第32卷 第18期 计 算 机 工 程 2006年9月
Vol.32 № 18 Computer Engineering September 2006
·开发研究与设计技术· 文章编号：1000—3428(2006)18—0278—02 文献标识码：A 中图分类号：TP311.52

基于 Eclipse 框架的嵌入式 IDE 实现

李连云，李 毅，温利娜，张晓先，张 激
（华东计算技术研究所，上海 200233）

摘 要：可扩展性框架为软件开发提供了统一的机制，使软件开发者能更专注于软件本身的功能和实现，从而大大缩短软件的开发周期，提高软件的移植性和扩展性。该文介绍了 Eclipse 的插件，扩展点以及插件发现机制，并探讨了如何利用 Eclipse 的可扩展性框架进行 IDE 开发，在此基础上完成了基于 Eclipse 框架的嵌入式 IDE 的实现。

关键词：框架；插件；可扩展性；嵌入式；IDE

Implementation of Embedded IDE Based on Eclipse Framework

LI Lianyun, LI Yi, WEN Lina, ZHANG Xiaoxian, ZHANG Ji
（East-China Research Institute of Computer Technology, Shanghai 200233）

【Abstract】The extensible framework provides a uniform rule for software developer, for which the developers may be absorbed in software function and implement. Therefore, the developer may shorter the cycle of development of software and improve software's transplantation and extension. The paper introduces plug-in, extensible point and the rules finding plug-in, analyses the extensible framework and discusses to develop IDE using Eclipse framework, and the embedded IDE based on Eclipse is implemented.

【Key words】Framework; Plug-in; Extensible; Embedded; IDE

随着嵌入式开发人员对第 3 方工具的利用越来越普遍，要求嵌入式 IDE 具有良好的可扩展性。Eclipse 正是基于这种需求而产生的一个基于插件的可扩展性框架(framework)平台，它为开发者定义了大部分的基础服务，使得产品开发完全可以通过插件组合而成，而不需要考虑底层平台的细节。Eclipse 这种友好的可扩展性结构对嵌入式 IDE 的开发尤为方便，基于 Eclipse 框架开发的嵌入式 IDE 可以很灵活地集成新的工具链和第 3 方工具，因为这些仅仅是开发一组插件，并不需要了解 IDE 是如何集成这些插件的，Eclipse 的插件发现机制会自动完成这些功能。

目前，基于 Eclipse 框架进行应用开发已经有很多成熟的案例，如 IBM 的 WebSphere Studio Workbench，风河公司的嵌入式 IDE Workbench 2.0 等。本文对 Eclipse 的框架进行了分析，探讨了如何基于 Eclipse 的可扩展性框架进行嵌入式 IDE 开发，并在此基础上实现了嵌入式操作系统 ReWorks 的 IDE ReDe。

1 Eclipse 的可扩展性框架

1.1 框架

在一个特定的域中，通过领域分析可构造出一个领域模型、得出领域框架。Ralph E.Johnson 对框架给出了如下定义：

(1)框架是整个系统或部分系统的可重用设计，由一组抽象构件及构件实例间的相互作用方式组成；

(2)框架是由开发人员定制的应用系统的骨架。

由此可见，框架是构成一类特定软件可复用设计的一组相互协作的类。框架规定的是应用体系结构，定义的是整体结构、类和对象的分割与协作、各部分的主要职责及控制流程。框架预定义这些设计参数，使应用开发者专注于应用功能的设计与实现，而不管其它细节。框架记录了设计决策，是面向对象系统获得最大复用的方式。较大的面向对象应用一般由多层彼此合作的框架组成。应用的大部分设计和代码来自它所使用的框架或受其影响。框架对应用最主要的贡献是它所定义的体系结构，框架的设计应尽可能灵活、可扩展。

利用框架可以快捷地建立应用，由于这些应用具有相似的结构，从而使得它们不仅具有一致的用户感观，而且很容易维护。在获得框架好处的同时，也要付出一定的代价，这代价就是失去了一些表现创造性的自由，因为许多设计决策已经由框架决定了。

