



# 快速行动应用软件开发之中间件架构设计

沈文祥

(侨光科技大学 信息管理系,台湾 台中 40721)

**摘 要:**行动应用服务可以提升企业的工作效能与竞争力,能带来更多新的潜在消费者,然而太多异质系统、行动平台、开发与维护成本过高的因素,使得在既有系统上开发行动应用软件是件不容易的事。目前主要的行动平台有 iOS、Android 与 Windows Phone 等,因在不同平台上实作行动应用程序的技术皆不同,导致企业开发支持多平台的行动应用软件时,需耗费非常多的成本与人力。利用行动中间件架构设计,使软件开发者于行动服务程序开发过程中,进行与既有系统间沟通和输入/输出更为容易。行动中间件基本上能隐藏行动环境中工作的复杂性,以致降低多平台支持的困难度,并减少重复行动软件开发的人力与增加开发组件重用的机会。在此论文中,提出一个行动中间件架构设计,让整合既有系统的行动应用服务程序的开发可以更有效率。

**关键词:**中间件设计;MVC 模型;行动应用程序;插件样式

68

中图分类号: TP311.1

文献标识码: A

文章编号: 1671-931X (2015) 05-0068-07

## 一、前言

由于智能型手持装置快速的发展,2011 年全球手持装置出货量达 4 亿 5000 万只,而所有行动应用软件下载量已超过 249 亿次,并且预测 2015 年行动应用软件下载次数将接近 1830 亿次<sup>[2]</sup>。因此,越来越多的企业纷纷投入人力开发行动应用软件,甚至开发整合现有系统的行动服务。而企业在众多的应用服务中采用行动技术可以为企业带来的好处有增加工作效率、藉由行动化服务提升竞争力与增加潜在的消费者群<sup>[3]</sup>。

再根据 IDC(国际数据信息)发表的研究结果显示:2014 年台湾 4G LTE 的行动上网用户数预期将超过 100 万户,预计 2015 年 4G 用户数相对 2014 年将出现 3 位数的成长。2012 至 2016 年亚太地区(不含日本)应用开发及部署软件(application deployment software)的复合成长率高达 7.6%,以云端运算及行

动化应用为两大市场推手。问世近 30 年的中间件(Middleware),随网络应用如野火般蔓延的普及速度,持续深入网络应用的各个层面<sup>[4]</sup>。Enea 公司发布了名为 Element,适用于电信、汽车、产业控制和医疗设备的新高可用度中间件解决方案。该产品为一整套处于操作系统和应用程序之间的服务。它提供了用于多个操作系统和处理器应用程序线程间同步、执行、监控和通讯设立等核心服务<sup>[5]</sup>。台湾学者设计一个网页式中间件之开放软件架构,藉由设计一套统一函式库接口,支持跨平台且提供用户可自行扩充组件的功能<sup>[1]</sup>。

### (一)整合既有系统的行动软件开发

目前企业很多运行已久的软件系统,大多是利用传统结构化技术来设计开发,而这些旧有的软件系统内部组件彼此的耦合关系非常密切,功能及数据的相依性非常高,所以这些传统的软件系统在维护上所花费的成本会比重新开发一个新系统的成本

收稿日期:2015-06-15

作者简介:沈文祥(1963-),男,侨光科技大学学生事务处服务学习组组长,助理教授,研究方向:软件工程、服务学习。

要高出很多。从了解程序代码原始的需求和设计规格的识别发现，在行动软件建置或软件的维护和发展上，好的软件架构(framework)占相当重要的地位，因为它可以帮助我们了解旧有程序结构与逻辑，针对旧有软件系统进行修改或新增功能时也很有帮助。软件工程的工作需要去找出一些较高阶层的信息，如系统的结构性、功能性、动态行为与设计原理等。除此之外，在从事软件整合的过程中也要建立一个结构性较好的架构，并依此架构去重新实作此系统，如此，软件系统的可维护性就会增加，要再既有系统上发展新的组件或功能也较具有弹性。

