彩铃统一管理业务生成平台的设计与实现*

Design and Implementation of Unified Color Ring Back Tone Management Service Creation Platform

徐 磊 廖建新 王 纯
(北京邮电大学网络与交换技术国家重点实验室 北京 100876)
(东信北邮信息技术有限公司 北京 100083)

摘要: 彩铃业务目前发展势头迅猛,本文在深入分析现有彩铃平台和彩铃管理平台的基础上,提出了改进方案,在彩铃管理平台基础上形成彩铃统一管理业务生成平台,该平台统一生成"管理业务"对彩铃业务进行管理。详细阐述了彩铃统一管理业务生成平台的设计原理和实现机制,最终达到增强彩铃业务可维护性,稳定性和可控性的目的,并最大限度保证了彩铃业务管理与接入方式无关。

关键词: 智能网 彩铃管理平台 语音可扩展标记语言 依赖注入 面向方面编程

1 引言

彩铃业务按照实现方案的不同可以划分为基于智能网的实现方式和基于交换机的实现方式[1]。本文讨论的彩铃业务实现方案是基于智能网的实现方案,该彩铃平台是由智能网网络实体 IIP(Independent Intelligent Peripheral)演化而成的,IIP 作为提供 SRF(Service Resource Function)的物理实体,向智能业务的使用者提供各种特殊资源。本文中的 IIP 根据智能网 CS2 规范要求加以扩充,将大量业务执行能力部署到 IIP 上。

目前 IIP 平台内部的体系结构如图 1 所示,控制节点 CN(Control Node)是整个系统的核心,CN 控制资源节点 RN(Resource Node)播放彩铃铃音或语音提示音,CN 与短信网关相连,控制其给用户发送短信,通过 N7server 与 SCP(Service Control Point)相连,通过 I7server 与 SSP(Service Switch Point)相连[1]。

CN 上运行业务逻辑程序 SLP(Service Logic Program),按照其指定的流程实现用户的彩铃业务,除此以外 CN 上还运行 SLP 以语音或短信接入方式实现对彩铃业务的管理,或者通过 socket 接口与 web 服务器相连,实现 web 接入,WAP 接入以及第三方 SOAP 的接入来实现彩铃业务管理。

彩铃业务功能丰富性的要求使得业务的管理逐步复杂,用户对于管理接入方式多样性的要求也在日益增长。而当前 IIP 平台中管理业务的生成对于迅速开发新功能,满足多样的管理接入方式等要求需要投入大量的人力和时间,可扩展性,可维护性均不理想。

彩铃管理平台管理业务逻辑的改进目标为:增强复用性,开放性,并加强可维护性,可控性,灵活性,以达到管理业务逻辑对管理接入方式的透明并降低维护阶段对业务及管理业务修改的风险。以下针对如上目标,提出彩铃统一管理业务生成平台的概念,并详细分析其对原有彩铃管理平台的改进思路和实现策略。

2 设计原理和实现机制

2.1 系统接口

如图 2,彩铃统一管理业务生成平台连接了各种

* 基金项目:国家杰出青年科学基金(No. 60525110);国家 973 计划项目(No. 2007CB307100, 2007CB307103);新世纪优秀人才支持计划(No. NCET-04-0111);电子信息产业发展基金项目(基于 3G 的移动业务应用系统);电子信息产业发展基金重点项目(下一代网络核心业务平台);电子信息产业发展基金项目(基于内容的综合通信网络计费平台);国家高技术产业化信息化装备专项项目(支持数据增值业务的移动智能网系统)。

接入方式代理，第三方接入代理通过携带 SOAP 消息的 HTTP 请求调用业务封装底层，web，WAP 和 IVR 接入代理分别通过 HTML，WML 和 VoiceXML 调用业务封装底层提供的服务，客户终端与 SMS 接入代理采用 CMPP 协议通信，SMS 接入代理调用业务封装底层的

