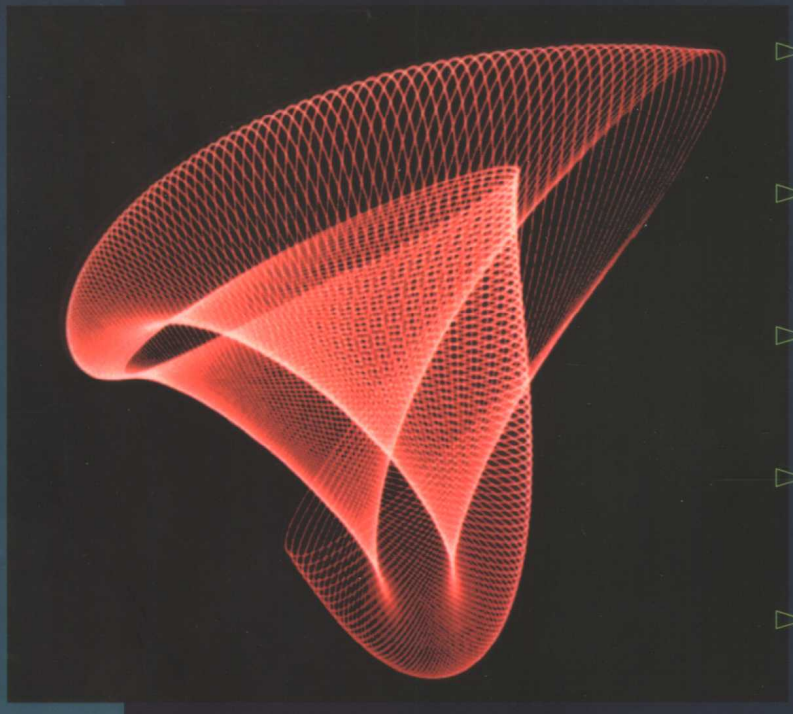
Mc Graw Education

Practical J2EE
Application Architecture

实用 J2EE
应用程序体系结构



- 使用用例驱动型过程构建大型 J2EE 应用程序
- 运用三个引导性的模式学习 Struts 实现语义
- 在一个基于MVC的体系结构中实现最佳设计模式
- 从需求到实现开发一个端对端的解决方案
- 使用 J2EE 组件开发 Web 服务

