第 5 章　软件体系结构集成开发环境

5.1　软件体系结构集成开发环境的作用

5.1.1　形式化描述方法的比较

第 3 章提到了使用形式化方法描述软件体系结构，但是仅依靠形式化方法，很难适应软件进一步发展的需要。近年来，越来越多的研究者注重在特定领域对体系结构开发工具进行研究，从形式化方法到体系结构集成开发环境的转变是软件体系结构发展的趋势。而为了支持基于体系结构的发展，研究者也更倾向于功能强大的辅助开发工具——软件体系结构集成开发环境。

与形式化描述方法相比，体系结构集成开发环境研究软件体系结构具有以下几个优点。

- 集成开发环境使开发者摆脱了繁杂的语法、语义、标识符号和公式的困扰，可以集中精力设计系统的结构。
- 对于规模庞大、结构复杂的软件系统，资源的有效管理和利用极其重要。集成开发环境使用文件系统有效地管理和支配文件和文件夹等资源，用文档支持系统开发和维护，使项目跟踪和控制变成可能，有效地提高了软件生产率，降低了开发和维护成本，保证了软件产品的质量。
- 集成开发环境把开发过程中所需的各项功能集合在一起，实现了体系结构分析、设计、建模、验证的自动化，这是形式化方法无法取代的。
- 集成开发环境提供了友好的图形用户界面和可视化操作，形象化了开发过程和结果。

上述优点是从开发者的角度阐述的。从用户角度出发，集成开发环境给用户提供了一个清晰明确、易于理解的软件设计产品。形象化描述不仅是系统相关人员相互沟通交流的工具，达成共识的基础，也是他们学习和理解的助手。总之，体系结构集成开发环境的出现迎合了软件体系结构的发展。

5.1.2　体系结构集成开发环境的作用

目前已经出现了很多支持体系结构的分析工具，如支持静态分析的工具、支持类型检查的工具、支持体系结构层依赖分析的工具、支持体系结构动态特性仿真工具、体系结构性能仿真工具等。本节将探讨集成开发环境有哪些具体功能。

集成开发环境是一个集编辑、编译、运行计算机程序于一体的工具。体系结构集成开发环境基于体系结构形式化描述，从系统框架的角度关注软件开发。体系结构开发工具是体系结构研究和分析的工具，给软件系统提供了形式化和可视化的描述。它不但提供了图形用户界面、文本编辑器、图形编辑器等可视化工具，还集成了编译器、解析器、校验器、仿真器等工具；

不仅可以针对每个系统元素,还支持从较高的构件层次分析和设计系统,这样可以有效地支持构件重用。具体来说,体系结构集成开发环境的功能可以分为五类:辅助体系结构建模,支持层次结构的描述,提供自动验证机制,提供图形和文本操作环境,支持多视图。

1. 辅助体系结构建模

建立体系结构模型是体系结构集成开发环境最重要的功能之一。集成开发环境的出现增加了软件体系结构描述方法的多样性,摒弃了描述能力低的非形式化方法,摆脱了拥有繁杂语法、语义规则的形式化方法。开发者只需经过简单的操作就可以完成以前需耗费大量时间和精力的工作。在5.1.1节中提到,形式化时期建模是将软件系统分解为相应的组成成分,如构件、连接器等,并用形式化方法严格地描述这些组成成分及它们之间的关系,然后通过推理验证结果是否符合需求,最后提供量化的分析结果。而集成开发环境提供了一套支持自动建模的机制以完成体系结构模型分析、设计、建立、验证等过程。用户可根据不同的实际需求、应用环境和体系结构等因素选择不同的开发工具。

2. 支持层次结构的描述

软件系统规模越来越大、越来越复杂,简单体系结构风格已无法表达结构复杂的系统。这时就需要层次结构的支持,因此开发工具也需要提供层次机制。图5-1描述了一个简单的具有层次结构的客户端-服务器系统。