## (二)中间件与行动平台

中间件(middleware)是基础软件的一大类，属于可复用软件的范畴。顾名思义，中间件处于操作系统软件与用户的应用软件的中间。建立中间件是一个可以解决大多数软件维护问题的方案之一，所有的应用程序都应透过原先定义好的中间件来存取系统资源，而中间件就扮演传递上下层信息及错误管理的角色，藉以提供服务及保护系统运作的平顺，为了让所有的应用程序可以容易的透过中间件来取用系统资源，且方便日后扩充软件组件，所以无论对上或是对下都需要一个统一的接口来提供存取，这即是中介程序的重要定义<sup>[6]</sup>。中间件已经成为一个支持分布式应用程序的重要的建构组件，中间件的作用是提出一个统一的程序模型给应用程序设计师，并掩盖了异质性和分散性的问题。这个课题在反映正规化活动能见度增进的重要性，如 ISO/ITU-T Reference Model for Open Distributed Processing (RM-ODP),OMG 之 CORBA,Java RMI,Microsoft 之 DCOM 以及 Open Group 之 DCE<sup>[7]</sup>。

目前主流的行动平台有 iOS、Android 与

Windows Phone 等，但因在各平台实作行动应用软件的技术皆不同，导致企业想开发一个支持多平台的行动应用软件时，需要耗费非常多的成本与人力。而企业若要开发整合既有系统(legacy system)的行动应用服务则更加的不容易，除了各种不同的行动平台之外，还要处理众多异质性的系统整合问题、开发与维护的成本、为了行动服务需求可能要修改现有系统与行动设备硬件资源的限制等等因素。为了解决此种问题，建立中间件将是一个解决方案(图1)。透过中间件扮演企业异质系统与行动中间件之间的沟通桥梁，可降低需修改既有系统的机率；中间件可以提供不同行动平台统一标准的资料封装格式，减少支持多行动平台的难度以及减少重复开发与增加重复利用已开发的组件机会。

## (三)研究动机与目的

目前现有的行动中间件尚未有完善的架构可以支持快速且易开发扩充整合既有系统功能的行动应用服务的设计。因此，在本研究中，设计一个行动中间件架构，让整合既有系统的行动应用服务开发可以更加的快速与容易。

本论文主要目的为针对行动应用软件，设计一行动中间件架构并具备以下特性：①可快速开发扩充行动应用服务的方法；②有效降低修改既有系统的需求；③开发插件的简单设计；④支持多种异质性行动平台。

此一行动中间件架构主要是基于 Web 服务运作方式，加入插件样式(Plug-in Pattern)使得在开发整合系统的功能时，能够容易快速地在行动中间件上扩充新功能；另加入可配置(Configuration)功能，透过读取配置文件内容，让开发人员能够在不必修改原系统程序情况下（仅需修改配置文件参数），即可支持更多的系统或更多的任务。

本论文架构共分为五个部分，第一节为前言，介绍整合现有系统之行动软件开发方式、行动中间件在行动软件开发扮演的角色与研究动机目的；第二节介绍相关技术，探讨既有系统整合技术演进；第三节说明行动中间件设计；第四节描述实作流程与结果讨论；最后第五节为结论。

## 二、相关研究

研究中利用插件样式架构让行动中间件易于扩充整合既有系统功能的行动应用服务，并使用 MVC 样式 (Model-View-Controller Pattern) 衍生之 MVP 样式 (Model-View-Presenter Pattern) 建构行动中间件使其易于维护。而在本研究应用到相关的技术，以下将依序介绍。

