TURING 图灵程序设计丛书

PEARSON

OSGi与Equinox

创建高度模块化的Java系统

OSGi and Equinox: Creating Highly Modular Java Systems

[美] Jeff McAffer Paul VanderLei Simon Archer 著

郭庆 李楠 池建强 译

OSGi框架与Equinox相关技术的最佳实践指南

业界权威专家详尽解析高度模块化系统的构建过程

完善创建模块化系统知识结构的高级主题

人民邮电出版社
POSTS & TELECOM PRESS

TURING 图灵程序设计丛书

OSGi与Equinox

创建高度模块化的Java系统

OSGi and Equinox: Creating Highly Modular Java Systems

[美] Jeff McAffer Paul VanderLei Simon Archer 著

郭庆 李楠 池建强 译

人民邮电出版社

北京

图书在版编目（CIP）数据

OSGi与Equinox：创建高度模块化的Java系统 /（美）麦克艾弗（McAffer,J.），（美）万德雷（VanderLei,P.），（美）阿彻（Archer,S.）著；郭庆，李楠，池建强译. -- 北京：人民邮电出版社，2014.1
（图灵程序设计丛书）
书名原文：OSGi and equinox:creating highly modular java systems
ISBN 978-7-115-33744-3

Ⅰ. ①O… Ⅱ. ①麦… ②万… ③阿… ④郭… ⑤李… ⑥池… Ⅲ. ①JAVA语言－程序设计－研究 Ⅳ. ①TP312

中国版本图书馆CIP数据核字(2013)第276876号

内 容 提 要

本书分为四个部分。第一部分主要介绍了OSGi和Equinox，以及OSGi标准的Eclipse实现；第二部分采用非正式教程的方式教给读者如何从零开始构建真实的Toast应用，针对每一个步骤都提供了完整的在线示例代码；第三部分由原型构建转向实际的产品开发，主要介绍了OSGi和Equinox中用来构建成熟的OSGi应用必不可少的API——服务器端、声明式服务和发布过程，以及如何用它们来解决实践中的一些问题；第四部分呈现了动态性的最佳实践、整合代码库等主题，以及一些OSGi和Equinox难题，帮助读者全面理解如何创建高度模块化系统。

本书适合有Java编程基础以及对OSGi技术有兴趣的开发人员阅读。

◆ 著 [美] Jeff McAffer Paul VanderLei Simon Archer
译 郭 庆 李 楠 池建强
责任编辑 丁晓昀
执行编辑 陈婷婷
责任印制 焦志炜

◆ 人民邮电出版社出版发行 北京市丰台区成寿寺路11号
邮编 100164 电子邮件 315@ptpress.com.cn
网址 http://www.ptpress.com.cn
北京艺辉印刷有限公司印刷

◆ 开本：800×1000 1/16
印张：23.5
字数：555千字 2014年1月第1版
印数：1－3 000册 2014年1月北京第1次印刷
著作权合同登记号 图字：01-2010-5084号

定价：89.00元

读者服务热线：(010)51095186转600 印装质量热线：(010)81055316
反盗版热线：(010)81055315
广告经营许可证：京崇工商广字第0021号

站在巨人的肩上

Standing on Shoulders of Giants

www.ituring.com.cn

序

作为SpringSource的首席技术官，我经常会接触到许多正在构建企业级应用的公司，其中许多是在世界500强名单中耳熟能详的公司。我很快就发现，企业级应用领域凌乱且错综复杂。早在四五年前，采用Spring的用户都会设法向我们咨询解决方案，来帮助他们管理所构建应用的规模和复杂度。在这其中，规模庞大的团队和具有成百上千内部组件（Spring beans）的应用并不罕见。在越来越短的时间框架内开发出日益复杂的应用，企业压力与日俱增。在很多案例中，应用现今都依然保持着活跃状态，并且在不断演进。无论是来自内部规划还是外部用户需求，将软件作为“服务”交付，只能加速这种趋势。

