<<Executable UML模型驱动 >

图书基本信息

书名：<<Executable UML模型驱动开发>>

13位ISBN编号：9787302256311

10位ISBN编号：7302256314

出版时间：2011-10

出版时间：清华大学出版社

作者：[美]Dragan Milicev

页数：591

译者：车立红

<<Executable UML模型驱动 >

内容概要

作者DraganMilicev认为，肯定可以提高软件开发效率。
在这一理念的支撑下，他提出了一种可以在软件开发中提高抽象层次并减少偶发复杂性的方法。
本书内容全面，重点讲述了可执行的UML配置文件，以及一种通过使用建模节省大量时间并避免人为错误的有效方法。

《Executable
UML模型驱动开发》是一本有关模型驱动开发的深入教程，重点讲述了信息系统，并详细描述了该系统的含义。
阅读本书后您会发现，可以通过使用对象范式、模型驱动的开发以及形式化且可执行的UML配置文件，更好地理解信息系统并实现更高效的开发。
书中提供的示例有助于演示开发过程，并且展示了如何使用UML构建信息系统。
此外，((Executable
UML模型驱动开发》还全面描述了面向对象的基本概念，进一步讨论了面向对象和信息系统开发的全面结合，以及如何处理在系统开发中面临的挑战。

主要内容
◆信息系统的特征及信息系统开发的过程模型
◆用于构建信息系统的主流技术的优点、常见问题以及其影响开发的方式
◆介绍在构建信息系统中应用的对象范式
◆在构建信息系统中很可能需要使用UML的概念及各个部分
读者对象
本书适用于分析、指定、设计、建模、开发或测试信息系统的软件从业人员。
通过学习书中内容有助于扩展他们的知识，使用可执行的uML进行模型驱动的快速应用程序开发来提高工作效率。

作者简介

米利塞维，Dragan Milicev博士是贝尔格莱德大学电子工程学院计算机科学系的副教授。
他是Serbian Object
Laboratories
d.o.o.(SOL，www.sol.rs)公司的创始人兼CTO，这是一家软件开发公司，致力于使用模型驱动的技术构建软件开发工具，并构建自定义应用程序和系统。
在构建复杂的软件系统方面，Dragan
Milicev拥有25年的丰富经验，曾在20多个学术和国际行业项目中担任首席软件架构师、项目经理或顾问。
值得一提的是他曾担任大部分SOL项目及其产品的首席软件架构师和项目经理，这些SQL产品包括：SOLoist，一个用于信息系统的快速应用程序模型驱动开发架构；SOL
UML Visual Debugger，世界上首批UML可视化调试程序之一，针对UML建模工具Poseidon而设计；SOL Java
VisualDebugger，用于Eclipse的插件，支持使用UML对象图表对测试对象结构建模。
他曾在大部分知名的科学和专业的报刊、杂志上发表论文，为模型驱动的开发和UML的理论和实践作出了巨大贡献。
Dragan
Milicev以前曾在塞尔维亚出版了3本有关C++、面向对象编程和UML的书籍。

书籍目录

章节摘录

版权页：插图：问：何时定义GUI配置？
在建模时还是运行时？
答：作为对象结构，GUI配置在建模时和运行时均可以定义。
如果是在建模时定义的，就如本章所述，是根据作为模型一部分的对象结构定义的。
模型中的此类对象结构具有形式化的、创造性的语义，而非UML对象规范一般拥有的非形式化的、说明性的用途。
GUI配置指定了应该在运行系统的对象空间内创建什么样的对象结构。
这些结构仅用GUI配置图表表示，但是由在这些图表中所描述的相应模型元素（即对象和链接规范）定义。
另一方面，该结构也可以在运行时定义。
GUI环境可以提供在运行时以交互方式定义GUI配置结构的手段。
甚至可以在系统执行期间，以编程方式创建和修改该结构。
这使得系统不需要进行任何建模工作或重新编译工作，便可以以动态方式、按需更改其外观。
通用OOIS UML运行时环境提供了这样一个特性。
问：GUI配置结构是可以定义的，甚至可以以交互方式修改它，请问是开发人员还是用户完成此工作？
如果是用户完成此工作，用户就可以修改应用程序，这样不是太危险了吗？
答：基本上，应用程序GUI的交互定义和修改的特性主要由开发人员使用。
不太可能由普通用户完成此工作。
该特性的主要目的是改进系统的灵活性，并减少开发和维护GUI的工作。
然而，如果需要的话，最终用户也可以使用这一特性。
例如，可以扩展基本的GUI配置类模型（定义GUI配置元素的内置类的模型），也可以将其与处理系统用户及其权限的概念模型合并。
这样，用户就可以（以一种严格的方式）使用该特性个性化GUI。
例如，可以授权Easyleam学校信息系统的一组用户使用Courses and Students上下文，而另外一组用户可能对使用Coursesand Teachers上下文更感兴趣。
通过将常见的对象结构作为用于自定义GUI的底层概念模型的实例化进行管理，此方法实际上提供了无限的灵活性。
然而，如果无经验、未经授权或怀有恶意的系统用户使用此特性，确实很危险。
这种情况下，应该禁止使用此特性。
禁用该特性即可，或在必要时限制该特性的使用权限，这要比提供应用程序的高级、轻松且灵活的自定义简单得多。
前面已经讲过，该特性旨在减轻开发人员的工作负担，提供更大的灵活性，并对应用程序提供更好的维护。
但是，如果不需要该特性，禁用即可。
与用传统方法开发自定义的GUI相比，这肯定要简单一些。
问：在GUI配置中可以指定哪些表现特性？
答：GUI配置结构支持的具体的表现特性集不是由OOIS UML配置文件定义的。
此定义在实现阶段完成。
描述的GUI配置方法仅是一个架构，用于定义一致、简单、非冗余且灵活的结构（此结构在一个中心位置指定应用程序GUI的外观和行为），而非将此信息分散在应用程序的代码或模型中。
然而，为允许GUI的多样性，该方法没有定义具体的特性集，而是只定义了一些基本概念。
每个GUI开发架构、方法或运行时环境都可以定义其自己的概念和特性集，从而指定GUI项设置和GUI上下文已定义的一般概念。
这样，这些特化就形成了由给定环境所支持的模型库。

OOIS UML通用GUI环境就提供了这样的一个支持，如本章所示。
在16.2节，可看到有关该主题的更多信息。

版权说明