### (一)旧系统演进

近年来行动装置的快速发展，企业对于在既有系统支持行动装置服务的需求越来越高，而要完成

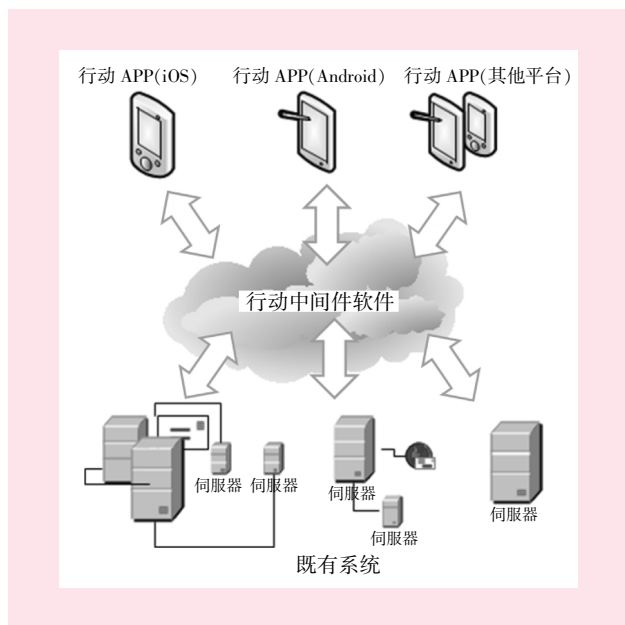


图1 行动中间件示意图

这些任务是一大挑战。为了能够完成目的,让既有系统支持 Web 服务是目前较为常见的方法。然而各种不同的企业信息系统设计皆不大相同,实作上需要使用各种解决方案。因此 Almonaies 等学者<sup>[8]</sup>针对既有系统与服务导向架构(SOA)的演进收集相关的文献,归纳其实作策略分成以下四大类型:

**Replacement:**以新系统取代既有系统,此方法又称大跃进(Big-Bang)策略。此策略可以降低新系统的维护成本;但是缺点耗时、昂贵以及需要有经验累积。

**Wrapping:**针对既有系统提供 Web 服务接口。此策略的优点在于开发快速,对于无法修改既有系统原始码且商业价值高的系统是不错的选择;而缺点是在维护成本较高且此方法无法改善既有系统的质量。

**Redevelopment:**利用再工程(Reengineering)与逆向工程(reverse engineering)分析既有系统,修改原始码提供 Web 服务接口。优点可增加弹性与降低成本;此方法缺点则须取得原始码。

**Migration:**方法介于 Wrapping 与 Development 间,透过撷取原始码并修改,新增提供 Web 服务接口功能与既有系统并存。优点有工具支持以及稳定的环境;缺点耗时、须源代码以及经验的累积。

## (二)插件样式

插件目的是为了某些软件功能用途,能够在运行时间(run time)透过配置决定其实际对应的对象,而让软件设计具有更佳的弹性(图 2)。而使用此样式的好处是让软件容易的扩充新功能、让第三方开发者也能扩充软件、减少软件的大小以及切割软件原始码易于软件授权等等。

图 3 为插件样式运作流程,当执行时期欲使用某插件时,呼叫 Plugin factory 检查插件配置文件,透过反射机制找寻对应的插件类别并生成该插件对象。

## (三)MVC 模型与 MVP 模型

MVC 样式模型与组件互动方式如图 4 所示,为软件工程中一种架构样式,主要把软件系统分为三大部分:①模型(Model):负责数据存取与商业逻辑;②检视(View):主要任务为数据与使用接口的呈现;③控制器(Controller):主要处理用户请求与状态改变通知模型和检视改变。

MVC 样式运作流程为当控制器接受一个使用者请求时,控制器会依据使用者请求的内容通知对应的模型处理数据存取或商业逻辑,当模型完成任务之后会改变检视呈现数据或封装数据回传给用户。因明确地将软件系统分为三大部份,使得软件系统具低耦合性、高维护性、扩充性、可测试性以及重用性等优点。