在企业级Java领域，传统的部署单元是将一个企业级应用构建为一个Web应用归档（Web Application Archive，WAR）文件。我将讨论一些企业级开发团队面临的共同话题。

- 作为打包并部署的一个大型独立单元，WAR文件会放慢开发进程，并使得组建大型开发团队变得更加困难，因为在所有东西可以部署之前，必须用一个独立的打包步骤将它们放在一起。
- WAR文件过于庞大和笨重——一个普通的企业级应用可能会依赖几百个第三方资源，所有这些都要被打包到WAR文件中。这会对上传和部署时间带来不利影响。
- 试图通过在同一容器中同时部署多个WAR文件来减少复杂度，会导致在JVM中出现堆使用（heap usage）的问题，因为每个WAR文件都各自拷贝了所有的依赖资源，而根本不管其中很多资源在理论上应被共享。
- 一起部署WAR文件时，无法轻易共享公共服务。
- WAR文件作为最小的变更单元，意味着在大型企业级应用中，变更不能轻易被隔离和包含。
- 在设计中尝试引入自我监控（即未实施）的模块化约束通常会失败，尽管出发点是好的。

为了管控并处理开发现代企业级应用的大型团队规模及复杂需求，我们显然需要用一种更合理的方式来“分而治之”。我们需要将系统中定义良好的部分作为模块进行封装，隐藏其内部细节，并小心管理对外接口；还需要能够独立打包和部署部分模块，这样就不会迫使我们修改整个系统；另外，最好还能提供原则化的机制，可以将这些模块一起放入现运行系统中，并在不断演进的过程中将引进的变化拷贝进来。

早在2005年，面对这些需求，SpringSource（接下来是Interface21）就做出了一个简单的决定，转向OSGi——“面向Java的动态模块化系统”，将其作为模块化企业级应用的基础技术。实际上，那时OSGi服务平台已经成熟，并在一些工业环境中经过验证，同时凭借着在嵌入式系统领域内

的技术遗产，它还保持着轻量级的规模。

OSGi模块层提供了将系统划分为独立模块的机制，即bundle，它们可以独立打包、部署，并有各自独立的生命周期。这为我们解决了一部分问题——有助于保持模块私有的实现类型，只暴露组成模块部分公共接口的类型。我们当然希望企业开发者继续使用Spring开发他们的应用，通过Spring Dynamic Module的开源工程我们创建了一个简单模型，从而使每个模块都有它自己的组件集合。这些组件中的一部分是模块私有的，但有一些应公开，以供其他模块中的组件使用。OSGi服务层提供了此问题的一个解决方案，促进了一个面向内存服务的设计。模块中的组件可以在OSGi服务注册中心发布，其他模块可以从中找到并绑定那些服务。OSGi也提供了必备的基本方法来追踪服务，因为这些服务可能会随着模块的安装、卸载、升级而不断变化。

OSGi发展行程中的下一站将是引入SpringSource dm服务器：一个企业级应用服务器，它不仅构建在OSGi之上，最关键的是它也支持部署OSGi bundle集合。Spring Dynamic Module可用于任何符合OSGi服务平台规范的实现，但对于dm服务器我们必须选择一个OSGi服务平台作为其构建的基础。我们选择在Equinox上构建，Equinox是OSGi服务平台在Eclipse上的实现，也是OSGi核心规范的参考实现。Equinox的开源特性与我们自己的开源理念非常吻合，让我们能够与Equinox开发者紧密合作，不断提交补丁和变更需求，这是非常宝贵的。Equinox的广泛使用（这只是Eclipse的一个基本插件的发展境遇，却颇具代表性）给了我们很大信心，它必会勇于进取，终将适用于企业级开发。