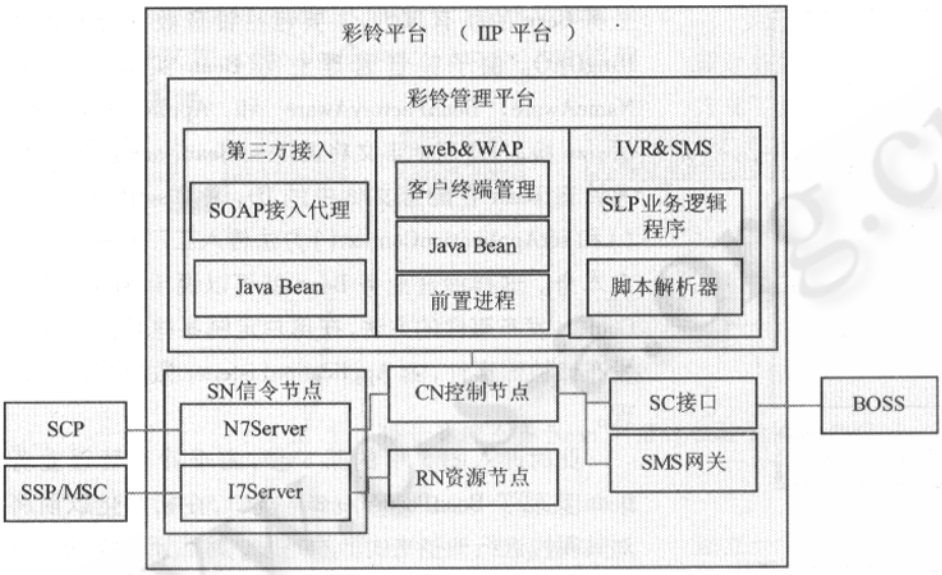

图 1　IIP 平台内部模块及相关外部实体

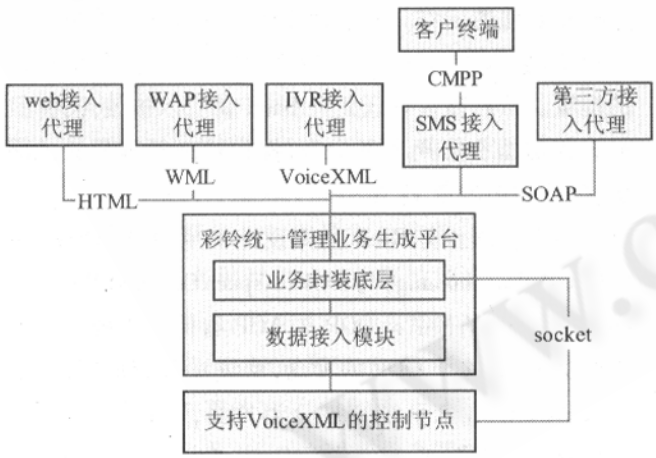

图 2　彩铃统一管理业务生成平台的接口

能力，给接入的客户端反馈消息。业务封装底层其下通过 socket 方式与支持 VoiceXML 的控制节点连接[2]。

2.2　内部模块

2.2.1　各模块功能概述

彩铃统一管理业务生成平台内部主要包括业务封装底层模块和数据接入模块，其中业务封装底层是整个平台的核心，数据接入模块使用 Hibernate 作为框架接入底层数据，如图 3 所示。

业务封装底层面向两类用户：彩铃用户通过各种管理接入方式，调用业务封装底层提供的服务，完成自主管理彩铃的功能；第三方 SP（Service Provider）根据用户的需求，设计新的管理业务属性，编写 HTML，WML，VoiceXML，XML 或者符合 CMPP 协议规范的代理，调用业务封装底层提供的服务，可以方便形成新的管理业务流程。

业务封装底层向下通过 DAO（Data Access Object）层的数据接入模块与数据库相连，将业务数据持久化，借助 CN 的业务执行能力与 SCP，SSP 等实体进行信令的交互，最终完成管理业务的实现。

2.2.2　业务封装底层模块

业务封装底层从横向功能上划分主要包括管理业务 Bean 模块，Bean 管理模块，安全管理模块，鉴权管理模块，事务管理模块，标签管理模块，配置管理模块，负荷管理模块，日志管理模块，异常处理模块等，如图 4。以下将首先介绍关键的 Bean 管理模块，然后以铃音管理业务 Bean 为例详细介绍业务封装底层向上层接入代理提供服务的一次完整过程。