图 5 为 MVP 样式模型与组件互动方式,应不同

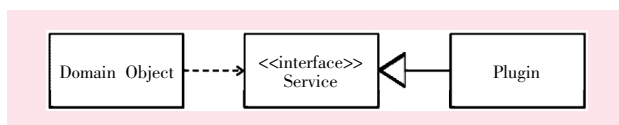


图 2 插件样式架构

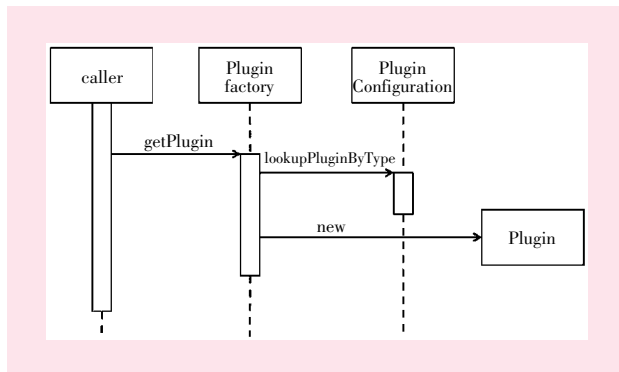


图 3 插件样式运作流程

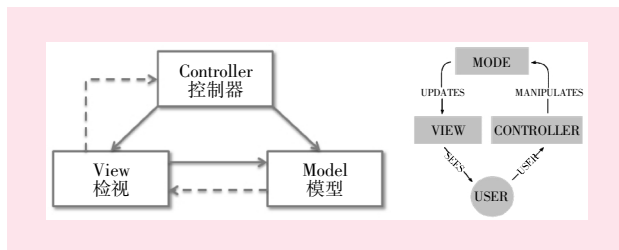


图 4 MVC 样式模型与组件互动方式

(source: Wikipedia—<http://en.wikipedia.org/wiki/Model-view-controller>)

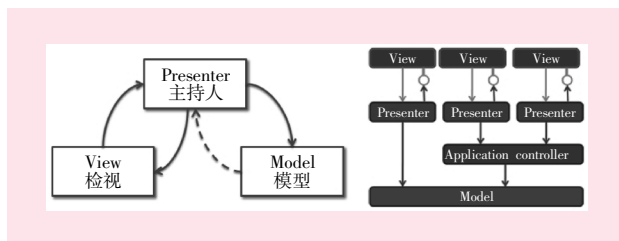


图 5 MVP 样式模型与组件互动方式

(Source:<http://blog.vuscode.com/malovicn/archive/2007/12/18/model-view-presenter-mvp-vs-model-view-controller-mvc.aspx>)

的软件设计架构需求,从 MVC 样式衍生出 MVP 样式(Model-View-Presenter Pattern),主要差异在于让 Model 与 View 之间完全无任何关系,而改由主持人(Presenter)来控制检视(View)所要呈现的内容,其优点是可以由 Presenter 来集中处理事件,View 只需负责画面样板。而因为 Web 服务不需要实作 View 部分显示画面,因此 MVP 样式较 MVC 样式更适合 Web 服务。

## 三、行动中间件设计

本节说明本研究提出的行动中间件架构设计,如何将中间件与插件框架结合以及设计让行动应用



软件呼叫使用的接口；最后描述如何实作与既有系统交接的通用模型。

### (一)中间件设计架构

行动应用软件透过中间件与既有系统做整合,带来了许多好处例如异质性的既有系统如有相同功能时,即可共享相同的中间件组件;透过中间件可降低修改既有系统的需求;与各平台行动应用软件的交接处理较容易等等。然而,现有的方法尚未能有效的快速且容易开发整合既有系统功能的行动应用服务,导致目前开发与既有系统整合的行动应用服务需花费许多的人力与成本,效率十分不彰。为了解决问题,将插件框架(Plugin framework)加入行动中间件架构中,让开发者仅需实作与既有系统整合的插件组件,即可完成支持行动应用服务功能。图 6 说明行动中间件架构,服务路由器(Service Router)主要任务为与行动应用软件做交接以及分派对应行动端请求的应用服务;插件管理员(Plugin manager)负责管理插件,检查可挂载的插件组件以及指定之插件挂载后的服务对象;最后插件组件为实作与既有系统功能交接的方式,并且使用通用模型(Common Models)加速开发者开发插件组件,例如若影片是使用 Youtube 管理,开发插件组件时,即可使用整合 Youtube 服务的模型来加快开发速度。

同时行动中间件设计也套用了 MVP 设计样式,区分出 Model、View 和 Presenter 的部分,处理数据存取与商业逻辑由中间件的模型层(Model)实现;View 与 Presenter 则都在插件框架部分处理,让开发者决定这两部分的设计。而提出的行动中间件架构<sup>[10]</sup>,主要分为以下四层:

**服务路由器:**主要用来处理外部程序或行动端所发出的请求,将其对应到合适的服务或插件处理请求的内容。

**插件框架:**主要实作插件组件与服务对应管理,达到易扩充的特性,使行动中间件易于扩充支持新功能,并且为了让插件具弹性且重用性,在插件框架同时于插件组件与插件管理员中加入可配置(configurable)功能,藉由配置功能,让插件仅需改变配置文件内容,即可将同一个插件套用在其他系统。