1.2 Eclipse 和它的扩展点机制

Eclipse 是一个开放源代码的、基于 Java 的开发平台，它是基于插件(plug-in)的可扩展性框架，并通过扩展点机制提供了插件组合的灵活性和扩展性。

插件是一种结构化组件，同时也是运行时最小的管理单元，拥有完整的生命周期，它负责扩展点的定义并贡献扩展。一个插件可以定义零个或多个扩展点，也可以贡献零个或多个扩展，所有的扩展都是依据扩展点的。插件使用清单文件(plugin.xml)来向系统描述它自己。Eclipse 的插件开发环境(Plug-in Development Environment，PDE)是为扩展 Eclipse 的软件开发人员提供的，它允许构建与 Eclipse 环境无缝集成的工具。当然，Eclipse 框架还可作为与软件开发无关的其他应用程序的基础，比如内容管理系统等，这里，主要是研究基于 Eclipse 的嵌入式 IDE 的开发，对基于 Eclipse 的其它应用程序的开发将不作探讨。

Eclipse 的可扩展性架构通过扩展点机制实现。扩展点定义了一组规则，最小的扩展点可以仅仅定义一个 ID。扩展点

作者简介：李连云(1981—)，男，硕士生，主研方向：嵌入式操作系统及应用；李 毅、温利娜，硕士生；张晓先，高工；张 激，研究员

收稿日期：2005-12-03 **E-mail：**llybluelee@gmail.com

可以被定义它的插件扩展，也可以被其他插件扩展。一个扩展点可以对应多个扩展。定义扩展点的插件和使用扩展点的其他插件之间的依赖关系通过插件的 manifest 文件建立。平台在运行时可依据插件间的依赖关系找到它所对应的实现。

Eclipse 的核心是动态发现、装入和运行插件的体系结构。平台处理查找和运行正确代码的数理逻辑。平台用户界面提供标准的用户导航模型。每个插件可以专注于执行少量的任务。Eclipse 内核(OSGi framework+Eclipse runtime)在运行时会利用所有插件的 plugin.xml 文件在内存建立一个共享的插件注册表。

1.3 Eclipse 的可扩展框架

Eclipse(V3.0)核心包括 OSGi framework 和 Eclipse runtime 两部分，其架构如图 1 所示。OSGI framework 和 Eclipse runtime 是 Eclipse 的微内核，负责插件的加载和管理插件的整个生命周期。其中，Eclipse Platform 主要包括如下插件：SWT，Jface，Workbench，Resources Management，Help，Update，Team 和 Debug。

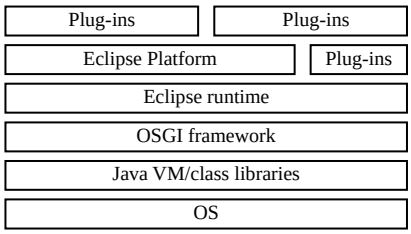

图 1 Eclipse 3.0 架构

Eclipse 的可扩展性框架如图 2 所示。从该图可以看出，JDT 和 PDE 也是作为一组插件集成进去的，这充分体现了 Eclipse 的插件集成思想。作为应用开发者，在 Eclipse 的可扩展性框架中，可以很方便地利用 PDE 来开发新的插件，并进行集成。基于 Eclipse 开发的 IDE 最大特征是几乎无限的扩展能力，它可以集成任何插件来扩展功能，也可以删除插件来去除用户不需要的功能。

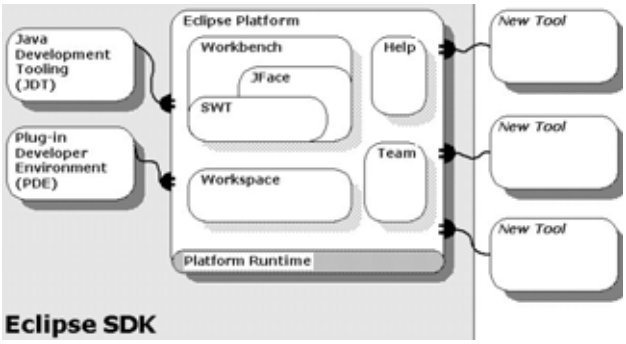

图 2 Eclipse 的可扩展性架构

2 基于 Eclipse 框架的实践

2.1 ReDe 的开发思想

在基于 Eclipse 的可扩展性框架的开发中，一般的思想是尽量利用平台提供的扩展点，然后考虑自己产品的扩展性。Eclipse 不仅自己具有很强的可扩展性架构，而且基于 Eclipse 开发的产品也有很强的扩展性。基于 Eclipse 产品的每一个功能都应该是扩展进来的，而每一个扩展都对应一个扩展点，确定好产品的功能后，就应该为这些功能设计扩展点，并且确定这些扩展点由哪些插件定义。