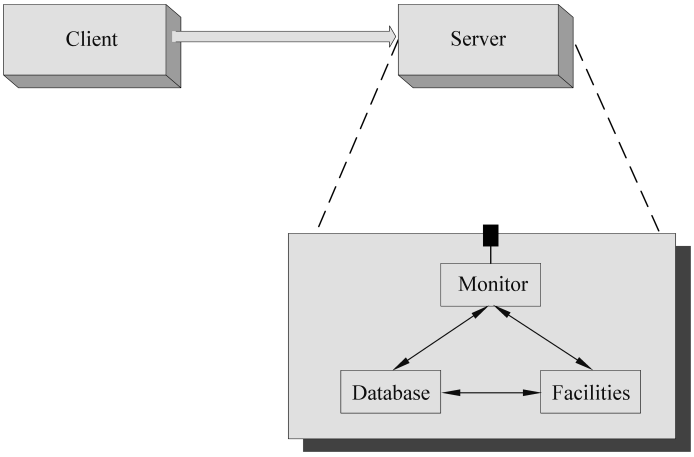

图5-1 层次结构的客户端-服务器系统

系统由客户端和服务器两个构件组成,客户端可以向服务器传输信息。服务器是一个包含了三个构件的复杂元素,内部构件之间相互关联形成了一个具有独立功能的子系统,子系统通过接口与外界交互。体系结构集成开发环境提供了子类型和子体系结构等机制来实现层次结构。用户还可以根据需要自定义类型,只需将这种类型实例化为具体的子系统即可。类似构件、连接器也可以通过定义新类型表达更复杂的信息。

3. 提供自动验证机制

几乎所有的体系结构集成开发环境都提供了体系结构验证的功能。体系结构描述语言

解析器和编译器是集成开发环境中必不可少的模块。除此以外，不同的集成开发环境根据不同的要求会支持特定的检验机制：WRIGHT 提供模型兼容性检测器来测试构件和连接器死锁等属性，通过一组静态检查来判断系统结构规模说明的一致性和完整性，同时还支持针对某一特定体系结构风格进行检查；C2 风格通过约束构件和连接器的结构和组织方式来检查一致性和完整性；SADL 利用体系结构求精模式保证使用求精模式的实例的每一步求精过程的正确性，采用这种方式能够有效地减少体系结构设计的错误；ArchStudio 中的 Archlight 不但支持系统的一致性和完整性检查，还支持软件产品线的检测。

集成开发环境的测试方式可分为主动型和被动型两种。主动型是指在错误出现之前采取预防措施，是保证系统不出现错误状态的动态策略。它根据系统当前的状态选择恰当地设计决策保证系统正常运行。例如，在开发过程中阻止开发者选择与接口不匹配的构件；集成开发环境不允许不完整的体系结构调用分析工具。被动型是指允许错误暂时存在，但最终要保证系统的正确性。被动型有两种执行方式，有的允许预先保留提示错误，稍后再作修改，有的必须强制改正错误后系统才能继续运行。例如，在 MetaH 的图形编辑器中，启动“应用”按钮之前必须保证系统是正确的。

4. 提供图形和文本操作环境

体系结构集成开发环境是开发者研究体系结构的可视化工具和展示平台，它具有友好的图形用户界面和便捷的操作环境。体现在以下四个方面。

- 集成开发环境提供了包含多种界面元素的图形用户界面，例如工具栏、菜单栏、导航器视图、大纲视图等。工具栏显示了常用命令和操作，视图以列表或者树型结构的形式对信息进行显示和管理。
- 集成开发环境提供了图形化的编辑器，它用形象的图形符号代表含义丰富的系统元素，用户只需选择需要的图形符号，设置元素的属性和行为并建立元素之间的关联就可以描绘系统。例如，Darwin 系统提供基本图元代表体系结构的基本元素，使用空心矩形表示构件，直线表示关联，圆圈表示接口。每个图元都有自己的属性页，通过编辑构件、关联和接口的属性页来设置体系结构的属性值。
- 集成开发环境利用文本编辑器帮助开发者记录和更新体系结构配置和规格说明。通常，集成开发环境会根据模型描述的系统结构自动生成配置文档。当模型被修改时，它的文本描述也会发生相应的变化，这种同步机制保证了系统的一致性和完整性。
- 集成开发环境还支持系统运行状态和系统检测信息的实时记录，这些信息对分析、改进、维护系统都很有价值。