彩铃统一管理业务生成平台提供一种可选的表现层处理机制以及业务层和表现层的结合方式，采用 webwork 提供的机制提供接受并分发请求的控制 servlet。webwork - default. xml 作为 webwork 的主要配置文件，配置 webwork 使用的拦截器，xwork - ring. xml，xwork - community. xml，xwork - user. xml，xwork - cpchargeset. xml 分别配置处理铃音，新闻，用户和套餐相关请求的 action。webwork. properties 配置文件则指明采用 SpringObjectFactory 来装配 action[3]。

彩铃统一管理业务生成平台还集成了一种可选的标签处理模块，该模块配置在 sitemesh. xml 文件里。

下面以铃音管理业务 Bean 为例，详细介绍一个 Bean 如何将管理业务细节封装，对外提供统一的服务，供各种管理接入代理调用。

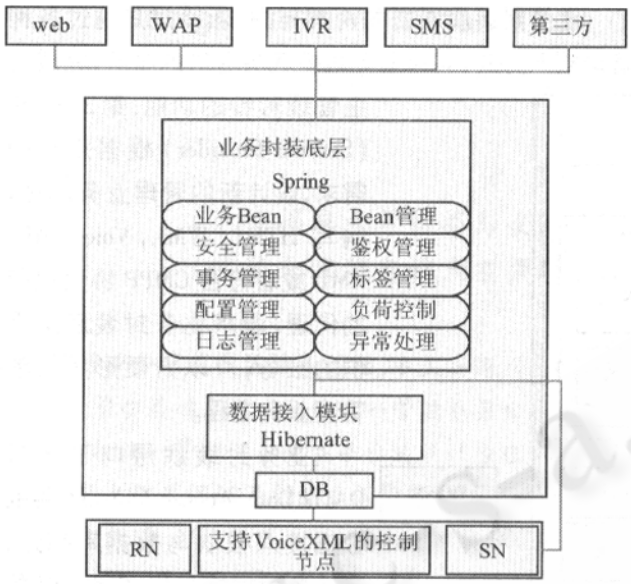

图 3　彩铃统一管理业务生成平台的内部模块

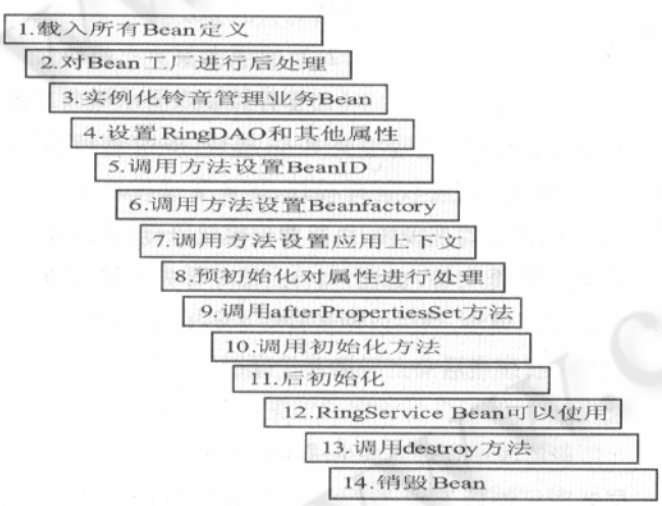

图 4　抽象的铃音管理业务 Bean 的生命周期

如图 4，应用上下文载入 applicationContext - service. xml 配置文件中所有管理业务 Bean。BeanFactoryPostProcessor 在实例化铃音管理业务 Bean 之前，对应用上下文做一些后处理工作[1]，将载入应用上下文中 Bean 的个数记录下来。

当接收到调用者的一个请求时，控制流处于 webwork 的 ServletDispatcher 处，该 sevlet 将请求分发给相应的 action，3.3.2.1 节中提到 webwork. properties 配置文件指明采用 SpringObjectFactory 来装配 action，因此，此后控制流将转到 spring 框架。

