2代表协处理芯片，……；S^1代表操作系统，S^2代表某种编程软件，……。

矩阵H^k的第i行第j列元素H_{ij}^k代表第k类硬件元素中的第j种型号所具有的第i种属性的权值；矩阵S^k的第r行第t列元素S_{rt}^k则代表第k类软件元素中的第t种类型所具有的第r种功能属性的权值。比如，H_{11}^1代表CPU的第1种型号(如MCU)的第1种属性(如价格)的权值，H_{22}^2代表处理芯片的第2种型号(如CS9211)的第2种属性(如运算速度)的权值，……；S_{11}^1代表操作系统的第1种类型(如VRTX)的第1种属性(如价格)的权值，S_{22}^2代表编程软件的第2种类型(如Visual C++)的第2种属性(如体积)的权值，……。

定义3 本文用式(6)、(7)来计算系统硬、软件各个组成元素的属性的权值：

$$H_{ij}^k=\pm\frac{h_{ij}^k}{\frac{1}{n}\sum_{j=1}^{n}h_{ij}^k}, \tag{6}$$

$$(i=1,\cdots,m;j=1,\cdots,n)$$

$$S_{rt}^k=\pm\frac{S_{rt}^k}{\frac{1}{v}\sum_{t=1}^{v}S_{rt}^k}, \tag{7}$$

$$(r=1,\cdots,u;t=1,\cdots,v)$$

其中h_{ij}^k,s_{rt}^k分别代表第k类硬、软件组成元素的相应型号的某种属性的原始值。比如，h_{11}^1代表CPU的第1种型号(如MCU)的第1种属性(如价格)的值(如1000元人民币)，……；s_{22}^2代表编程软件的第2种类型(如Visual C++)的第2种属性(如体积)的值(如600MB)，……。

为便于处理，由于设计目标的不同，各组成元素的相应属性的权值H_{ij}^k,S_{rt}^k可取正值或负值。

定义4 本文用如下矩阵范数来定义选择硬、软件的参考标准：

$$\|H^k\|_1=\min_{1\leqslant j\leqslant n}\sum_{i=1}^{m}H_{ij}^k \tag{8}$$

$$\|S^k\|_1=\min_{1\leqslant t\leqslant v}\sum_{r=1}^{u}S_{rt}^k \tag{9}$$

式(8)表示，选择第k类硬件功能模块中各项基本属性的权值和为最小的一种型号，作为协同设计的硬件组成部分；式(9)表示，选择第k类软件功能模块中各项基本属性功能的权值和为最小的一种类型，作为协同设计的软件组成部分。

定义5 本文用函数$F(HW,SW)$(式10)来定义协同设计中硬、软件功能模块的选择参考：

$$F(HW,SW)=\min_{\substack{1\leqslant i\leqslant m\\ 1\leqslant j\leqslant n\\ 1\leqslant r\leqslant u\\ 1\leqslant t\leqslant v}}\left(\sum_{k=1}^{p}\sum_{i=1}^{m}H_{ij}^k+\sum_{k=1}^{q}\sum_{r=1}^{u}S_{rt}^k\right) \tag{10}$$

在式(10)中令m,n和u,v取确定的值，且$i=r$，$j=t$，$m=u$，$n=v$。

根据式(10)所提出的移动嵌入式系统的硬/软件协同设计模型，可以为设计人员提供系统设计的优化方法。一旦确定了系统各个组成元素的价格、功能、速度、重量等因素后，即可通过式(6)(7)，确定其相应的权值，通过式(10)可以找到系统硬/软件组成元素的优化方案，设计出合适的目标产品。表2给出了硬件组成元素CPU的各种属性的原始值以及由式(6)得到的各个属性的权重值。表2列举了嵌入式CPU的权重值。

表 2 嵌入式 CPU 特性的权重值

嵌入式 CPU 名称	ARM920T	ARM7EJ-S	StrongARM SA-1110	Geode™ GX1-333	TMS320C6713
价格	1200.00	850.00	600.00	600.00	680.00
工作频率	250	133	206	330	300
体积	3.5mm²	3.0mm²	3.5 mm²	3.0mm²	3.5mm²
容量	—	—	—	—	—
工作电压	1.62V	1.08V	1.75V	2.2V	1.4V
汉化	0	0	0	0	0
价格 (权重)	1.527	1.081	0.763	0.763	0.865
工作频率(权重)	−1.025	−0.546	−0.845	−1.354	−1.231
体积 (权重)	1.061	0.909	1.061	0.909	1.061
容量 (权重)	0	0	0	0	0
工作电压(权重)	1.006	0.671	1.087	1.366	0.870
汉化 (权重)	0	0	0	0	0

则由式(4),可得到 CPU 的权重矩阵:

$$H^1=\begin{bmatrix} 1.527 & 1.081 & 0.763 & 0.763 & 0.865 \\ -1.025 & -0.546 & -0.845 & -1.354 & -1.231 \\ 1.061 & 0.909 & 1.061 & 0.909 & 1.061 \\ 0 & 0 & 0 & 0 & 0 \\ 1.006 & 0.671 & 1.087 & 1.366 & 0.870 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{11}$$