(美) Nadir Gulzar 著
陈晓燕 丁炎炎 译

Mc Graw Hill
清华大学出版社

实用 J2EE 应用程序体系结构

(美) Nadir Gulzar 著

陈晓燕 丁炎炎 译

清华大学出版社

北 京

Nadir Gulzar

Practical J2EE Application Architecture

EISBN: 0-07-222711-7

北京市版权局著作权合同登记号　图字：01-2003-3808

本书封面贴有 McGraw-Hill 公司防伪标签，无标签者不得销售。

图书在版编目(CIP)数据

实用 J2EE 应用程序体系结构/(美)盖尔兹(Gulzar,N.)著；陈晓燕，丁炎炎译.—北京：清华大学出版社，2003

书名原文：Practical J2EE Application Architecture

ISBN 7-302-07571-9

Ⅰ.实… Ⅱ.①盖…②陈…③丁… Ⅲ.JAVA 语言—程序设计 Ⅳ.TP3112

中国版本图书馆 CIP 数据核字(2003)第 100863 号

出 版 者：清华大学出版社　　**地　　址**：北京清华大学学研大厦

http://www.tup.com.cn　　**邮　　编**：100084

社 总 机：010-62770175　　**客户服务**：010-62776969

组稿编辑：曹康

文稿编辑：于平

封面设计：康博

版式设计：康博

印 刷 者：北京市清华园胶印厂

装 订 者：三河市李旗庄少明装订厂

发 行 者：新华书店总店北京发行所

开　　本：185×260　**印张**：19　**字数**：486 千字

版　　次：2003 年 12 月第 1 版　2003 年 12 月第 1 次印刷

书　　号：ISBN 7-302-07571-9/TP・5570

印　　数：1～4000

定　　价：39.00 元

绪　论

如今，每个人都可以访问 Internet，连我们的孩子们很早就开始接触 Internet，它对年龄是没有限制的。我们可以方便快捷地找到任何信息。一想到总是可以浏览其他一些网站，我们就对一些慢吞吞的网站失去了耐心。对于需要的信息，您的耐心指数是多少？一般而言，人们仅在 4 秒钟之后就开始变得不耐烦了！现在信息技术已经在我们的工作中普及，而我们将同样的期待带到了 Web 体验上。我们开始抱怨工作系统中那些静态的内容，希望它们被个性化，并且在使用它们的过程中可以学到知识。不过，最希望的是我们不需要记忆和重复输入如此繁多的口令。

高级技术人员也许会说："这些都很简单——您所需要的只是一个 Portal 系统，这个系统拥有个性化能力，活动管理，支持业务逻辑的可靠应用程序服务器，单点登录安全性以及一些集成技术。"这是千真万确的，就好像您只需要一座大坝来控制中国的长江一样，它们将极其复杂的细节都忽略掉了。

在航空业，每当我们踏上飞机时，我们都应该感激人们所做出的艰苦分析以及实现和测试阶段前的设计工作（当然，我们并没有想到感谢，甚至没有想过使得巨大的"金属管"升天的物理原理，也没有想过使飞机可以飞行的相互作用的系统—— 也许还是不要去想为好）。航空工程师必须做出正确的决定，生命是他们的赌注。即使我们可能会认为商业软件不需要经历如此严格的开发过程，但事实上与之相差不远。毕竟，有许多需要执行的操作，尤其在金融方面，如果系统瘫痪将导致每天数百万美元的损失。即便是在稍小的软件规模中，如果您的竞争对手拥有一个能够更加灵活地满足客户需求的系统，那么您的业绩就要吃紧了。

因此我们遇到了难题。用户对我们开发的系统的要求非常高（可以说是越来越高），这就加速了系统本身对性能的要求。而与此同时，当今的焦点是投资收益（ROI，Return on Investment），强调金钱的价值，市场效率和转变的灵活性。那么我们怎么解决这个问题呢？非常简单，您只需要进行谨慎的设计。根据我们的观察，这类笼统的句子会令人误解而且会使设计过程变成很棘手的工作。但是最后，您的系统将更好。您很可能需要避免范围蔓延（在开发时改变项目计划，加入新的要求），最好是在客户需要时为他们作改变。

本书所要讲述的内容是了解客户需要什么以及在 J2EE 架构中呈现这些要求。企业应用程序的开发是非常艰巨的任务，所以了解当前的相关材料对开发将具有指导意义。本书努力说明所有技巧中最难的一部分——以简单的方式解释复杂的论题，便于您了解与工作相关的问题。毕竟，您可能刚接触这些技术或者对它略知一二，需要在尽可能短的时间内达到熟练。

以后当您在线查询账户收支余额或者给服务台打电话时，可以根据您的经验估计一下您的期望值，看看他们的系统设计制作得如何？

Simon Rowell
技术管理部主管
US 西部全球联盟
BEA 系统有限公司

前　言

开发大型企业应用程序涉及到一些过程和技术，包括创建真正可扩展、可维护及可伸缩的面向对象的体系结构。为了使应用程序既能够满足提出的业务目标，又能够连续使用数年而不需要进行重大的重新设计，必须保证应用程序是一个开放的、灵活的、标准友好的体系结构。对问题域和客户需求进行详细的阐述只是确保所创建的体系结构能够满足业务当前和将来需要的一部分工作。问题域的定义也必须成为指导设计的方法和开发过程。我们应该由设计和开发产品追溯到最初的要求，以便保证声明的要求与实际交付结果之间的一致性。这种要求和其他与项目相关的产品之间的可追溯性确保了系统的设计视图与其用例视图的一致。