如图 6 中的第 3 步，应用上下文将铃音管理业务 Bean 实例化，使用依赖注入，根据配置文件配置铃音业务 Bean 的所有属性，主要包括铃音数据接入组件 RingDAO。由于铃音管理业务 Bean 实现了 BeanNameAware，BeanFactoryAware 和 ApplicationContextAware 接口，应用上下文将调用 setBeanName( ) 方法传递铃音 Bean 在配置文件中的 ID，调用 setBeanFactory( ) 和 setApplicationContext( ) 方法传入工厂和应用上下文本身。这样铃音业务 Bean 就可以调用 ApplicationContext 发布事件的方法，在用户定制某些特定铃音时发布该特殊事件，由 ApplicationListener 做出相应的处理。

此时进入图 5 中的第 8 步，由于铃音管理业务 Bean 实现了 BeanPostProcessor 接口，在初始化以前将对其属性进行一些格式上的处理。然后调用初始化方法，铃音管理业务 Bean 没有设置该方法，如果用户在配置文件指定了铃音管理业务 Bean 的 init - method 方法，通过该方法可以在使用铃音管理业务 Bean 以前做一些初始化的工作。在初始化以后 Spring 又提供了一次机会，切入到 Bean 的生命周期内检查或修改 Bean 的配置。然后铃音管理业务 Bean 就可以被使用，并且一直被保留在应用上下文中，提供铃音管理相关的服务，如图 6 中的第 12 步。

铃音管理业务 Bean 主体设计采用 OOP(Object - Oriented Programming) 与 AOP(Aspect Oriented Programming) 相结合的模式。OOP 与 AOP 看待应用程序的方式截然不同，按照 OOP 的观念，系统被分解成对象，而按照 AOP 的观念系统则被分为方面，两者各有利弊。在彩铃统一管理业务生成平台中，设计应用系统的原则是各取其长处，目标是减少代码重复，增加可读性和可维护性。

在平台的设计实现中，将铃音看成是一个对象，铃音对象独有的方法采用 OOP 设计模式，继承机制可以在处理不同类型铃音时使用相同的行为，多态机制可以使用相同的方式处理不同类型的对象。但是在有些问题的处理上，无法用 OOP 来避免代码的重复，比如安全性检查，日志处理和话单处理。例如，可以使用

OOP 的思想添加一个新的安全管理业务方法，但是在识别调用安全检查的那一点上，OOP 无法提供更多的帮助，完全依赖 OOP 这些具有横切性的代码就会毫无规律地散布在整个对象模型中，造成无法避免地代码重复[5]。替换策略是利用 AOP 将这些代码规整起来，将用于解决每个横切问题地代码收入一个独立地模块中，并在配置文件 applicationContext - service. xml 增加一条声明性配置。

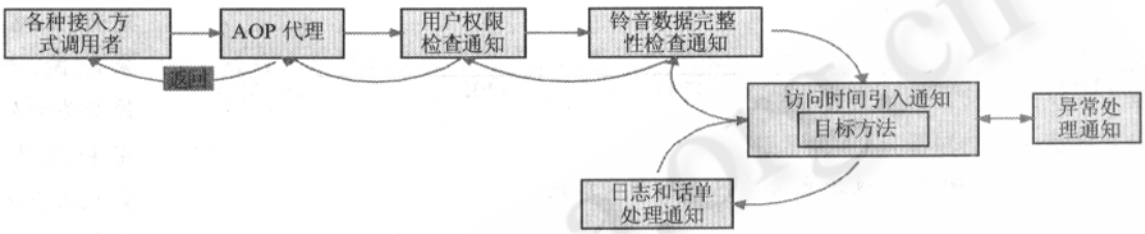

图 5　铃音管理业务 Bean 调用过程控制流示意图

如图 5，仍然以铃音管理业务 Bean 为例，图中展示了控制流如何从调用者到 AOP 代理，流经通知链，再流至目标方法，最后返回的过程。

彩铃统一管理业务生成平台采用的 Spring 框架是在运行期创建 AOP 代理的，该代理为 JDK 动态代理，它首先依次调用"用户权限检查"和"铃音数据完整性检查"的前置通知，然后调用铃音管理业务目标方法。目标方法执行过程中发生异常，将交给异常处理通知统一处理，目标方法处理完成以后，将调用后置通知完成记录日志和上报话单的功能。前置，后置和异常通知是在铃音管理业务对象的方法周围被织入，除此以外，引入通知将各种对象公用的属性或方法提取出来，铃音管理业务 Bean 利用引入通知添加了访问时间这个属性以及相应的 getter 和 setter 方法，使目标对象只需专注于自身独有的方法的编写。