就我所知，大大小小的公司都对OSGi抱有与日俱增的浓厚兴趣。OSGi能为应用的模块化提供坚实的基础，反过来又有助于你构建更加高效的团队。能将复杂的应用分成各个独立部分，也就能较好地构建并划分团队职责，“组织跟随架构”的意义即表现于此。在其他场景中，团队可能是固定的，现在你可能会需要用一种架构来让他们最为有效地合作。同样，将系统划分为模块的原则性基础也会有助于此。将OSGi作为基础，打包和部署的单元将会变为一个独立的模块，这会消除处理过程中的瓶颈，同时使变更的影响减到最小。OSGi也非常适合产品线的建造，而当你为了能让第三方扩展你的软件而需要向其提供扩展或插件机制时，它也是非常适用的。

OSGi前景十分光明。规范的4.2版本已经发布，OSGi核心平台和企业专家组非常活跃。浏览一下OSGi联盟和专家组的成员组成，你就会明白企业供应商们对OSGi有多么重视。我相信阅读和学习本书所花费的时间一定会物超所值。我坚信OSGi是大势所趋。全面了解OSGi服务平台的优缺点，将会在创建灵活的模块化软件过程中，为你带来无限价值。

——Adrian Colyer
SpringSource首席技术官
2009年10月

前　言

OSGi是当前的一个热门话题，所有主流的Java应用服务器提供商都选择了OSGi作为他们的基础运行时，Eclipse至少在过去5年内一直使用OSGi作为对于模块化的基本描述与基本运行时，而不计其数的系统也在嵌入式和“界面后台”场景中使用了它。所有这些对OSGI的应用都有着充足的理由。

Eclipse作为工具平台的成功是OSGi所尊奉的强大模块化特性导引的直接结果。它使开发者能够游刃有余地处理变更，让团队变得更加灵活，组织得以改变软件开发方式，进而促进生态系统的实现和运行。这些优点也同样可以在其他任何软件领域内实现。

OSGi的主要规范非常简洁——只有27个Java类型。它设计精良，特别适于在实践中的实现与使用。不过，选择OSGi也并非没有任何挑战：毫无疑问，实现高度模块化和动态的系统是非常困难的。俗话说得好，天下没有免费的午餐。有人已经指出OSGi复杂晦涩。在多数应用场景中，复杂是主要问题——想要模块化或动态性，就会出现这样的问题，但这不是原因。模块化处理已有的单层系统尤其具有挑战性。

本书会着重强调这些主题并提供相关的知识、指导和最佳实践，以缓解OSGi的复杂性。我们重点讨论模块化、组件和动态性，并向你展示强化系统灵活性和敏捷性的技术。

尽管我使用OSGi已经很多年，参与过OSGi规范的撰写，也实现过Equinox（OSGi框架规范的参考实现），但在撰写本书的过程中，关于OSGi、Equinox和高度模块化的动态系统，我依然收获颇多。我们相信读完本书，你也定会大有收获。

关于本书

本书通过一个基于OSGi的远程信息管理和车辆管理系统示例Toast，为即将使用和已经使用OSGi的开发人员详细介绍了开发的所有步骤。

我们从零开始，将Toast开发成一个功能完备的客户端和服务器端系统。对于大部分开过车或运送过包裹的人们来说，这个领域是非常熟悉的。简单来说，远程信息管理涉及汽车中全部的电子设备——无线电设备、导航、车内的气候控制系统，等等。车辆管理系统则涉及对包裹和车辆在位置变更过程中的追踪与协调。

其中出现的一系列问题和机会使你接触到各种各样的主题：从模块化和组件协作，到服务器端的编程和打包，以及高度模块化系统的交付。我们创建了单独的客户端应用，单独的嵌入式服

务器端配置，并对两者进行了动态增强。本书的这些知识和经验完全可以移植到你所在的软件领域中。

本书大致分为两部分。前半部分，即第一部分和第二部分，为介绍OSGi和Equinox做好铺垫，并提供了一份构建Toast的教程，逐步介绍如何将Toast构建成一个具有很多高级功能的车辆管理系统。教程内容撰写方式通俗易懂，不至使人彷徨困惑。另外，还分享了一些在开发应用和撰写教程的过程中，我们所经历过的陷阱和意外。