同理,由式(4),(5)可分别得到组成元素 FLASH 存储器、操作系统和嵌入式数据库应用软件的权重矩阵 H^2,S^1,S^2 如下:

$$H^2=\begin{bmatrix} 1.011 & 0.647 & 0.539 & 0.916 & 1.887 \\ 0 & 0 & 0 & 0 & 0 \\ 2.577 & 0.638 & 0.542 & 0.814 & 0.429 \\ -1.265 & -0.316 & -0.316 & -0.632 & -2.470 \\ 1.1 & 0.9 & 1 & 0.9 & 1.1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{12}$$

$$S^1=\begin{bmatrix} 1.786 & 0.089 & 0.074 & 2.976 & 0.074 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 2.239 & 0.784 & 0.672 & 1.209 & 0.096 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1.667 & 1.667 & 1.667 & 0 \end{bmatrix} \tag{13}$$

$$S^2=\begin{bmatrix} 4.808 & 0 & 0 & 0 & 0.192 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0.886 & 0.237 & 1.499 & 1.425 & 0.954 \\ 0 & 0 & 0 & 0 & 0 \\ 1.25 & 1.25 & 1.25 & 1.25 & 0 \end{bmatrix} \tag{14}$$

由式(10)得

$F(H^1, H^2, S^1, S^2) = \min\limits_{1\le j\le 5}\{16.961, 8.011, 8.993, 12.209, 3.827\} = 3.827$ 说明选择第五列硬、软件组成元素作为系统的协同设计的方案,可以达到优化的效果。

小结 随着嵌入式系统复杂性的不断提高,当前热门的硬/软件协同设计方法已显现出优势。本文采用系统组件的权重值为参考,组成元素划分为依据的设计理念,构建了移动嵌入式系统设计的 EHSC 方法,试图寻找移动嵌入式系统设计中硬/软件协同设计的最佳搭配方案。并依照此模型,完成一种移动嵌入式系统"电子书包"[6]阅读器的设计和实现。由于不同移动嵌入式系统对成本的要求以及各种限制条件,在设计复杂移动嵌入式系统时,可以根据计算的复杂性和具体的限制条件,将应用系统按照不同领域分类,以便用科学和系统的方法而不是靠经验来选择目标结构。因此在这个研究领域还有许多研究工作要做。

参 考 文 献

1 Chou P, Ortega R B, Borriello G. Interface Co-Synthesis Techniques for Embedded Systems. In: Proc. of the IEEE/ACM International Conference on Computer-Aided Design. San Jose, CA, Nov. 1995

2 Athanas P, Silverman H F. Processor Reconfiguration Through Instruction-Set Metamorphosis. Computer, 1993, 26(3)

3 Kumarjames S, et al. THE CODESIGN OF EMBEDDED SYSTEMS. Kluwer Academic Publishers, 1996

4 Thomas F, Nayak M M, Udupa S, Kishore J K, Agrawal V K. A hardware/software codesign for improved data acquisition in a processor based embedded system, Microprocessors and Microsystems, 2000

5 郭晓东,刘积仁. 一种基于遗传算法的硬件/软件划分方法. 计算机辅助设计与图形学学报,2000(4)

6 郑世珏,张江陵. 基于嵌入式系统模式的电子课本. 计算机工程,30(4):193～195

(上接第 209 页)

11 Chavez C, Lucena C. A Metamodel for Aspect-Oriented Modeling. Workshop on Aspect-Oriented Modeling with the UML. In: The First Conf. on Aspect-Oriented Software Development (AOSD 2002), the Netherlands, April 2002

12 Kandé, Kienzle, Strohmeier. From AOP to UML: Towards na Aspect-Oriented Architetural Modeling Approach. Agosto, 2002

13 Katara M, Katz S. Architectural views of aspects. In: Proc. of the 2nd Intl. Conf. on Aspect-oriented software development, Boston, Massachusetts, March 2003

14 Rumbaugh J, 等编. 姚淑珍, 唐发根, 等译. UML 参考手册. 北京: 机械工业出版社, 2001

15 Aldawud O, Bader A, Ellrad T. Weaving with Statecharts. Aspect-Oriented Modeling with UML workshop at the 1st International Conference on Aspect-Oriented Software Development, April 2002

16 Mahoney M, Bader A, Elrad T, et al. Using Aspects to Abstract and Modularize Statecharts. In: The 5th Aspect-Oriented Modeling Workshop in Conjunction with UML 2004 Lisbon, Portugal, Oct. 2004

17 Aldawud O, Bader A, Constantinides C, et al. Modeling Intra-Object Aspectual Behavior. In: Proc. of the 1st ICSE Workshop on Describing Software Architecture with UML, Toronto, Canada, May 15, 2001

18 Groher I, Schulze S. Generating Aspect Code from UML Models. The 4th AOSD Modeling With UML Workshop, 2003

19 Selic B, Rumbaugh J. Using UML for Modeling Complex Real-Time Systems. ObjecTime Limited/Rational Software whitepaper, 1998

20 OMG Document ad/02-04-01. UML Profile for CORBA Specification. http://www.omg.org

21 Stein D, Hanenberg St, Unland R. Designing Aspect-Oriented Crosscutting in UML. In: AOSD-UML Workshop at AOSD'02, Enschede, The Netherlands, Apr. 2002

22 Han Y, Kniesel G, Cremers A B. A Meta Model and Modeling Notation for AspectJ. http://www.cs.iit.edu

23 Harrison W, Tarr P, Ossher H. A position on considerations in UML design of aspects. In: Position paper in Workshop on Aspect-Oriented Modelling with UML in conjunction with AOSD 2002, Enschede, The Netherlands, Apr. 2002