铃音管理业务 Bean 处理完业务逻辑后，将流程控制权交给 webwork 提供的 sevlet，根据 xwork. xml 里的配置，跳转到相应页面，在向调用者返回页面标签以前，可以使用彩铃统一管理业务生成平台提供的标签处理机制对标签进行处理，标签处理机制 sitemesh 根据使用 servlet filter 截取的请求，捕捉返回的标签。查找配置文件以确定哪一个装饰器需要被应用，然后 servlet 向包含装饰器的 jsp 发送请求，最终返回被装饰器页面包装后的标签给调用者。

如果不再需要铃音管理业务 Bean 了，由于其实现了 DisposableBean 接口，可以调用 destroy( ) 方法将其销毁。

3　性能分析

为了提高复用性，可维护性和可控性，彩铃统一管理业务生成平台将流行的设计模式应用于传统智能网方式下业务的生成，借鉴 IOC(Inversion of Control) 和 AOP 的思想，对原有彩铃管理业务平台进行了架构上的调整。

AOP 的实现需要借助 Java 的反射机制，而且需要建立一个对象来代表被拦截的方法调用，而原有的彩铃管理业务平台使用传统的 MVC 架构满足 web 和 WAP 的接入方法，相比而言，改进后的平台在内存和 CPU 资源的占用上都会有所增加。

基于此，在设计时尽量避免在非常细的粒度下使用 AOP，而只在业务对象(相当于 EJB 中的无状态本地 session Bean) 上使用 AOP 提供的增强。如果只在业务对象的层面上使用 AOP，这种性能开销将会很小[6]，再加上 Java 在反射和垃圾收集上性能的大幅提升，这种性能开销将会更加不明显。

以下使用 jconsole 工具，分别从堆内存，非堆内存和线程数目来比较原系统和架构调整后系统性能的差别。

Java 虚拟机的堆内存存放运行时数据，所有类实例和数组的内存均从此处分配，对象的堆内存由垃圾回收器的自动内存管理系统回收。

如图 6，图中 1，2 曲线分别对应改造后和改造前系统堆内存的占用情况，可以看出，改进后的彩铃统一管理业务生成平台由于采用 spring 的 IOC 机制，业务 Bean 的所有属性自动装配，会占用较多的堆内存空间，但 Java 垃圾回收性能的提高，使内存仍然维持在相对较低的水平。