5. 支持多视图

多视图作为一种描述软件体系结构的重要途径，是近年来软件体系结构研究领域的重要方向之一。随着软件系统规模不断增大，多视图变得更为重要。每个视图都反映了一组系统相关人员关注的系统的特定方面。多视图体现了关注点分离的思想，把体系结构描述语言和多视图结合起来描述系统的体系结构，能使系统更易于理解，方便系统相关人员之间相互交流，还有利于系统的一致性检测及系统质量属性的评估。图形视图和文本视图是两

种常见的视图。图形视图是指用图形图像的形式将系统的某个侧面表达出来的结果。它是一个抽象概念,不是指具体的哪一种视图。逻辑视图、物理视图、开发视图等都属于图形视图。同样,文本视图是指用文字形式记录系统信息的视图。此外,还存在很多特殊的体系结构集成开发环境特有的视图。例如,Darwin 系统中的分层系统视图、ArchStudio 的文件管理视图、Aesop 支持特定风格形象化的视图等。

5.2 体系结构 IDE 原型

现阶段,出现了越来越多的体系结构集成开发环境以满足种类繁多的体系结构和灵活多变的需求。尽管这些集成开发环境针对不同的应用领域,适用不同的体系结构,但是它们都依赖相似的核心框架和实现机制。把这些本质的东西抽象出来,总结得出一个体系结构集成开发环境原型。该原型只是一个通用的框架并不能执行任何实际的操作。它可以帮助开发人员深入理解开发工具的结构和工作原理,结合 XArch(eXtensible Architecture Research System)系统来介绍原型。

从集成开发环境的工作机制看,原型是三层结构的系统。最上层(用户界面层)是系统和外界交互的接口;中间层(模型层)是系统的核心部分,系统重要的功能都被封装在该层。本层通过接口向用户界面层传输数据,用户界面层要依赖这一层提供的服务才能正常运行;底层(基础层)覆盖了系统运行的所必需的基本条件和环境,是系统正常运行的基础保障。此外,模型层和用户界面层的正常运行还需要映射模块的有效支持,映射文件将指导和约束这两层的行为(见图 5-2)。

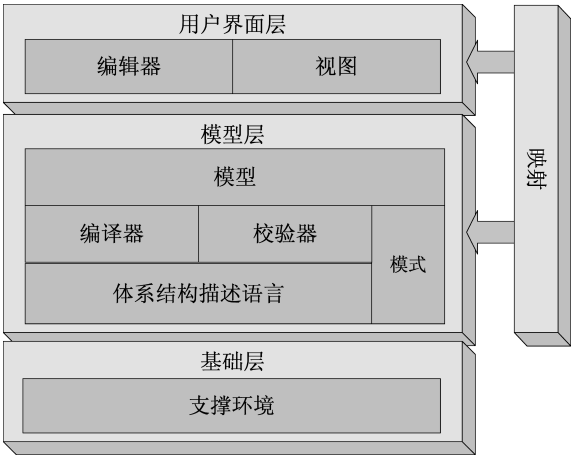

图 5-2 原型框架

5.2.1 用户界面层

用户界面层是用户和系统交互的唯一渠道,用户需要的操作都被集成到这一层。这些操作可以通过编辑器和视图来实现。编辑器是开发环境中的可视构件,它通常用于编辑或浏览资源,允许用户打开、编辑、保存处理对象,类似的文件系统应用工具如 Microsoft Word,执行的操作遵循“打开—保存—关闭”这一生命周期模型。同一时刻工作台窗口允许

一个编辑器类型的多个实例存在。视图也是开发环境中的可视构件，它通常用来浏览分层信息、打开编辑器或显示当前活动编辑器的属性。与编辑器不同的是，同一时刻只允许特定视图类型的一个实例在工作台中存在。编辑器和视图可以是活动的或者不活动的，但任何时刻只允许一个视图或编辑器是活动的。

XArch 系统的工作台是一个独立的应用窗口，包含了一系列视图和编辑器。工作台基于富客户端平台(Rich Client Platform)，它最大的特点是支持用户建立和扩展自己的客户应用程序。如果现有的编辑器不能满足需求，用户可以灵活地在接口上扩展新的功能。