**通用模型:**实作各类型常见的模型,将整合既有系统服务主要的商业逻辑交由此部分处理,藉由不断的扩充通用模型,使得开发者在开发插件时较为容易且弹性,并且为了让通用模型也能够具弹性与重用性,也在通用模型当中加入可配置功能,透过此功能让模型也能够仅在修改参数的情况下,变更所支持的服务系统。

**工具模块(Utilities):**实作常见基本的工具,如网络通讯、数据库连接、讯息数据格式处理与安全相关模块等等。

将整合既有系统功能的主要商业逻辑交由通用



图 6 行动中间件架构

模型处理,并在通用模型与插件框架中加入配置功能,加强其弹性与重用性,让开发者可以更专注的开发对应既有系统功能的插件组件,促使开发者可以提升开发的效率。若当通用模型中,没有支持开发者欲开发插件功能时,也能够自行快速开发模型或调整模型的参数,使其能很容易地完成新模型让插件可以使用时(图 6)。

### (二)插件框架设计

为了能够加速扩充行动应用服务功能,在研究中实作插件框架,利用插件架构可扩充性的特性来完成目标。同时为了能够加强插件的可重用性,将配置功能加入插件框架。让开发者可以透过只修改配置文件内容,即可让插件支持不同的系统,主要分为五个类别:

**PluginManager 类别:**实作管理插件功能,如加载所有插件、挂载与卸除插件。

**ConfigurablePluginManager 类别:**继承 PluginManager 类别,扩充插件绑定服务功能,使插件变成 Web 服务与可配置功能让 PluginManager 可以透过配置文件功能来管理插件服务绑定服务的 URI 与启用的功能。

**IPlugin 类别:**实现插件功能,如挂载与卸除。

**MethodView 类别:**处理 HTTP 请求功能,如 Get、Post、Put 与 Delete,透过此类别可以让开发者实作 RESTful<sup>①</sup>服务。

**BasePlugin 类别:**实作 IPlugin 与 MethodView 类别,使其具插件功能与 HTTP 处理的功能,并且加入配置功能(load config)与服务注册功能(register api),让插件也能够透过可配置功能达到更好的弹性与可重用性以及服务注册功能让实体的插件可以绑定服务。

插件管理员的配置文件内容主要采用 YAML 语言作为配置文件格式,其优点在于它是可读性高的数据序列程序语言<sup>[9]</sup>。目前配置内容主要是针对插件挂载服务的位置、插件支持的 HTTP 方法以及是否挂载功能编辑设定参数,如此可以让开发者或管理人员不需要更改程序,即可变更服务的设定。