Java 虚拟机非堆内存存储每个类结构，如运行时常

数池、字段和方法数据，以及方法和构造方法的代码。

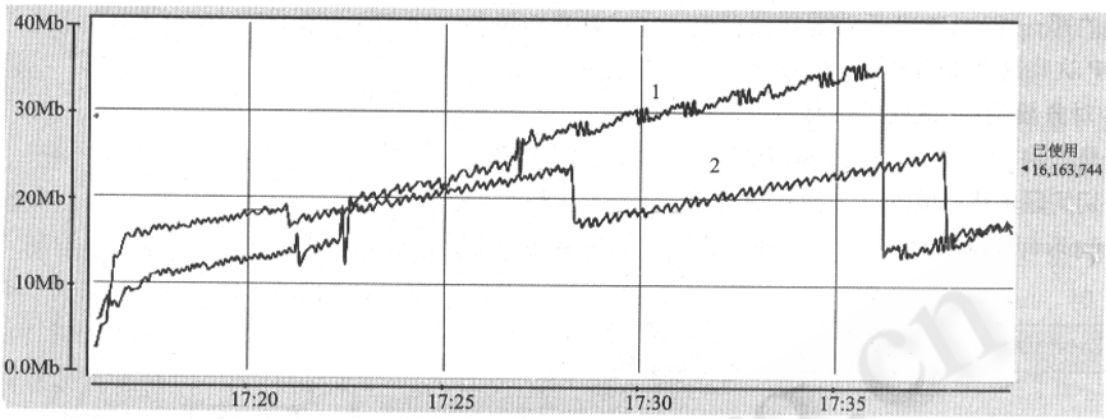

图 6　改进系统堆内存占用曲线图

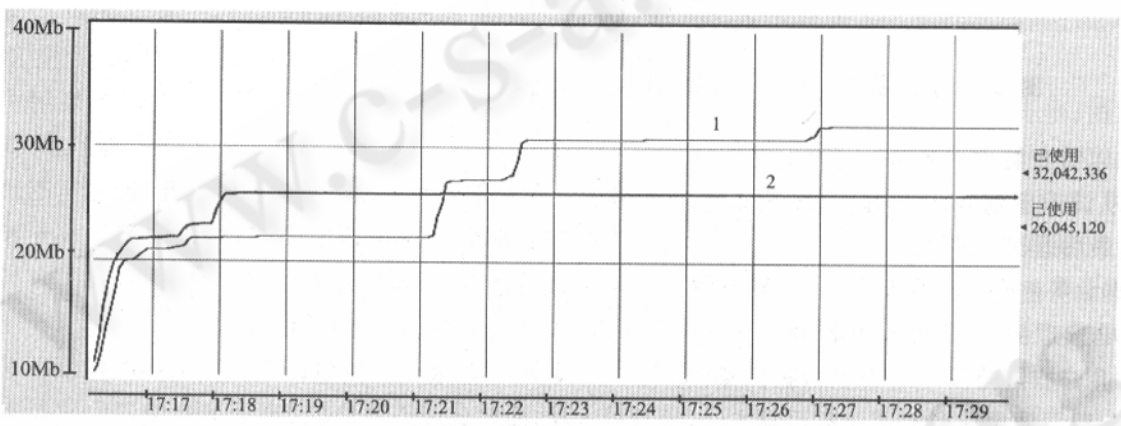

图 7　原系统非堆内存占用曲线图

如图 7，图中 1，2 曲线分别对应改造后和改造前系统非堆内存的占用情况，改进后的彩铃统一管理业务生成平台使用引入 AOP 的思想，为每一个被拦截到的方法调用再增加一个对象，而且，使用 hibernate 框架实现 DAO 层，类的数目比原系统有所增加，但由于将一些公共的属性和方法抽象出来，每个类的复杂度大大降低，使代码总量反而比原系统减少了 20% 到 30%。

4　结束语

本文提出的彩铃统一管理业务生成平台，是在传统智能网模式下的 IIP 平台上抽象出一个统一的业务封装底层，使得各种接入管理方式能够分别通过 HTML，WML，VoiceXML，XML 和 CMPP 标准协议来调用封装底层的服务。该封装底层在设计时考虑到彩铃业务目前用户量增多，管理接入方式多样，管理业务属性增加频繁等特点，引入了 IOC 机制和 AOP 思想，为平台提供良好的安全管理模块和可选的标签管理模块。

参考文献

1　沈奇威、廖建新、王纯、朱晓民，彩铃业务的研究和设计，第九届全国青年通信学术会议论文集，重庆，中国，2004. 5：484－489.

2　沈奇威、温雪、廖建新，基于 VoiceXML 的增强智能外设，电信工程技术与标准化，2004 年 05 期：65－69.

3　[美]Patrick Lightbody，Jason Carreira 著，谭颖华，张云飞，唐勇译，Webwork in Action，电子工业出版社，2006 年 11 月.

4　[美]Craig Walls，Ryan Breidenbach 著，李磊，程立，周悦虹译，Spring in Action，人民邮电出版，2006 年 4 月.

5　[美]Rod Johnson，Juergen Hoeller 著，JavaEye 译，Expert one－on－one J2EE Development without EJB，电子工业出版社，2006 年 2 月.

6　[美]William Crawford，Jonatban Kaplan 著，刘绍华，毛天露译，J2EE 设计模式，中国电力出版社，2005 年 10 月.