创建系统的用例视图是一个十分仔细的过程，信息体系结构在这个过程中也起了重要作用。将用例视图转换成相应的多层体系结构的设计视图需要进行渐近性、迭代式的域建模过程，业务层处理建模，实现表示语义——这些都将包括在设计的上下文中，该上下文将允许在应用程序的所有层中进行最大数量的重用。当在这个纲要中加入其他条目，如需要理解基本的组件技术并且需要按照管理项目生命周期的开发方法学和过程进行操作时，可以看出，创建一个用于验证大型解决方案的原型应用程序需要学习极多的内容。虽然我们刚才讨论的解决所有不同方面的问题所需的处理和技术在许多书中都有详细的介绍，但是，让所有的 J2EE 狂热者在能实际创建端到端的解决方案之前通读每本都是几百页的书是不实际的。连续学习软件开发的每一原则的效率也不高，因为这会占用更多时间，而且我们无法将暂时不用的信息保留很长时间。本书使用“学以致用”的方法，为大部分基于 Web 应用程序的基本体系结构的创建提供了蓝图。这种方法为 J2EE 狂热者提供了机会，使他们可以很快地使用所学的概念解决现实世界的问题。它还为有志成为体系结构设计师的开发人员和学生提供了下列内容：

- 用例驱动型建模和开发
- 用例细节中信息体系结构的角色
- 阐明应用程序中的安全策略
- 理解表示框架在 MVC(模型-视图-控制器)体系结构的上下文中的需要
- 使用 Struts 框架
- 确定使用一致的方式启用 Struts 的实现模式
- 运用类图表示设计视图的静态方面，用序列图表示其动态方面
- 使用最好的实践设计模式为表示层组件之间，业务层组件之间以及中间层组件之间(在表示层和业务层之间)的交互关系建模
- 使用 J2EE 组件技术实现设计视图
- 使用 J2EE 组件技术实现 Web 服务

0.1 本书读者对象

本书的第 I 部分对体系结构设计师，开发人员，项目经理，质量评估小组，信息体系结构设计师以及任何其他要了解需求分析过程的人很有益处。本书其余部分适合想在 J2EE 平台上构建企业级业务应用程序的初级体系结构设计师、团队开发人员和学生使用。本书第 II 部分要求读者熟悉面向对象的原理及概念，并且最起码要了解 UML(统一建模语言)。本书假定读者已熟悉基本的 J2EE 概念和开发，以及 J2EE 组件的开发。

为项目创建参考体系结构或者原型编程模型的技术团队。将从示例应用程序中得到数个通过无数次实践检验的 J2EE 设计模式，以及它们的交互和依赖。示例应用程序为基于项目特定的要求来开发编程模型提供了很好的起点。那些需要了解表示层架构体系结构及其使用原理的技术团队将从对 Struts 的讨论中获益。

本书重点强调体系结构和设计而不是强调编程方面等内容，因为这不是一本囊括所有关于 J2EE 平台或者 Struts 框架的书。本书对设计模式的介绍也并非详尽，因为设计模式的相关主题较多，也有很多书籍介绍设计模式，同时还有很多专门讨论设计模式的网站。本书中实现的 Web 服务基于 BEA WebLogic Workshop，它提供基于 JAX-RPC API 的抽象，诸如此类的使用 JAX-RPC API 进行的编程将不在本书中作介绍。

0.2 如何使用本书

由于本书的编写是连贯性的，所以最好从头至尾阅读本书。本书第 1 章首先开发了示例应用程序 GreaterCause 的用例，然后在第 2 章中讨论了信息体系结构对开发用例的影响。第 3 章是可选读的，但是我们鼓励读者浏览一下应用程序体系结构部分，因为它为学习本书其余部分打下了基础。如果您已经熟悉了 Struts 和相关的体系结构，那么可以跳过第 4 章。第 5~7 章为应用程序的每一层建立组件，同时实现了相关的用例。第 8 章使用第 6，7 章开发的组件实现了 Web 服务。请注意第 8 章的 Web 服务的实现是基于 BEA WebLogic Workshop 的。第 9 章提供了安装和练习示例应用程序的信息。如果您选择 WebLogic 作为应用程序服务器，那么我们在第 9 章还将循序渐进地指导如何安装 WebLogic Server 7.0 以及部署和实践示例应用程序。