本书后半部分侧重于如何将该教程内容变成现实。编写原型和开发产品完全是两码事。为了使你脱离原型阶段，第三部分用了多个章节深入探究了完成这些工作的细节——改善和重构首个原型、定制用户接口、构建并交付产品。本部分可作为参考阅读，但仍介绍了少量的步骤示例和代码示例。我们想既深入研究，又要把大部分在社区中已报告的，以及在我们自己开发的专业产品中发现的主要问题都囊括进来。

最后一部分，即第四部分，则是单纯的参考资料。它涵盖了OSGi和Equinox的基本方面，并涉及了前面章节所没有涵盖的各种功能。我们也讨论了一些最佳实践和高级主题，例如集成第三方代码库和具有动态性。

OSGi尽管相对较小，但是非常全面。正因如此，单靠一本书不可能涵盖所有可能的主题。我们聚焦的功能和服务，都是我们在每天的开发过程中和系统中使用的，这些也许会在你的开发过程中起到一定的作用。

OSGi、Equinox和EclipseRT

OSGi社区非常有活力。至少有3个活跃的开源框架实现社区，有着许多使用者和扩展者。本书的绝大部分内容涵盖了对任何OSGi系统或实现可用的通用主题。在本书中，我们一直使用Equinox作为我们示例和讨论的基础，它是OSGi框架规范的参考实现。有时，我们会介绍一些只在Equinox下可用的特性和工具。通常，这些功能已经被加入Equinox来解决现实问题，而这些问题你也可能会遇到，所以有必要在此进行简要介绍。

贯穿本书，我们也介绍了编写和构建OSGi bundle的Eclipse插件开发环境（PDE）工具。PDE是综合的、有效的、成熟的工具，已经在OSGi环境中使用多年。如果你不使用PDE来创建基于OSGi的系统，或许你可以借此机会反省一下。

最后，Eclipse是工具领域的动力之源。它越来越多地被用于纯运行时、服务器端和嵌入式环境。这些工作已经融入到EclipseRT中。EclipseRT包含了许多在Eclipse下开发的技术，它们针对（或有益于）典型的运行时环境。这里开发的Toast应用已经捐献给Eclipse Exapmles工程，并成为EclipseRT技术的示例。我们鼓励你从http://wiki.eclipse.org/Toast上签出代码，以便更好地了解人们如何开发Toast，以及用Toast能做什么。

读者对象

本书面向一些Java开发者群体。你必须拥有一些Java编程经验，我们不会介绍Java的概念和语法。

对于OSGi和Equinox的新手，书中会介绍这些技术的起源，如何使用Eclipse OSGi bundle工具，以及如何创建第一个基于OSGi的系统。读者最好之前用过Eclipse，但这并不是必须的。

针对有开发OSGi bundle和系统经验的开发者，本书正式介绍了有助于创建使用OSGi的高度模块化系统的诸多技术和最佳实践——从服务协作方式，到服务器端的集成和作为发布工程组成部分的系统构建，以及部署和安装。

对于有经验的OSGi开发者，本书详细介绍了一些在Equinox中可用的特殊功能，并全面涵盖了一些有用的工具，例如声明式服务、buddy类加载、Google地球集成，以及一些能够使基于OSGi的系统在设计、编码与打包等方面比以前更容易的Eclipse bundle工具。

示例代码

在阅读本书的过程中，读者可以亲自试验。很多时候，可以按照书中的步骤进行试验，也可以编写自己的代码。本书附件下载中包含了每章的代码。获取和管理这些示例的指令在第3章中有所提及。通常，所有需要的资源都可以从http://eclipse.org或http://equinoxosgi.org在线获取。正如前面提到的，Toast的简易版依然存在，成为了Eclipse的一个开源工程。具体参见http://wiki.eclipse.org/Toast。