### (三)中间件运作流程

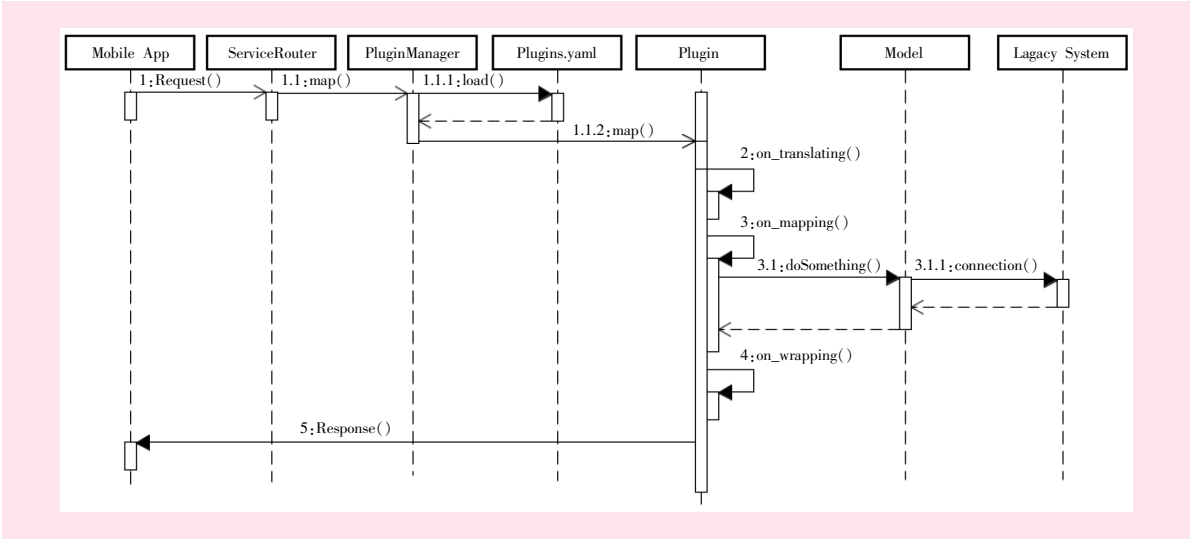


图 7 插件运作循序图

中间件整体运作流程如图 7 循序图(sequence diagram)所示:

行动应用软件发出 HTTP 请求给中间件后,首先服务路由器会依据请求的内容 URI, 对应到插件管理员所管理的插件; 管理员会先加载配置文件 (plugins.yaml)<sup>②</sup>, 查询其对应的插件是否存在, 如存在就将请求分配给插件; 插件接收到之后, 根据不同的动作先执行对应的解析方法 (on\_translating), 解析 HTTP 请求的参数; 当完成之后, 将参数传递给对应方法 (on\_mapping) 处理请求任务, 呼叫对应的模型处理与既有系统沟通以及存取数据; 最后再呼叫封装方法 (on\_wrapping) 将结果封装成特定的数据格式如 JSON<sup>③</sup> (JavaScript Object Notation)、XML<sup>④</sup> 等等, 加入到 HTTP 响应回传给行动应用软件, 即可完成一个整合既有系统功能的行动应用服务。

四、实作流程与结果讨论

本研究实际将设计架构套用在 Google News, 提供 Android 与 iOS 平台行动应用软件测试。在本节将以 GoogleNews 讯息为例, 说明整合既有系统功能的行动应用服务。

(一) 实作插件流程

新增一整合既有系统的行动应用服务时, 其开发流程图(图 8)及步骤如下:

- 步骤 1: 先进行需求分析, 确认实作内容;
- 步骤 2: 检查是否有无可用的插件, 若没有可用的插件则进入第 3 步骤, 反之进入第 6 步骤;
- 步骤 3: 检视有无可用的通用模型, 若没有则进入第 4 步骤, 反之进入第 5 步骤;
- 步骤 4: 开发扩充新的通用模型, 提供开发插件使用;
- 步骤 5: 开发新的套件, 并使用通用模型提供的模块与既有系统做沟通;

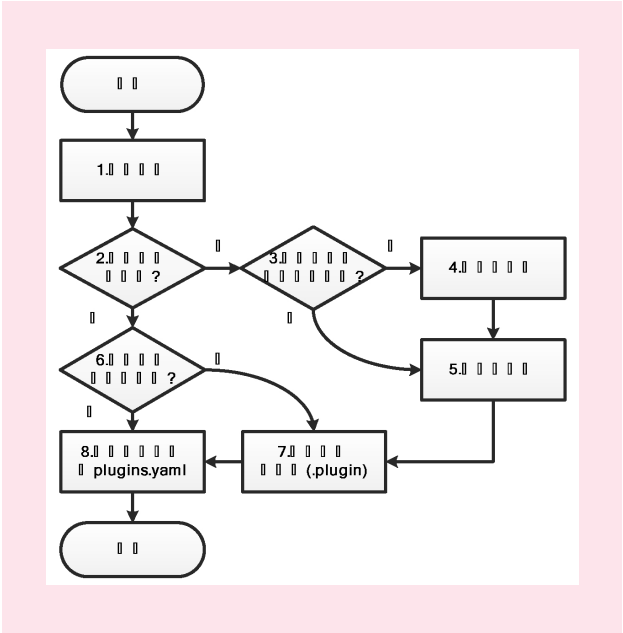


图 8 用中间件架构之开发流程图

- 步骤 6: 检查是否可直接使用插件, 若否则进入第 7 步骤, 反之进入第 8 步骤;
- 步骤 7: 设定插件配置文件, 使插件可以与欲整合的既有系统能够使用;
- 步骤 8: 在 plugins.yaml 中, 新增新的插件信息配置。

