A Component Based Architecture for Web Content Management: Runtime Deployable WebManager Component Bundles

Jurriaan Souer and Martin van Mierloo
GX, Wijchenseweg 111, 6538 SW Nijmegen, The Netherlands
{jurriaan.souer, martin.van.mierloo}@gxwebmanager.com

Abstract

Customizing Web Applications is complex and result in possible complications when upgrading. The Web Content Management system 'GX WebManager' offers WebManager Component Bundles (WCBs): a new plug-in system for adding custom functionality runtime to the Web Application. WCBs consist of three, open source and open standards based technologies: Open Services Gateway initiative (OSGi), Java Content Repository (JCR) and Spring Model View Controller (MVC). In this demonstration we will show how to build and deploy a WCB based on an archetype and elaborate on the various forms of applications. The goal is to allow developers to easily create custom web applications plug-ins and share their application with the community on the one hand, and provide a library of custom functionalities for online business owners.

1 Background

Most public and private sector enterprises, faced with challenges in satisfying evolving customer needs and meeting regulatory and compliance dictates utilize Web Content Management (WCM) software to manage content on their internal and external Web sites. However, many organizations struggle with the implementation and change management processes of WCM systems, since the processes are complex and time consuming [3]. Although the Web Engineering research field has resulted in several methods to support the complex task of designing and creating Web Applications, the research on the implementation of WCM in the context of Web Engineering is scarce.

The WCM system 'GX WebManager' provides web content management and online functionalities such as personalization, application integration, multi-channel publishing, search engine optimization (SEO), digital asset management, online transaction processing, and WebTV for streaming media. Most of the Web application functionality built on the GX WebManager platform is configured without customization. However, online marketers always want some new functionalities (e.g. Web 2.0, Wiki, Semantic technologies). It is impossible to incorporate all new technologies within a standardized Web Content Management platform hence they are often customized to meet the business requirements. Customization is one of the most troublesome processes within WCM implementations. Customization can lead to higher implementation costs, longer implementation time, and could give problems when upgrading to a new version of the software.

2 Design goals

Confronted with these challenges with our proprietary WCM system – as described in [4] – we developed the design goals for the most recent version of GX WebManager – version 9. GX WebManager should facilitate a solid migration strategy and should allow for different solution frameworks (product verticals). Moreover, it should enhance customizations. We therefore defined the following design goals:

- Ease of Development
- Ease of Maintenance
- Ease of Deployment

We will elaborate on these design goals below.

2.1 Ease of Development

Developing custom functionalities should be easy for developers. Ease of development includes good debugging support, minimize the effort for adding new functionalities, use standards for development (i.c. Java, JSP), out-of-the-box support from software development kit Eclipse + Webtools, Java and Tomcat, and it should be easy to develop in teams within larger projects. To conclude, it should be easy to change the graphical user interface of GX WebManager based on standards.

978-0-7695-3261-5/08 $25.00 © 2008 IEEE
DOI 10.1109/ICWE.2008.32

2.2 Ease of Maintenance

A lot of organizations rely on their Web Applications for a large part of their business. Downtime for deployment, upgrades and failure should be minimized. Ease of maintenance consists of good support to analyze bottlenecks and to see if the issue (e.g. bug or performance) is part of the GX WebManager platform or the custom developed functionality. It should be possible to change, uninstall or upgrade functionalities without having to create a new deployment.

2.3 Ease of Deployment

Still, there are situations where new deployments are needed. However, these deployments should be as easy as possible to minimize the downtime and to allow different hosting providers to manage the deployments themselves. Moreover, easy of deployment encompasses easy deployment over a complete Development, Test, Acceptance and Production environment (DTAP support): when a custom functionality is developed in a development environment it should be very easy to install it in the Test environment without any configuration.

3 Implementing the Design Goals

To implement the design goals we introduced a new concept called a *WebManager Component Bundle* (or WCB): a plug-in system for adding new functionality at runtime to the product. WCBs are single (JAR)-files that can be easily exchanged and deployed on different GX WebManager installs. WCBs are developed using common Java development tools such as Eclipse and Maven. Developers can use archetypes (a sort of blueprints) to create various types of WCBs with a single command. This, in combination with a set of tutorials, videos, forums and a wiki on online development community makes it easy to get started.

WCBs incorporate three open technologies which allows for ease of development, ease of maintenance and easy of deployment:

- **OSGI**: The open standard OSGi [1], with the implementation Apache Felix [2]
- **JCR**: The open standard JCR or Java Content Repository (JSR-170/JSR-283[3]), with the implementation Apache Jackrabbit [4]. GX is a member of the JSR-283 Expert Group and official committer for Apache Jackrabbit.
- **Spring MVC**: the Model View Controller framework of the Spring Application Framework [5]

The following sections elaborate on these three technologies.