排版约定

本书使用下面的格式约定。

粗体：用于UI元素，诸如菜单路径（例如**File>New>Project**）、向导和编辑器中的元素。

楷体：用于强调和突出术语。

`Lucida`：用于嵌入在文本中的Java代码、属性名称、文件路径、bundle ID，等等。

`Lucida`粗体：用于突出代码示例中的重要行。

备注和扩展内容经常用于强调那些读者可能感兴趣，或者有助于读者使用或理解正文所述功能的信息。我们努力营造非正式的结对编程的效果，好像你与别人相伴而坐，随处都能获得及时的技巧提示。

反馈

本书的官方网站是http://equinoxosgi.org。附件信息和勘误表位于informit.com/title/0321585712。读者可通过book@equinox.org向作者报告在本书或代码示例中发现的问题和错误。我们非常期待你的改进建议和反馈。

关于作者

Jeff McAffer是Eclipse RCP和Equinox OSGi工程的共同领导，是EclipseSource的CTO和共同创办人。他是Eclipse平台的架构师之一，也是*The Eclipse Rich Client Platform*的作者之一。他共同领导了RT PMC，是Eclipse Project PMC、Tools Project PMC和Eclipse 架构委员会成员，曾担任Eclipse基金会的董事会成员。Jeff目前致力于Eclipse组件的所有方面，从开发和构建bundle，到部署、安装及最终的运行。他曾经担任IBM高级技术专家，是国际对象技术组织（Object Technology International，OTI）的一个团队主管，工作涉及Smalltalk、分布式/并行OO编程、专家系统和元数据架构。他在东京大学获得博士学位。

Paul VanderLei是Band XI International的合伙人。他有超过25年的软件工程经验，着重于面向对象设计和敏捷实践。他那解决复杂问题时简单却富有创意的工程方案令他闻名遐迩。在美国亚利桑那州大学获得计算机科学硕士学位后，他加入了OTI，并广泛参与了基于Smalltalk的系统。IBM收购OTI后，Paul作为IBM的Embedded Java Enablement团队初创成员，针对汽车和医药领域，开发了嵌入式Java应用和用户接口。他已经在商业应用中使用OSGi 10多年了。他与家人生活在密歇根州的大急流城（Grand Rapids）。

Simon Archer有超过16年的软件工程经验，着重于面向对象设计、敏捷实践和软件质量。在英国朴茨茅斯大学获得计算机科学学士学位后，他作为一名Smalltalk开发者，就职于Knowledge System公司，后来在OTI工作。2000年在OTI时，Simon开始在远程通信和RFID等领域使用并教授OSGi技术。现在他致力于IBM Rational软件的研发，使用OSGi为Jazz Foundation工程构建协同的开发工具。他与家人现居北卡莱罗纳州的Cary小镇。

致　谢

没有众人的合作与帮助，我们就无法成就一本书。本书中，整个Equinox团队几乎全程给予支持，包括交谈、帮助编写代码、定义概念、bug修正、审查手稿或一般性的支持。

对于本工程，有些人贡献了大量时间和脑力，我们在这里对他们表示由衷的感谢。

Tom Watson：Tom是Equinox背后的驱动力，他在OSGi规范社区非常活跃。他务实的方式和冷静的头脑为你带来了Equinox，为我们编写本书提供了支持。

Chris Aniszyack：Chris带来了他对PDE的火样的热情，这一工具使得OSGi和Equinox变得易于编程。本书的编写带动了许多新的用例和需求。Chris热切地推动PDE，使其更加适合bundle开发环境，这让我们所有人的生活变得更加轻松。

Ian Bull：Ian应用了他的教育学技巧，关注了所有事情的细节，涉及p2、打包、构建，使构建和发布OSGi功能的整个过程更易管理。

Stoyan Boshev：Stoyan是Equinox声明式服务实现的得力助手。DS在书中和示例代码中占据了重要篇幅。Stoyan花费了大量时间实现DS，并和我们一起努力将它的功能展示给读者。