图 5-3 显示了 XArch 系统的部分编辑器和视图。左侧的系统资源管理器视图将系统所有的信息以树型结构显示出来。右边的属性视图显示了考察对象的属性和属性值，其下面是记录系统重要状态的日志视图。占据工作台最大区域的是中间的编辑器，是主要的对象处理场所。为了满足系统相关人员不同的需求，系统支持多视图。系统用标签对多个视图进行区分和管理，用户可通过选择标签在不同的视图间转换。

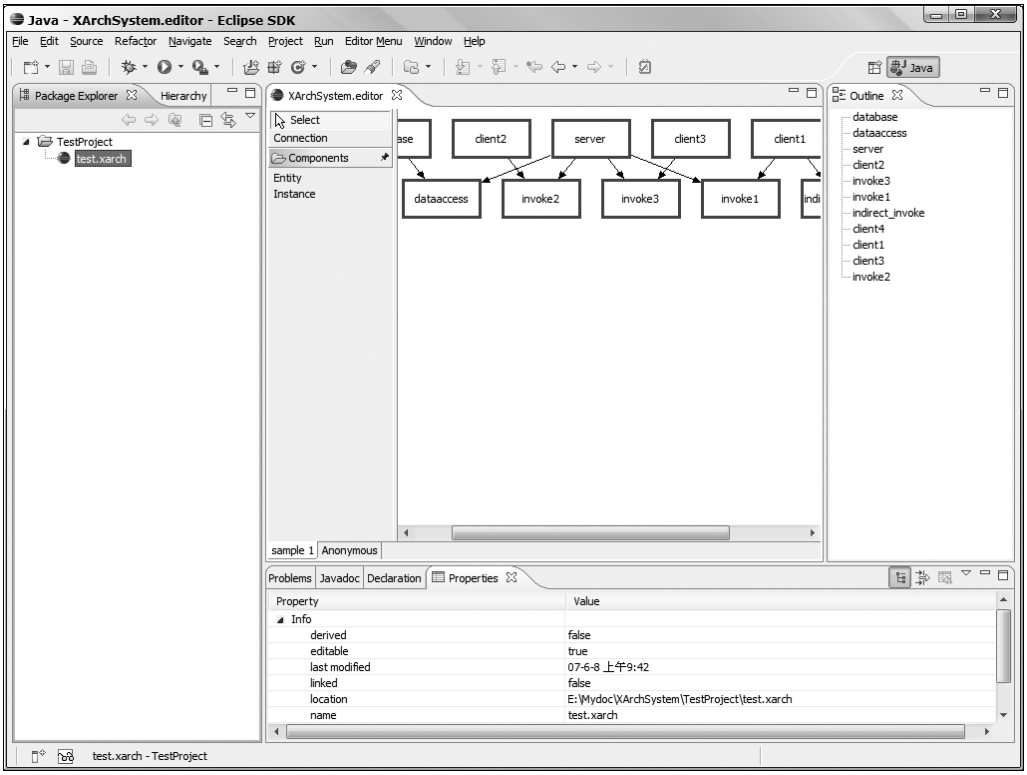

图 5-3　XArch 系统

5.2.2　模型层

模型层是系统的核心层，系统的大部分功能都在这一层定义和实现，它主要的任务是辅助体系结构集成开发环境建立体系结构模型。

体系结构描述语言文档是系统的输入源。有的体系结构集成开发环境对描述语言的语法有限制或约束，这就需要修改语言的语法以便与其兼容。输入的体系结构文档是否合法有效，则由专门的工具来检验。此处的编译器不同于往常的把高级程序设计语言转化为低