0.3 相关 Web 站点和下载

附带二进制库源文件、文档和勘误表链接的示例应用程序可从 http://www.osborne.com 上获得。请按照 Osborne Media 提供的说明找到本书的具体链接。关于本书源文件的发布请参考源文件下载包。要获得下载包的全部信息，请参考第 9 章。

0.4　本书主要内容

第 I 部分(第 1~3 章)——需求和体系结构定义

第 1 章——“使用用例进行需求分析”以系统用例视图的形式解释了定义问题域的过程。示例应用程序被分解成独立的功能单元，而每一个功能单元又被描述成独立的用例。每个用例使用标准模板进行解释，这些模板从与用例交互的外部实体的角度解释系统行为。用例视图对于业务域专家、应用程序设计师和开发人员对系统行为有一个基本了解是十分重要的，而这里不指定如何实现该行为。示例应用程序的用例视图的开发是理解其他章节的前提。

第 2 章——“用例细节的信息体系结构”说明了信息体系结构对概括定义用例的影响。在本章中，我们将通过更加明确地解释系统与用户的交互以及相关联的交互语义，详细描述示例应用程序的用例。信息体系结构对组织，标注，导航，索引和搜索内容的模式设计是至关重要的。这些方面在创建系统 UI 的原型视图时会被汇集到一个情节板中。使用站点流程图说明了应用程序的导航语义，它明确地表示了与用户动作有关的页面转换，这部分信息将在配置 Struts 架构时用到。

第 3 章——“应用程序的体系结构、安全性和缓存”介绍了应用程序体系结构的重要方面，因为这些适用于 J2EE 平台(但是示例应用程序的实际体系结构是使用用例驱动型方法一步一步建立起来的)。本章讨论了安全性，在突出技术和规范的上下文中作了高度概括，这可以帮助读者确定他们独特的安全基础构造的需求，最终选择最好的解决方案。本章同时介绍了基于 Project Liberty 体系结构的联合网络身份。本章结束部分讨论了缓存，解释了常见的缓存解决方法并说明了基本的缓存体系结构。

第 II 部分(第 4~9 章)——设计和构造

第 4 章——“基于 Struts 的应用程序体系结构”讨论了对基于 MVC(模型-视图-控制器)体系结构的表示层框架的益处及其设计考虑。本章还讨论这种框架在 Struts 上下文中的重要方面。我们将从 Struts 的不同方面研究 Struts 体系结构，其实现和配置语义及基本使用方法，以便读者快速熟悉 Struts。书中的内容提供了读者关于 Struts 的基本知识，使读者能够具有必要的背景知识来衡量其在问题域中的可应用性。本章提供的内容足够让读者理解第 5 章中的用例。

第 5 章——“表示层的设计与实现”主要讨论了示例应用程序中表示层功能的用例实现。重点就是在使用最好的 J2EE 实践设计模型实现客户端语义时，创建了系统的静态和动态模型。本章也确定了 Struts 的实现模式，提供了解决复杂用户交互的可重用解决方案。我们可以从这些模式派生出模板，帮助开发小组建立一致的设计词汇表，以及跨所有用例的实现，从而提高了代码的可读性和可维护性。今后将以这些模式为起点继续开发。

第 6 章——“域模型的设计与实现”重点介绍了创建域模型和永久保留域对象的相应数据库模式。在本章中，我们将要标识域实体及其关系。我们使用 J2EE 容器服务对域实体及其关系进行访问和保存。本章同时介绍了容器管理持久性的实体 bean 的容器管理域和容器管理关系的配置问题。本章实现的域模型构成了第 7 章中实现业务层组件的基础。

第 7 章——“业务层的设计与实现”主要介绍示例应用程序的业务层功能的用例实现。本

章讨论并实现了一些最好的业务层设计模式。本章的重点是在问题域的上下文中决定适当的设计模式，以及这些模式的应用程序用来解决业务层的设计和开发时遇到的常见问题。除此以外，本章还介绍了有状态和无状态的会话 bean 的配置以及与 Enterprise JavaBean 有关的事务语义。