许多人提供了本书部分示例代码或做了深入的代码审阅，或对书中的内容做了技术指导。尤其是DJ Houghton和Scott Admiraal完整测试和审阅了指南部分，在此过程中给了我们很大帮助。Rafael Oliveira Nóbrega和Chris Aniszyzck对创建声明式服务工具做了巨大贡献，使得DS人皆可用。Andrew Niefer、Pascal Rapicault、Simon Kaegi和 Scott Lewis主要贡献于修正、示例和技术指导，范围从PDE 构建、服务器端OSGi p2到ECF。Patrick Dempsey贡献了Crust代码，并孜孜不倦地做了Mac相关的所有支持。BJ Hargrave是OSGi的坚定领导，他耐心地讨论了所有设计点、最佳实践和编码方式。

我们也有幸找到Eclipse社区和一些人审阅了本书，并提供了有价值的建议和帮助。他们是Joel Rosi-Schwartz、Benjamin Muskalla、Kevin Barnes、Grant Gayed和SWT团队、Ralf Sternberg、Matt Flaherty、早期草稿的读者Rough Cuts，以及所有涉及开发Toast示例代码的人们。

当然，没有哪本书可以离开出版团队。我们非常有幸，Greg Doench和Michelle Housley、Barbara Wood、 Elizabeth Ryan以及Addison Wesley整个团队持续担任了Eclipse 系列的编辑，这使得整个过程轻松快乐。

作者将单独致谢如下。

Jeff McAffer：Nancy、Sydney和Toby，你们是我生命中的至爱。妈妈、爸爸和Val，我深深地爱着你们！因为你们我才有了今天，对此我表示由衷的感谢。整个EclipseSource团队，谢谢你

们给我发挥的空间，以及对Toast和此工程的满腔热情。

Paul VanderLei：我将感谢在Band XI International的搭档——John Cunningham、Brett Hackleman、Patrick Dempsey和James Branigan，他们慷慨地为我提供了时间来完成此工程。同样感谢我的妻子和孩子们，谢谢你们的耐心和爱。最后，我永远感谢我的父亲，他的鼓舞和孜孜不倦的指导已经影响了我的整个职业生涯。

Simon Archer：承担起本书的撰写意味着花费大量的时间、奉献和牺牲。我要感谢合作者Jeff和Paul对此工程花费的时间和所做的贡献。对于我的妻子Lisa以及孩子Thomas和Emma，我表示最真挚的感谢，因为你们做出了所有的牺牲。谢谢你们永恒的支持与爱，允许我在本书上投入时间。对此我永远感激不尽。

在本书之外，如果没有下面这些人就没有OSGi的今天。

BJ Hargrave：BJ是OSGi联盟的CTO，他一直致力于促进OSGi技术的发展。他是IBM OSGi实现SMF的负责人，SMF作为Equinox的先驱已经捐献给Eclipse。他不断推动和指导OSGi脱离原有领域演变发展。

Peter Kriens：Peter是OSGi的传道者和OSGi社区的长期领导者。他充满激情地积极推动OSGi的传播发展。在OSGi规范中我们看到的连续性和清晰性，均是Peter编辑和设计技巧的直接体现。

Tom Watson：Tom是Eclipse中Equinox OSGi工程的共同领导者和关键决策者，是OSGi专家组的重要成员。他负责整个框架实现和许多插件工具。他的实用主义和全面的思想塑造了Equinox的今天。

Richard Hall：Richard是Apache Felix工程的领导，他积极参与OSGi规范的过程。Felix是Oscar工程的演变，是第一个开源OSGi框架实现，也是Equinox团队决定选择OSGi的灵感所在。Felix工程提供的可选择的观点持续地丰富了规范和实现过程。

目　录

第一部分　简介

第二部分　OSGi 示例

第三部分 进阶篇