级语言(汇编语言或机器语言)的编译器,它是一个将体系结构描述转化为体系结构模型的工具。为实现此功能,编译器一般要完成下列操作:词法分析、解析、语义分析、映射、模型构造等。词法分析是依循语言的词法规则,扫描源文件的字符串,识别每一个单词,并将其表示成所谓的机内 token 形式,即构成一个 token 序列;解析过程也叫语法分析,是指根据语法规则,将 token 序列分解成各类语法短语,确定整个输入串是否构成一个语法上正确的程序,它是一个检查源文件是否符合语法规范的过程;语义分析过程将语义信息附加给语法分析的结果,并根据规则执行语义检查;映射是根据特定的规则(如映射文档)将体系结构描述语言符号转换成对应的模型元素的过程;模型构造紧跟映射过程,它把映射得到的构件、连接器、接口等模型元素按语义和配置说明构造成一个有机整体。在编译器工作的过程中会有一些隐式约束的限制,例如类型信息、构件属性、模块间的关系等;校验器是系统最主要的检查测试工具,采用显式检验机制检查语法语义、类型不一致性、系统描述二义性、死锁等错误,以保证程序正常运行;模式是一组约束文档结构和数据结构的规则,它是判断文档、数据是否有效的标准;映射模块是抽象了体系结构描述语言元素和属性的一组规则,这组规则在模型层和用户界面层担任了不同的角色。在模型层,它根据映射规则和辅助信息,将开发环境无法识别的体系结构描述语言符号映射成可以被工具识别的另一种形式的抽象元素。在用户界面层支持模型显示,它详细定义了描述语言符号如何在模型中表示,如何描绘模型元素及它们之间的关系。

建立体系结构模型是本层的最终目标,模型层用树或图结构抽象出系统,形象地描述了系统的各构件及它们之间的关系。通常,一个系统用一个体系结构模型表示。对于一个规模庞大、关系复杂的模型,不同的系统相关人员只需侧重了解他们关注的局部信息,而这些信息之间具有很强的内聚性,可以相对独立地存在。针对某一观察角度和分析目的,提取一系列相互关联且与其他内容相对独立的信息,就可以构成软件体系结构视图。一个模型可以构造出多种视图,以通过不同的视角细致全面地研究系统。

XArch 系统只处理基于 XML 的可扩展的体系结构描述语言,即 FEAL 兼容的体系结构描述语言,如果不符合这一要求,可以适当调整语法结构以满足 FEAL 的规范。软件体系结构描述不仅是 XML 结构良好的,还必须是符合模式规定的有效的文档。该系统不但支持对系统的分析、验证和序列化等操作,还支持视图和模型之间的相互转化。

XArch 系统不仅是一个体系结构开发环境,还是一个扩展工具的平台。它的扩展性主要体现在两个方面:一是可以灵活地创建和增加一种新的软件体系结构描述语言或语言的新特性,以满足新功能和新需求。如图 5-4 所示,系统通过引入一个中间介质 FEAL,使模型脱离与体系结构描述语言的直接联系,从而拓展了体系结构描述语言符号到模型元素固定的对应关系。体系结构描述语言的元素首先根据映射规则被映射为 FEAL 元素(FEC)的形式,FEC 再对应到相应的模型的构件。因此,只要体系结构描述语言符号到 FEC 的映射是有效的,那么无论哪种体系结构描述语言都可以构造对应的体系结构模型。当新的体系结构描述语言或新的语言特性出现时,只须修改映射规则就能有效地支持。二是 XArch 系统提供了一系列可扩展的可视化编辑接口,支持定义新界面元素。