以上即是应用中间件架构之开发流程。

(二) 实作插件范例—Google News 讯息

GoogleNews 讯息功能主要是透过数据库取得 News 相关的讯息内容。先针对通用信息模型实作 BaseMessageModel 抽象类并且定义四种方法分别实现新增、取得、更新与移除讯息; 然后继承讯息抽象类, 实作 MessageModel 类别, 主要实作取得讯息功能; 而插件部分, 继承 BasePlugin 抽象类实作 MessagePlugin 类别, 在类别中仅实作

get\_on\_mapping 函式。

实作流程如下：首先 GoogleNews 讯息功能：①先进行步骤 1 需求分析,确定 Google 讯息所需要实作的内容；②然后步骤 2 检查是否有可用的插件可用，而由于是最先开发的功能所以一开始没有可用的插件,所以进入下一步骤找寻有无可用的模型；③步骤 3 经检视过后，行动中间件尚未建立取得 Google 讯息的通用模型,因此需要开发新的模型,④步骤 4 继承通用信息模型,实作 MessageModel 类别与取得 GoogleNews 讯息功能,根据不同的参数设定不同的条件,最后执行数据库查询并将结果回传。此外若需要新增、更新与删除讯息等功能,则只需实作对应的方法即可。⑤步骤 5 与步骤 7 在开发 News 讯息插件时，使用预设的解析与封装方法，因此在 MessagePlugin 的实作仅需完成 get\_on\_mapping 函式,主要呼叫 MessageModel 中取得讯息功能,然后再将结果回传即完成任务。⑥当插件与其插件配置文件完成之后，步骤 8 需要修改 plugin.yaml 新增此插件的配置,重载之后,即可将插件上线。程序代码程序片段如下所示。

程序代码(in-python)

```
class MessagePlugin(BasePlugin):
def get_on_mapping(self, **kwargs):
try:
check_api_key()
model=MessageModel()
messages=model.get_messages(kwargs)
count=message.count()
return dict(data=data,count=count)
except MappingHandlerError:
log.error('get on mapping error')
abort(403)
```

配置文件

[Core]

Name=news\_messages//对应 PluginManager 配置

文件的 Plugin 名称

Module=news\_messages//对应的 Plugin 文件名

[Documentation]

Authod=Shen

Version=1.0

Description=google message for news

当上述流程完成之后,即完成 GoogleNews 讯息的插件,然后利用此 Plugin 再开发相关行动应用程序新闻功能,GoogleNews 讯息功能在 iOS 与 Android 平台画面呈现方式如图 9 所示：

(三)结果讨论

以上实作可以发现应用此中间件设计架构与开发流程，实际上大多数功能都可以仅实作 get\_on\_mapping 函式以及对应的模型类别即可快速



图 9 iOS 与 Android 之 GoogleNews 讯息画面

完成，设计中还可以让开发人员清楚地分离商业逻辑提升重用性以及降低程序的耦合度。而透过配置文件的设定，也让开发人员可以达到可动态配置与加载功能,因此可以加速应行动 APP 的开发与大幅减少开发人员的负担。

## 五、结论

本研究中提出了一个行动中间件设计架构,用来降低企业开发行动服务应用程序的复杂度。藉由此设计架构,建置多种通用的模型,减少重复开发的成本,并且让开发人员只需专注在插件的设计,加快开发的速度。同时在插件框架中,加入了可配置的功能,让开发人员可以透过动态的调整,即可让插件的重用性增加。在此研究中,也利用所设计的行动中间件架构，仅需修改变插件组件配置参数的简单方式实作 Google News 讯息呈现于不同行动平台，证明此依设计确实让行动应用服务程序的开发可以更加的快速与容易。

注释：

①RESTful 路由设计是 Rails 的创新,它使用了 REST 概念来建立一整组的命名路由(named routes)。表象化状态转变 REpresentational State Transfer,简称 REST,众多网络服务中最为普遍的 API 格式，像是 Amazon、Yahoo!、Google 等提供的 API 服务均有 REST 接口。

