第5章 软件架构描述语言

目　　录

5.1 引言

- 根据ISO/IEC/IEEE 42010标准，软件架构描述语言（ADLs，architecture description language）就是任何用于软件架构的表示形式。
- 目前主要的架构描述语言有Aesop、MetaH、C2、Rapide、SADL、Unicon和Wright等，尽管它们都描述软件架构，却有不同的特点。

- 这些ADL强调了架构不同的侧面：
  - Aesop支持架构风格的应用
  - MetaH为设计者提供了关于实时电子控制软件系统的设计指导
  - C2支持基于消息传递风格的用户界面系统的描述
  - Rapide支持架构设计的模拟并提供了分析模拟结果的工具
  - SADL 提供了关于架构的形式化基础
  - Unicon支持异构的组件和连接类型并提供了关于架构的高层编译器
  - Wright支持架构组件之间交互的说明和分析等

5.1 引言

- 但是，每一种ADL都以独立的形式存在，描述语法不同且互不兼容，同时又有许多共同的特征，这使设计人员很难选择一种合适的ADL。

5.1 引言

- ADL与需求语言的区别：后者描述的是问题空间，而前者则扎根于解空间中。
- ADL与建模语言的区别：后者对整体行为的关注要大于对部分的关注，而ADL集中在组件的表示上。
- ADL与传统的程序设计语言的构成元素既有许多相同和相似之处，又各自有着很大的不同。

5.2 ADLs的核心设计元素

- 根据ISO/IEC/IEEE 42010标准，一个ADL至少有如下核心设计元素：
  - **组件（Component）**：表示系统中主要的计算元素和数据存储，如客户端、服务器、数据库等；
  - **连接件（Connector）**：定义了组件之间的交互关系，如过程调用、消息传递、事件广播等；
  - **软件架构配置（Architecture Configuration）**：描述组件、连接件之间的拓扑关系；
  - **约束条件（constraint）**：定义组件之间依赖、组件与连接件之间依赖的约束
  - 通常，ADL还会采用一种形式化技术作为语义信息描述的理论基础

5.3 几种典型ADLs

- 12种比较有代表性的ADLs：
  - Aesop、C2 SADL、MetaH、Unicon、Rapide、Wright、Darwin、XYZ/ADL、ACME、xADL和ABX/ADL等。

5.3几种典型ADLs

- C2 SADL语言简介:
  - C2是一种用于用户界面密集系统的软件架构风格。
  - C2 SADL（Software Architecture Description & Evolution Language）是用于描述C2风格的架构的ADL，它为架构的演化提供了特别支持。

5.3几种典型ADLs

- **适用范围和目地**：主要面向C2风格软件架构的描述，适用于分布式异构环境中、基于消息的图形用户界面应用系统的设计。
- **基本元素**：组件、连接件、以及它们之间的拓扑关系。组件间只能通过连接件相连，具有“受限的可见性”；异步通知消息和请求消息是组件间唯一的通信方式。
- **具备的主要能力**：架构演化；架构动态配置；多形式的类型检查；设计决策过程支持；可执行代码生成。

5.3几种典型ADLs

- 语法语义:

C2 对组件的描述:

```
Component ::=
        component component_name is
                interface component_message_interface
                parameters  component_parameters
                methods  component_methods
                [behavior component_behavior]
                [context component_context]
        End component_name;
```

5.3几种典型ADLs

- 语法语义:

C2 对接口的描述:

```
Component_message_interface ::=
        top_domain_interface
              bottom_domain_interface
              top_domain_interface ::=
                    top_domain is
                     out interface_request
                      in interface_notifications
             bottom_domain_interface ::=
              bottom_domain is
                            out interface_notifications
                    in interface_request
              interface_request ::= {request; } | null;
              interface_notifications ::= {notification; } | null;
              request ::= message_name(request_parameters)
              request_parameters ::= [to
component_name][parameter_list]
              Notification ::=
        Message_name[parameter_list]
```

编号	典型的ADLs	应用场景
1	Aesop	特定领域的软件系统
2	C2 SADL	用户界面密集系统
3	MetaH	实时、容错、安全等系统
4	UniCon	大量组件和连接件的管理
5	Rapide	基于事件的、复杂、并发、分布式系统
6	Wright	复杂交互
7	Darwin	基于消息传递的分布式系统
8	XYZ/ADL	架构逐层细化、分析、验证和自动代码的生成
9	ACME	架构在不同ADL之间变换、共享
10	XADL 2.0	实时系统、产品线架构
11	XBA	不同ADL的模型共享与协作开发
12	ABC/ADL	架构建模元素定义