定义 Eclipse 扩展点，必须遵从公平竞赛法则和发散性法则。所谓公平竞赛法则就是，所有使用扩展点的开发者都必须遵从相同的规则，包括定义扩展点的作者。这些规则应该是稳定的，至少在一段时间内保持稳定；这些规则必须是开放的，对所有的开发者开放，定义扩展的作者应该公布这个扩展点的所有规则。Eclipse 几乎从来没有违反过“公平竞赛法则”，就连工作台都是作为一个扩展被加载的。

所谓发散性法则就是一个扩展点可以接纳多个扩展，如果有人在扩展点上添加一个扩展，那决不应该影响其他扩展被添加到同一个扩展点上。扩展本身决不应该考虑有没有其他扩展的存在。这两个法则使得用 Eclipse 开发产品也具有良好的扩展性。

ReDe 的开发过程采用 Eclipse 的 Team 模块进行代码管理和维护。

2.2 实现部分

本项目是为实时嵌入式操作系统 ReWorks 做一个集成开发环境(IDE)ReDe。ReDe 的主要功能是使得用户能够方便地开发 ReWorks 工程，扩展新的工程类型和 BSP，并且能够无缝集成第 3 方工具。

ReDe 项目的主体实现需要 8 个插件来完成，插件的功能见表 1。这 8 个插件定义了 ReDe 的基础部分，并能完成初步功能，为以后的扩展及第 3 方工具的集成做了统一的部署。

表 1 ReDe 主体插件功能划分

插件 ID(*=cn.com.ecict.)	插件功能描述
*.rede	产品插件，定义产品启动缺省值和欢迎界面
*.rede.core	定义 ReDe 工程特性，工具链的顶层扩展，两个扩展点(BSP 和 module)
*.rede.ui	定义 ReDe 的整体 UI，包括透视图、视图、首选项页面、工具条、菜单、运行调试的顶层 UI 部分
*.rede.doc	定义 ReDe 的帮助系统，ReDe 和 ReWorks 帮助内容
*.rede.bootable.core	定义自引导工程的特性、构建器和 module 及仿真器扩展点
*.rede.bootable.ui	定义与自引导工程相关的 UI 部分，包括自引导工程向导，module 配置的编辑器等
*.rede.libs.i386.reworks	扩展 ReWorks 的 i386BSP 和相应的 module 配置
*.rede.tool.i386.reworks	扩展 ReWorks 的 i386 工具链和仿真器

ReDe 的扩展性表现在 Eclipse 平台及 ReDe 所定义的扩展点。利用 Eclipse 提供的新建工程向导扩展点，再结合 ReDe 工程的特性很容易扩展新的工程类型。BSP 的扩展要利用 ReDe 定义的 BSP 扩展点和 module 扩展点，BSP 扩展点定义新的 BSP 扩展所要遵从的法则，使新的 BSP 和它对应的 ReWorks 库互相感知，module 扩展点定义了新的 BSP 中模块配置的法则以及这些模块配置怎样和新的 BSP 互相感知。扩展用户不需要知道扩展点定义的细节，只须按照扩展点的要求进行扩展。

ReDe 项目主体完成以后，可以开发目标机为 i386 的自引导工程。最重要的一点，ReDe 主体定义了以后扩展所必须的扩展点，这些扩展点为以后的无缝集成第三方工具打下基础。交给用户使用后，用户很方便地集成了目标机为 arm 的自引导工程的模块配置和工具链。这样，用户可以很方便地开发目标机为 arm 的自引导工程。另外工程类型的扩展也很方便，有的用户已在 ReDe 中集成了库工程的插件。

2.3 经验和教训

正如 Ralph E.Johnson 所说的那样，一个好的框架能给产

(下转第 282 页)