5.2.3 基础层

这一层是系统的基本保障,涵盖了系统运行所需的软硬件支撑环境,它还对系统运行时

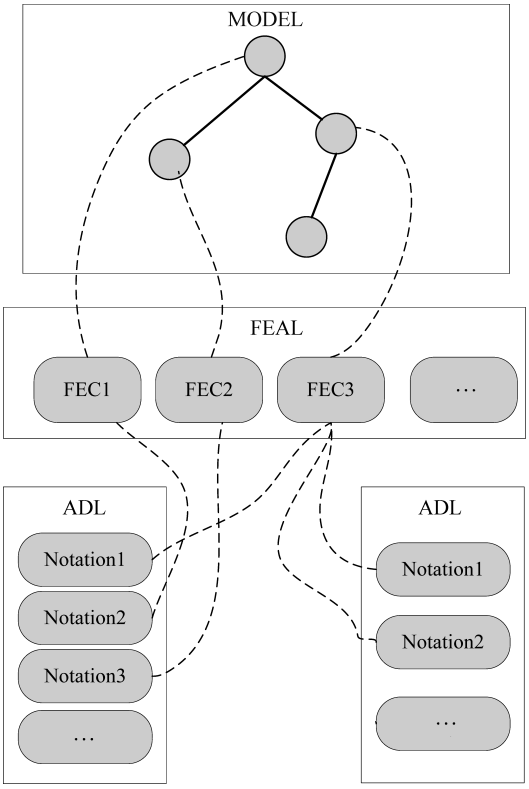

图 5-4　ADL、FEAL、MODEL 的关系

所用的资源进行管理和调度。通常，普通的简单配置就可以满足系统运行需求，但是有的体系结构集成开发环境需要更多的支持环境。例如，ArchStudio 5 作为 Eclipse 的插件，必须在 Java 和 Eclipse 环境下运行。

5.2.4　体系结构集成开发环境设计策略

目前，集成开发环境都很注重体系结构的可视化和分析，有的也在体系结构求精、实现和动态性上具有强大的功能。体系结构开发环境原型提供了一个可供参考的概念框架，它的设计和实现需要开发人员的集体努力。下面是体系结构集成开发环境设计的三条策略。

（1）体系结构集成开发环境的设计必须以目标为导向。集成开发环境的开发遵循软件开发的生命周期，需求分析是必须而且非常重要的阶段。开发者只有明确了实际需求，才能准确无误地设计。无论是软件本身还是最终用户都有很多因素需要确认。例如，集成开发环境可以执行什么操作？怎么执行？它的结构是什么？哪一种体系结构描述语言和体系结构风格最合适它？哪些用户适合使用该系统？怎样解决系统的改进和升级？这些问题为设计提供了指导和方向。

（2）为了设计一个支持高度扩展的体系结构集成开发环境，必须区分通用和专用的系统模块。通用模块部分是所有集成开发环境都必备的基础设施，例如支撑环境、用户界面等。但是不同的体系结构集成开发环境针对不同的领域，需要解决千差万别的问题，因此每种体系结构集成开发环境都有自己的特点。例如，Rapide 的开发环境建立一个可执行的仿

真系统并提供检查和过滤事件的功能,以此来允许体系结构执行行为的可视化;SADL 的支持工具支持多层次抽象和具有可组合性的体系结构的求精。它要求在抽象和具体的体系结构之间建立名字映射和风格映射,两种映射通过严格地验证后,才能保证两个体系结构在求精意义上的正确性。这样可以有效地减少体系结构设计的错误,并且能够广泛、系统地实现对设计和正确性证明的重用。

(3) 结构集成开发环境原型。原型框架为可扩展性开发工具的设计提供了良好的接口。例如 XArch 系统可以通过添加语言符号或者定义 FEAL 兼容的体系结构描述语言来扩展现有的功能。这样,体系结构专用的功能就可以作为动态插件应用到集成开发环境中,增强开发工具的功能,扩大它的使用范围。

5.3 ArchStudio 5 系统

5.3.1 ArchStudio 5 简介

当前流行的体系结构集成开发环境很多,由于篇幅原因不能一一介绍。本书选择其中两个典型代表(ArchStudio 5、SysADL Studio),通过详细介绍 ArchStudio 5 系统来深入了解体系结构集成开发环境。

1. 软件体系结构集成开发环境的发展

自从第一个支持 BASIC 的集成开发环境出现以来,集成开发环境的发展就一直没有停止。它已经从最初仅在控制台和终端做一些简单的系统开发,发展到现在的可以完成大型系统开发的独立程序。它不再是一个简简单单的命令行工具,而是提供了系统设计、修改、编译、部署、验证、实施和评估等多种功能的综合性工具。集成开发环境的发展经历了三个阶段:第一阶段是以存储为中心的集成开发环境,所有的工具都围绕一个共享的数据库为中心进行工作。Ada 的开发环境就是这种类型的原型;Interlisp 是一个依赖共享解析树工作的实例;带版本号的文件系统是这种工具的变型,例如著名的修订控制系统(Revision Control System,RCS)。19 世纪 80 年代出现了以进程为中心的第二阶段,着重考虑开发进程和相关的工作流。例如 Marvel 帮助开发者自动执行基本程序并协同工具的扩展开发工作。目前,以体系结构为中心的第三代集成开发环境影响着软件开发的整个生命周期。典型代表 ArchStudio 不但支持体系结构版本存储和程序自动化,还提供体系结构设计、评估、实施和编辑等功能。这一阶段的支持工具需要一个开放的网络环境来展示整个产品。基于体系结构的集成开发环境将成为体系结构发展领域的主流。