第 8 章——“使用 Web 服务进行应用程序集成”介绍了 Web 服务技术及其相关的标准。其中提到了 WSDL(Web 服务描述语言)和 SOAP(简单对象访问协议)规范的一些重点，以便读者理解 WSDL 结构和相应的 SOAP 消息结构之间的关系。本章所学的概念将在使用 BEA WebLogic Workshop 的示例应用程序的上下文中创建 Web 服务时使用。

第 9 章——“应用程序的装配与部署”主要介绍如何安装和配置 WebLogic Platform 7.0 以及如何部署 GreaterCause 示例应用程序。

目　　录

第 I 部分　需求和体系结构定义

第 II 部分　设计和构造

第III部分　附　录

第Ⅰ部分　需求和体系结构定义

第1章　使用用例进行需求分析

本章主要内容：

- 用例驱动模型
- 定义问题域
- 标识系统上下文
- 标识风险因素和依赖性
- 标识用例包
- 归档用例
- GreaterCause 用例概述
- 小结

开发大型软件解决方案提醒我们，不同的风险承担者对最终产品有许多不同的观点。在最初阶段，没有一个具体的实体可以从语义和结构上真实地描述最终产品。在项目的该接合点处，我们对所开发的软件有一个抽象的看法。正因为这样，我们必须找到一个所有风险承担者都同意的共同基础。如果没有这个共同的基础，我们就会承担这样的风险：即创建的产品往往只能为某个特定利益的团体服务。最终的产品要满足该组织的业务需求以及所有用户和发起人的需要。因此，我们必须提供一个需求词汇表，该词汇表可以很容易被所有的风险承担者理解。本章的重点是帮助读者利用用例、活动图和事件流程来创建这样一个词汇表。

然而，在开始之前，我们需要预测本章中的产品(artifact)对定义一个项目需求的影响程度。用例是我们研究的重点。用例可以在不同的抽象级别上创建，用例图可用于给一个完整的系统、子系统或一个类的行为建模。如果没有很好地理解系统需求，而在最初的迭代过程中采用太多细节，那么就会导致许多重复性工作。因此，保持一定程度的抽象化，从业务领域专家、项目发起人、终端用户、消费者和执行经理的观点中清楚捕捉需求是很重要的。我们将这个群体统称为风险承担者(stakeholder)。对于这个群体，我们应当避免从这些风险承担者中太早、太快地获得太多信息。您会发现，通过几次迭代可以达到高级需求。这是很平常的事情，因为需求定义的过程是不断进化的，每一次迭代我们都有机会发现问题并改善之。需求小组由风险承担者和一个或多个技术员工组成。用例是这两个群体之间的一个约定，所以这个小组必须由双方中适当的代表来组成，这一点对于项目的成功是很重要的。该项目的特别需求可通过在需求小组中增加适当的技术人员来实现；例如，如果这个系统要和外部的一个 CRM(客户关系管理系统)解决方案连接，那就可以在这个小组中增加一个熟悉 CRM 空间和 CRM 软件的人。

我们提出的另一个观点是，在整个过程中都要考虑重用和分解。这种思考模式可以帮助我

们将共同的行为纳入用例，最终将一系列相关的用例打包到子系统中。第二章是本章逻辑上的深入，它将帮助我们利用信息体系结构(information architecture)映射出这一章的需求，信息体系结构可以将一个最终产品的原型提供给风险承担者和开发人员。用例将会在信息体系结构中详细描述出来，因此我们对完整捕获功能需求的努力将会在第 2 章中给出。用例的实现在第 5~8 章中给出。

1.1　用例驱动模型

统一建模语言(UML，Unified Modeling Language)的用户手册是这样定义用例的：

用例指定了一个系统或者系统中一部分的行为，它是一组动作序列(包括变体形式)的描述，系统通过执行这些动作为参与者产生一个可观测的结果。

从与用例交互的实体来看，用例就是这个系统的一个外部视图。它用于捕获系统的需求。用例不是原子的；一个代表复杂系统行为的用例可以进一步分解为更多的用例。用例需要在业务领域专家、应用体系工程师和开发人员之间创建该系统行为的一个共同理解，而不指定这个行为如何实现。在设计阶段，用例是通过一系列相关对象共同传递该用例规定的行为来实现的。设计期间建立的模型必须能够通过它们的能力映射出这些需求，从而满足问题域中的每一个用例。因此，用例有助于验证体系结构。在一个迭代设计和开发过程中，用例能够在生命周期的早期捕获需求的偏差；为了在整个项目生命周期中准确反映系统的目的，所有模型和项目产品都是同步的。这样，风险就会在这一过程中被更早地识别出来，从而防止以后重做大量的工作。