3.1 OSGi

OSGi technology provides a service-oriented, component-based environment for developers and offers standardized ways to manage the software lifecycle. OSGi is a well known technology in the mobile industry, but is rarely used as a server side technology. OSGi is used within GX WebManager to manage the lifecycle of the WCBs, it allows starting, updating, stoping and uninstalling WCBs at runtime. The OSGi guarantees the stability (it kills an WCB when its corrupt and makes sure that the platform is not influenced by the WCB). OSGi also includes version control which allows for automatic updates when a new version of the custom functionality is available. With OSGi, developers can easily develop their WCB locally and activate it on the Test environment using OSGi. The runtime deployment aspect and the stability contributes to the ease of deployment and ease of maintenance.

3.2 JCR

The Content Repository for Java Technology specification, developed under the Java Community Process as JSR–170 (currently being enhanced in its successor JSR–283) separates infrastructural services from application services. This unified API allows access to any compliant repository in a vendor- or implementation-neutral manner, which will prevent possible vendor lock-in. The JCR API is a type of Object Database that stores, searches and retrieves hierarchical data and allows true content repository infrastructure so that different applications can use the same interface for many purposes, making it universally accessible. A major advantage of JSR-170 is that it is not tied to any specific architecture. The back-end data storage for a JSR-170 implementation could be a file system, a WebDAV repository, an XML- backed system, or even an SQL database.

Within GX WebManager, the JCR is used as an persistence layer, including the data storage for the WCBs. Developers will be able to avoid the effort associated with learning the particular API of each repository vendor. Instead, programmers will be able to develop content-based application logic independently of the underlying repository architecture or physical storage. Using a common interface reduces both times and risk, in so much as a company will no longer need to rely on any one proprietary repository.

[1] http://www.osgi.org
[2] http://felix.apache.org/
[3] http://www.jcp.org/en/jsr/detail?id=283
[4] http://jackrabbit.apache.org/
[5] http://www.springframework.org/

3.3 Spring MVC

MVC (Model-View-Controller) is a design pattern that helps to separate user interface logic from business logic. Usage of a MVC based library offers a good approach to render the edit interface of GX WebManager 9 in a way that is supported by many open source standards. The current proprietary scripting language can contain both business logic and user interface logic in one and the same script. For the implementation GX has chosen the open source Spring MVC Framework. The inclusion of Spring MVC is the implementation of the design goal that developers should be able to easily change the user interface of GX WebManager.

In summary, we separated the platform from its functional components in GX WebManager 9 allowing developers to easy create new functionalities and deploy and maintain them at runtime. Modifying the platform implies checking if the other functionalities of the platform are still intact. Making changes to a component however, implies only checking if that specific component still behaves as expected. As a result, the engine of GX WebManager will have a higher stability and will have a non-changing interface with the components. Only individual WCBs can be modified and therefore be corrupted. This will have no impact on the complete website in general. In this demonstration we will show how to build and deploy a WCB based on an archetype and elaborate on the various forms of applications.

3.4 Combining the three technologies

This combination of three open standards and technologies results in a number of interesting characteristics. From a technical point of view the three technologies provides Lifecycle management, Persistence, Controller framework, Runtime deployment, Dependency management, Headless Services, View and edit presentation, OTAP support and merge conflict resolution with UUIDs, and Schema evolution.

But also the different roles in organizations that deal with web content management can benefit in various ways from WCBs. The business owner or marketing can quickly launch new functionality and be innovative. They are no longer dependant on the supplier's product roadmap and resources. When a WCB is already available they are even not dependant on developers. The IT department take advantage from the fact that managing a web content management system is easier and costs are lower since the platform is standard and additional functionality is added as standards based bundles, which have a predictable and protected behavior. In DTAP environments functionality and content can easily be transferred to other servers for testing purposes. And also developers have advantages: working with

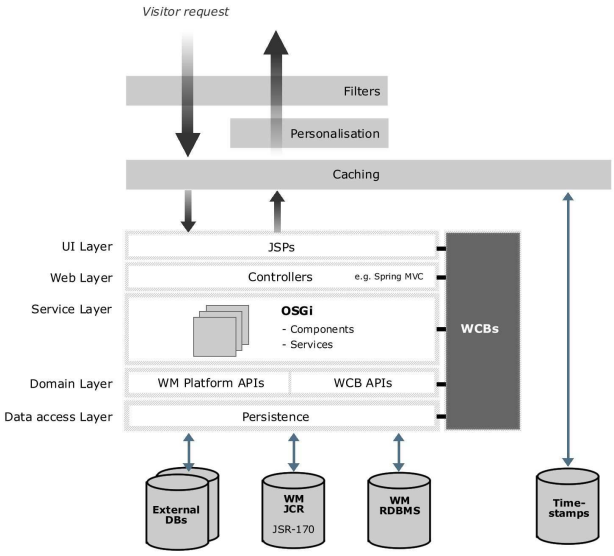

Figure 1. Application Architecture

WCBs is easy to learn because of the use of common tooling (e.g. Eclipse, Maven, Spring MVC, JSPs) and because there is an active development community. Develoment of WCBs is aimed at working in teams and reusing components and code.