2. ArchStudio 5

ArchStudio 5 是美国加利福尼亚大学欧文分校的软件研究实验室开发的面向体系结构的、基于 xADL 3.0 的开源集成开发环境。xADL 3.0 是基于 XML 模式定义的体系结构描述语言。除了具备普通的体系结构建模功能,还提供了对系统运行时刻和设计时刻的元素的建模支持,类似版本、选项和变量等更高级的配置管理观念,以及对软件产品线的体系结构的建模支持。此外,xADL 3.0 还利用 XML 的可扩展性简化了新的软件体系结构描述语

言的设计及其相应工具的开发过程。xADL 3.0 体系结构是符合 xADL 3.0 模式定义的简单的 XML 文档。

ArchStudio 5 在前一版的基础上添加了新的特性和功能，在可扩展性、系统实施和工程性上有了新的发展。ArchStudio 5 的作用主要体现在基本功能和扩展功能两方面。它不但实现了建模、可视化、检测和系统实施等基本功能，还良好地支持了这些功能的扩展。

（1）**建模**：作为软件体系结构开发辅助工具，ArchStudio 5 最主要的功能就是帮助用户用文档或者图形方式表达设计思想。模型像建筑蓝图一样从较高的角度把系统抽象成一个框架，抽象的结果将以 XML 的形式存储和操作。用户可以利用系统多个视角对该模型进行考察和研究。此外，ArchStudio 5 还支持体系结构分层建模，软件产品线建模，而且可以时刻监视变化的体系结构。

（2）**可视化**：ArchStudio 5 提供了多种可视化的构件，例如视图和编辑器。视图和编辑器用文本或图形方式形象化描述体系结构，例如 Archipelago、ArchEdit 等工具，同时也给系统相关者提供了交互和理解的平台。

（3）**检测**：ArchStudio 5 集成了功能强大的体系结构分析和测试工具 Schematron。它通过运行一系列预定的或用户定义的测试来检查系统。Archlight 可根据标准自动测试体系结构描述的正确性、一致性和完整性等。检查出来的错误会显示出来，同时帮助用户定位出错的地方并提供修改的途径和方法。

（4）**实施**：它帮助将体系结构运用到实施的系统中。ArchStudio 5 使用自己的体系结构设计思想和方法来实现自身。ArchStudio 5 的体系结构用 xADL 3.0 详细描述，这些文件都是实施的一部分。一旦 ArchStudio 5 在机器上运行，它的体系结构描述将被解析，这些信息将被实例化并连接到预定的构件和连接器上。

除此以外，ArchStudio 5 对上述功能提供了良好的扩展机制。由于它基于 xADL 3.0，而 xADL 3.0 是模块化的，不是一个独立的整体。它没有将所有词法和语法一起定义，而是采用类似 XML 模式分解模块的方式。如图 5-5 所示，每个模块相互分离，侧重实现系统的某一功能，四个模块都与中间的模块交互，五个模块共同组成了一个有机的系统。例如，可将构件和连接器分解为多个相互关联的模块。目前，模块技术已经不只能处理构件和连接器等低层次的构件，还能处理软件产品线、实施映射、体系结构状态等。ArchStudio 5 根据模式自动生成一个数据绑定库以方便提供别的工具共享功能。如图 5-6 所示，用户就可以扩展 xADL 语言的新特性并自动生成支持新特性交互的库。总之，ArchStudio 5 在 xADL 3.0 的支持下允许开发者定义新的语义和规则去获取更多的数据信息来满足新的需求。