用例归档了系统，并且构成了用户接受、集成、回归和系统测试的测试案例的基础。这种方法嵌入了可追溯性，因为所有的设计、开发和测试都是在用例场景(scenario)的基础上执行的。用例变为业务单元和 IT 组织之间的一个约定。利用增量和迭代的方法，这个约定在开发生命周期中根据用例规定的行为对中间产品进行验证而得到加强。用例模型是所有分析和设计产品以及项目计划的中心。

注意：

本书一直在强调其名字中的“实用”一词。因此，本书中的每一个概念都要用假设的 GreaterCause 应用程序来解释。本章讨论的用例将会为以后理解问题域打下基础。通过本书后面部分中对体系结构和设计产品的解释，大家将逐渐认识用例。

本章接下来的部分将通过示例应用程序解释什么是捕获系统需求、为什么要捕获系统需求、什么时候捕获系统需求和如何捕获系统需求。本章以下部分说明了为示例应用程序所创建的产品。

- GreaterCause 的系统定义
- GreaterCause 上下文图和参与者
- GreaterCause 风险因素
- GreaterCause 的依赖性
- GreaterCause 用例包
- GreaterCause 用例概述

熟悉上述的结构将帮助您将围绕产品的注释与项目产品区分开。

1.2　定义问题域

为了促进理解问题域，我们提出了许多问题，其中一些问题如下所示：

- 软件要解决的业务需要是什么？
- 这个系统的用户是谁？
- 这个系统需要支持什么功能？
- 不同子系统之间的交互是什么？
- 这个问题域中的哪些组件是可以由第三方提供的现成组件？
- 这个系统的哪些组件可以分离出来作为可重用的、自包含的子系统？

这些问题以及其他无数问题的答案将会帮助我们理解解决方案空间。随着本书的深入，我们将会逐一回答这些问题。

理解问题域的第一步是创建一个项目描述。一个项目描述应该解释这个项目的用途。该描述必须是简洁的，而且必须迅速演示业务目标。您将会感到惊奇，在这一时刻不同的风险承担者有如此多的不同观点。在这个项目阶段，大多数风险承担者关心的是投资收益。因此，一个项目描述就是风险承担者之间的第一个一致观点，因为它明确声明了新系统的目标。

提示：

在开始编写系统描述之前，您可能会发现为客户确定一个特定于域的术语将很有帮助；这将会为通信建立一个通用的词汇表。您可以为附加的说明可选地提供一个可操作的模型，如图 1-1 所示。