Figure 1 shows the application architecture of GX WebManager. It consists of five layers: the user interface layer, web layer, service layer, domain layer and the data access layer. As illustrated, the JCR is used as the persistency storage. The OSGi is part of the service layer and Spring MVC is used in the Web layer as the MVC framework. WCBs can access all layers of the application and can be used for creating user interface (graphical design of a website), the web layer (for functions and panels), the service layer (for services and components), the domain layer (to access platform functions and other WCBs) and the data access layer. Each WCB has its own version number and certification which is centrally managed.

An example that explains the various stages of development and maintenance is a recently developed 'Calendar 2.0' WCB. This WCB was developed for a customer that an adapted calendar that contained more functionality then the default calendar component. The component required web 2.0 functionality (e.g. customer could enter and share calendar dates) and the persistence had to be open and extendible. Two developers worked on this WCB: one on the persistence and he created a set of functions that the other developer could use to store and retrieve information. The second focused on the presentation and the business logic. The

panel and elements were created in several minutes from archetypes (blueprints). A first beta release was added to a test environment and after several improvements the calendar WCB was taken into production. Additional improvements were added later on and the application manager simply upgraded the calendar WCB instantly even though the data model had been changed. From the initial idea to real life production this example took less than two weeks.

4 Discussion

Web Application Development can still be a complex task. With WCBs, we separated customized functions from the application platform itself, thereby improving the overall stability of the Web application and implementing the design goals. The WCBs can currently utilize six APIs to create customizations. We are improving these APIs to make even more rich functionalities and easy the development process. For example, we recently introduced 'Extendability' as a characteristic of a WCB. Developers can now take an existing WCB and change it slightly to his needs without code replication.

Both GX customers and implementation partners have adopted the WCB technology on a large scale. Customers benefit from the WCBs because it allows them to quickly launch new innovations and because their IT management can safely deploy them on testing and production environments. Larger customers have created development studios that create their own WCBs and the option to reuse and inherit from WCBs is widely used to keep improving WCBs. A lot of knowledge and best practices are being shared on the open development community[6] where developers from GX, GX customers and GX partners find and help each other. Several freelancers have invested time to become WCB developers and they offer their services on GX Developerweb. Over the last year several customers have even teamed together to develop WCB in a joined effort. This clearly shows that the development model is open and that no one has to rely on GX as a supplier anymore.

A year after introducing WCBs, they have shown to be successful in its many forms and applications. Even a simple WCB that displays a Youtube video or Google Maps can make a difference for a website editor. But also several intelligent and productive WCBs have emerged from GX, implementation partners and customers. Examples are statistic tooling (Google AdWords management), a multichannel WebTV component suite, a performance management console, newsfeed imports (Reuters, ANP), a database management console, advanced calendar, weblogs, etc.

A possible side effect is a WCB Ecosystem. WCBs are small, single files that can be easily exchanged within organizations, or even traded. Independent developers, commercial organizations and partners can use an online marketplace[7] to simplify finding, combining or building the right WCBs for the right solution.

5 Related Work

There are several organizations working on a similar strategy for component based software. Mozilla Firefox and Salesforce.com are two well known examples. Also in the research field, component (or packaged) based software development is a popular research area. The extent of software reuse depends upon the reuse strategy followed [2]. Two relevant work worth mentioning: Wulf et al describe a similar problem in software development and a runtime component-based tailorability in [5]. Arndt et al propose a composition of customized and component libraries to cope with the tension of software customization and standardized software [1].

References

[1] Jens-Magnus Arndt and Jens Dibbern. The tension between integration and fragmentation in a component based software development ecosystem. In *HICSS '06: Proceedings of the 39th Annual Hawaii International Conference on System Sciences*, page 228.3, Washington, DC, USA, 2006. IEEE Computer Society.

[2] Marcus A. Rothenberger, Kevin J. Dooley, Uday R. Kulkarni, and Nader Nada. Strategies for software reuse: A principal component analysis of reuse practices. *IEEE Transactions on Software Engineering*, 29(9):825–837, 2003.

[3] Jurriaan Souer, Paul Honders, Johan Versendaal, and Sjaak Brinkkemper. Defining operations and maintenance in web engineering: A framework for cms-based web applications. In *IEEE ACM Second International Conference on Digital Information Management ICDIM 2007*, pages 430 – 435, October 2007.

[4] Martijn van Berkum, Sjaak Brinkkemper, and Arthur Meyer. A combined runtime environment and web-based development environment for web application engineering. In *Proceedings of the Advanced Information Systems Engineering. CAiSE04*, pages 307–321. LNCS 3084, 2004.

[5] Volker Wulf, Volkmar Pipek, and Markus Won. Component-based tailorability: Enabling highly flexible software applications. *Int. J. Hum.-Comput. Stud.*, 66(1):1–22, 2008.

[6] http://www.gxdeveloperweb.com

[7] http://www.wcmexchange.com