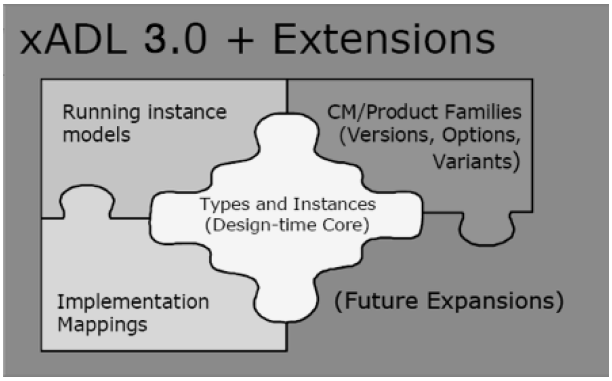

图 5-5　xADL 3.0 结构

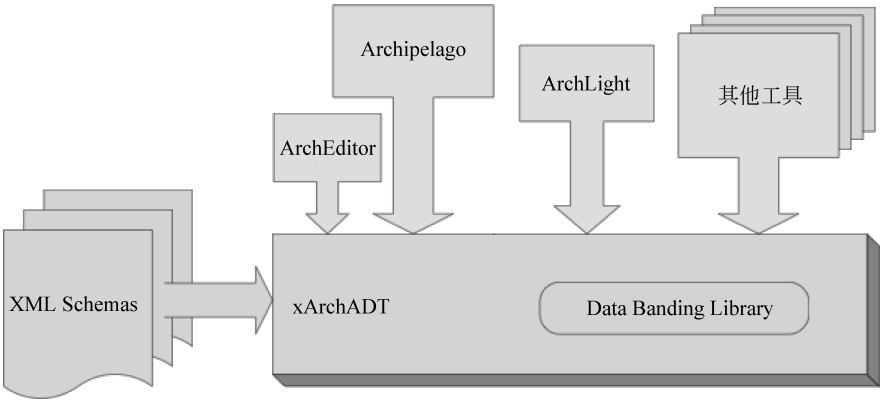

图 5-6 ArchStudio 5 的工具

- 可扩展的建模：开发 ArchStudio 5 的目标就是要实现体系结构建模的可扩展性。它基于第一种可扩展的体系结构描述语言 xADL 3.0，利用添加新的 XML 模式来支持模型扩展。
- 可扩展的可视化：可视化编辑器利用可扩展的插件机制添加对新体系结构描述语言元素编辑的功能。
- 可扩展的检测：用户可以在 Schematron 中设计新的测试，也可以集成新的分析引擎来满足高要求的检验。在 ArchStudio 5 中，所有的检测工具都作为 ArchLight 插件使用，因此用户可以通过添加插件完成新的测试。ArchLight 集成了功能强大的 Schematron XML 分析引擎，其他测试引擎也可以无缝地集成到 ArchLight 中(见图 5-7)。
- 可扩展的实施：用户可以灵活地把体系结构与 Myx 框架绑定起来。Myx 是在 ArchStudio 5 中建立的体系结构风格。此风格适合开发高性能的灵活的集成开发环境(Myx -whitepaper)。它定义了一套构件和连接器的构建规则，提供了定义构件同步和异步交互的模式，同时还规定了哪些构件可以相互约束，确定了构件间直接或者分层的关系。在 Myx 风格的约束下，构件之间的相对独立有利于构件重用，构件只能通过显示接口与外界传递消息。因此不需对构件重新编码就可以在不同配置的构件间建立联系。此外，动态代理和事件处理机制支持在运行时刻控制连接的状态。

5.3.2 ArchStudio 5 安装

硬件配置需求：硬件配置取决于具体的实际应用需求，例如程序规模、程序预期的运行时间等。对于 ArchStudio 5 来说，可使用 x86 体系结构兼容的计算机，Intel Pentium Ⅲ处理器、128MB 内存以上的配置即可。

软件配置需求：ArchStudio 5 是开源开发工具 Eclipse 的插件。它可以在任何支持 Eclipse 的系统上运行。因此，必须有 JRE 1.7(或更高版本)和 Eclipse 4.3(或更高版本)的支持。

安装过程只需按照安装向导进行，具体的步骤如下。

- 在 Eclipse 菜单栏上单击 Help，在菜单列表中选择 Install New Software；
- 在弹出窗口 Work with 一栏输入 http://www.isr.uci.edu/projects/archstudio-5/updatesite-4.3，并且按 Enter 键；