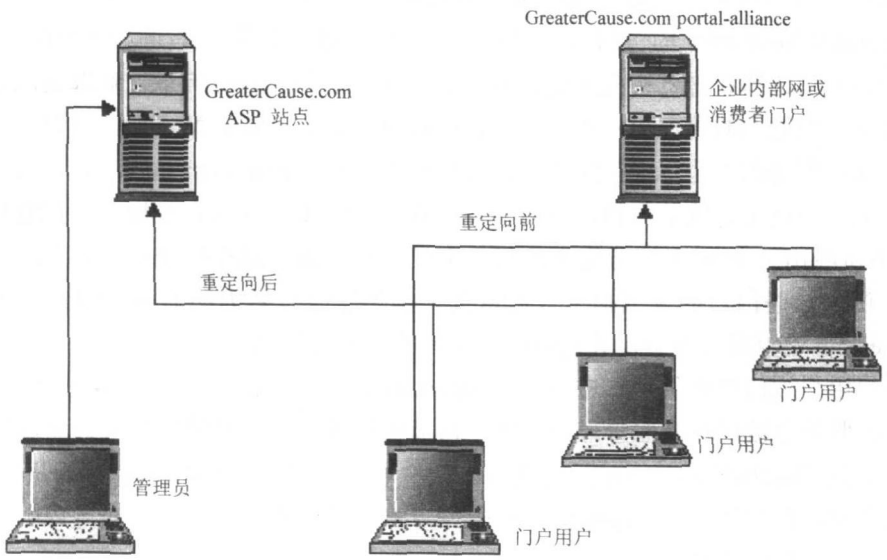


图 1-1　GreaterCause 操作模型

GreaterCause 系统定义

以下的术语用于定义问题域。

- GreaterCause 是一个驻留在中心位置的慈善性应用程序。
- GreaterCause.com 是 GreaterCause 的站点域名，其中 GreaterCause 应用程序作为驻留的服务被访问。为了简单起见，术语“站点”都是指 GreaterCause.com 站点。
- 门户(portal)是一个为业务和消费者服务提供的个性化访问点。
- 门户域是一个驻留消费者门户或企业内部网的域。
- GreaterCause.com 门户联盟的形成是门户在门户页上为将门户用户重定向到 GreaterCause.com 站点提供通行证或入口组件(也称为门户件)的结果。
- NPO 是一个非赢利组织，它注册 GreaterCause.com 站点是为了从预期的捐赠人那里请求慈善捐助。

GreaterCause.com 域负责将 GreaterCause 慈善捐赠应用程序驻留在中心位置，捐赠人可以通过不同的消费者门户和社团内部网访问这个站点。

门户提供者使用 GreaterCause 创建一个联盟(例如一个服务约定)，该联盟用于为它们的用户群获取 GreaterCause 服务。门户提供者和 GreaterCause 之间达成的用于服务于门户用户的协议称为门户联盟(Portal-Alliance)。每一个门户联盟都有一个相关的管理 ID 和口令，利用这个 ID 和口令，门户联盟管理员(门户提供者的一个雇员，或是其指定的人)可以维护与门户相关的配置文件信息。门户提供者提出使用门户的一部分不动产来将一个智能入口或通行证提供给 GreaterCause 站点的方法，从而使自己可作为 GreaterCause 应用程序的看护者。GreaterCause 通行证可作为门户件(portlet)使用，门户件被集成到门户域合作者的门户视图中。

非赢利组织(NPO)注册 GreaterCause，并在 GreaterCause 数据库中列出它们自己，从而可收到访问这个站点的访问者的慈善捐助(也就是捐赠物)。每一个注册的 NPO 都会获得一个管理 ID 和口令，NPO 管理员可以利用它们维护相关的配置文件信息。

虽然 GreaterCause .com 的访问者可以捐赠给任何可靠的慈善机构(例如 NPO)，但门户提供者可以影响捐赠人选择慈善机构的决定，这种影响是通过为慈善机构举行竞选完成的。门户联盟的管理员可利用他们的管理 ID 和口令登录到 GreaterCause.com 站点上，并且为非赢利组织举行国家和地区级别的竞选。这些竞选结果存储在 GreaterCause .com 站点上，并且随后显示在门户域各自的门户页上。这些由门户联盟管理员在 GreaterCause.com 数据库中创建的有号召力的 NPO 列表可经由一个 Web 服务提供给每一个门户域，然后这个列表由驻留在门户页中的门户件展示。访问这些门户的预期捐赠人可以选择捐赠给有号召力的非赢利组织，或者直接到 GreaterCause.com 站点进行搜索，并捐赠给所选择的非赢利组织。

一旦门户用户被门户提供者重定向到 GreaterCause.com 站点，正如门户用户所看到的那样，GreaterCause 服务会被门户联盟管理员定制，从而保存各自门户域的标记和导航结构。在将门户用户重定向到 GreaterCause.com 站点之前，门户域负责验证门户用户(也称为捐赠人)。在重定向捐赠人之前，门户域和 GreaterCause.com 站点相互进行验证。在重定向过程中，门户域将捐赠人的注册信息提供给 GreaterCause.com 站点。

所有的事务历史都使用捐赠人的注册 ID 和端口域的关联记入日志中，捐赠人可查看当前