②YAML 是一个可读性高，用来表达数据序列格式,它使用空白符号缩排和大量依赖外观的特色，特别适合用来表达或编辑数据结构、各种配置文件、倾印除错内容、文件大纲。

③JSON 是一种轻量级的数据交换语言,以文字为基础，且易于让人阅读。

④JSON 与 XML 最大的不同在于 XML 是一个完整的标记语言,而 JSON 不是。这使得 XML 在程序判读上需要比较大的功夫.XML 利用标记语言的特性提供了绝佳的延展性(如 XPath)，在数据储存，扩充功能及高阶检索方面具备对

JSON 的优势,而 JSON 则由于比 XML 更加小巧,以及浏览器的内建快速解析支持,使得其更适用于网络数据传输领域。

#### 参考文献:

- [1] 陈忠兴,何正得.网页式中间件之开放架构[J].工程科技与教育学刊,2011,(1):28-45.
- [2] IDC's Research.Asia/Pacific(Excluding Japan) 2011-2015 Cloud Services Forecast [EB/OL].[http://www.idc.com.tw/prodserv/cloud\\_computing.jsp](http://www.idc.com.tw/prodserv/cloud_computing.jsp),2011-12-12.
- [3] Wauters R.IDC Predicts 183 Billion Mobile App Downloads By 2015,Rise Of In-App Purchasing [EB/OL]. <http://techcrunch.com/2011/06/28/idc-predicts-183-billion-mobile-app-downloads-by-2015-rise-of-in-app-purchasing/>,2012-11-12.
- [4] Unhelkar B. and Murugesan S. The Enterprise Mobile Applications Development Framework[J].IT professional, 2010,(12):33-39.
- [5] Enea company. Embedded and Systems Management Middleware, Enea's technology report [EB/OL]. <http://www.enea.com/solutions/middleware/>,2015-05-25.
- [6] Bauer, F.L., Bolliet, L. and Helme, H.J. Report on a conference [Z].Nato Software Engineering Conference, 1968.
- [7] Blair, G.S., Coulson, G., Robin,P.,Papathomas,M.An Architecture for Next Generation Middleware [Z]. Proceedings of the IFIP International Conference on Distributed Systems Platforms and Open Distributed Processing,1998.
- [8] Almonaies, A.A., Cordy,J.R.,Dean,T.R.,Legacy system evolution towards service-oriented architecture [A]. SOAME 2010:International Workshop on SOA Migration and Evolution[C].Madrid;2010;53-62.
- [9] Evans C. C. The Official YAML Web Site (2012) [EB/OL]. <http://www.yaml.org/>,2015-02-01.
- [10]Hsueh N.L.,Ting D.H.,Shen Wen-Hsiang.and Lee W. T.A Configurable and Extensible Middleware Design for Mobile Application Integration [J].Journal of Software Engineering,2014,(8):1-13.

[责任编辑:刘 骋]

## Design of a Mobile Ware Framework for Rapid Mobile Application

SHEN Wen-xiang

(Department of Information Management, Overseas Chinese University, Taiwan 40721, China)

**Abstract:** Mobile applications can upgrade enterprises' work efficiency, competitiveness and even bring new potential customers. However, due to many heterogeneous systems and different mobile platforms, high cost of development and maintenance, it is not easy to develop the software for the legacy systems in mobile applications. Thus, using mobile middleware can facilitate in resolving most of the problems. The current mainstream platforms for mobile devices such as smart phones and tablet computers are iOS, Android, Windows Phone etc., but the skills for the implementation of mobile applications in each platform are different. It leads to high cost and requires more human labor when enterprises want to develop a mobile application software supporting multi-platforms. Middleware makes it easier for software developers to perform communication and input/output, mobile middleware essentially hides the complexities of working in mobile environments, so that it can reduce the difficulty of supporting multi-platforms, decrease the duplication of the development efforts and increase the opportunities of the developed components which can be reused. In this study, to design a mobile middleware framework can support quickly, development efficiently and integration conformably to the new mobile applications servers in the current system.

**Key words:** Middleware, Model-View-Controller model, Mobile APP, Plugin pattern