WIKI-WS AS A C^{2}NIWA WEB SERVICE MANAGEMENT PLATFORM

MAREK DOWNAR

Gdansk University of Technology
Narutowicza 11/12, 80-233 Gdansk, Poland

(received: 19 May 2015; revised: 22 June 2015;
accepted: 1 July 2015; published online: 1 October 2015)

Abstract: The Wiki-WS platform was implemented within the C^{2}NIWA project for production purposes. Wiki-WS stands for developing, managing and maintaining web services. The production deployment needed implementation of several functional improvements and establishing a strong security&safety policy. The WikiWiki-WS platform has to be used as an educational environment for developing web services and a production environment for execution of advanced web services using the computation capacity of the newly established supercomputer – TRITON. In the article the Wiki-WS architecture, security methods and results of real environment tests are presented.

Keywords: web service, WikiWS, service oriented architecture

1. Introduction

Despite the existence of many repositories containing information of web services there are only few that support searching and executing. Migration of web services between servers demands updating repositories about its last localization. Many repositories contain only the service location without having any information about its maintenance ratings.

Examples of projects similar to Wiki-WS with an open access policy include Xmethod [1] and WebServiceX [2]. The commercially available projects are GeoNames [3], XigniteGlobalMaster [4], StrikeIron [5], CDYNE [6]. There are also many web services used as interfaces to complicated systems (*e.g.* bank transaction systems).

A mature web services repository solution should provide a constancy of service localization with the ability of execution and assessment by developers and service users. The Wiki-WS platform [7] has been created to meet these requirements.

SOA (Service Oriented Architecture) implementations often base on services changing the localization and interfaces. Every change made by the web service

developer results in the necessity of making changes in the application using it. Therefore, implementations of SOA applications base on web services providing entry points to complicated systems developed by a single company. The existence and immutability of these services determines the correctness of application execution.

In real environment communication between service providers and application developers is reduced to informing developers about interface changes. Often changes are made top-down without communication and developers are informed about it while applications fail to execute due to calling a non-existent web service.

A C^{2}NIWA Wiki-WS platform team has tested and provided environment that allows web service monitoring and versioning so that applications using them could work constantly without the necessity of making changes and application developers would be informed that a new version of a web service is released.

2. Service monitoring

The main problem in SOA applications is the web service localization and interface evolution. Application developers have to be informed and applications should be changed when changes in a web service are made.

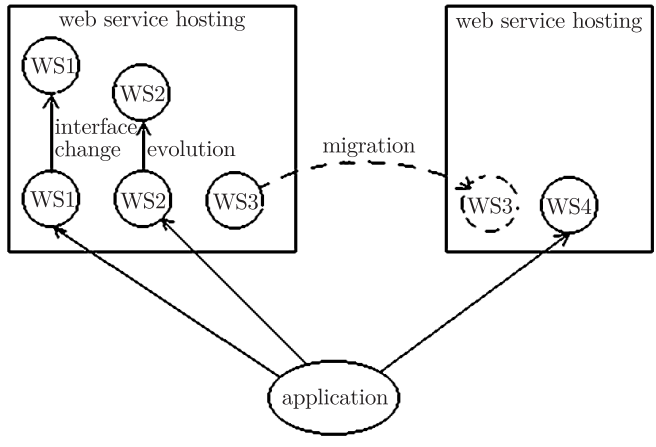

Figure 1. Problems with web service applications

Figure 1 presents most common problems of applications using web services: change of interface, evolution and migration. The occurrence of these situations causes ambiguity (evolution) and errors in execution (change of interface, migration) and requires changes in applications. These problems have been resolved by establishing a centralized Web Service platform allowing execution and monitoring of web service builds deployed on integrated Java and .NET servers. Information of any web service change and availability of a new version is automatically forwarded to every user involved in the project.

Additionally web service recommendation mechanisms based on subjective notes given by users and objective ratings automatically calculated during the

MOST WIEDZY Downloaded from mostwiedzy.pl

web service maintenance process are implemented on the platform. Information about successful and failed execution is stored in log files and the Wiki-WS database. It is used to provide real time monitoring statistics that are used by the recommendation engine that is proposing web services to users.

We provide standard maintenance factory metrics to the web service world:

- MTBF (Mean Time Between Failure) – a measure that informs about the average time in which the web service works correctly;
- MTTR (Mean Time To Repair) – the mean time between a web service failure detected and reported by Wiki-WS and the repair made by the team of developers;
- ALDT (Administrative and Logistic Downtime) – MTTR with the delay response time of the developers-system.

We also provide new metrics for the Knowledge Acquisition System:

- SFER (Success to Fail Execution Ratio) – every execution of a web service is logged and reported as successful or failed. The ratio of these statistics shows users information about how much they can rely on a given web service;
- MET (Mean Execution Time) – the average web service execution time shows users how long they should wait for the result.

Wiki-WS is collecting also web service development-time statistics to foresee the need for additional load balancing servers:

- CT (Compilation Time) – the time consumed by the compilation process (csc or javac),
- PRT (Project Ready Time) – an application pool or domain creation and applying security policy to a newly created project,
- DT (Deployment Time) – the time consumed for the web service deployment on Microsoft IIS or Glassfish;
- RT (Response Time) – the response time between the user action and the Wiki-WS response.

A web service in the Wiki-WS platform is a project. Every project has its developers, a subversion server and a deployment server. Within the project, developers can configure compilation presets. Based on the presets and code revision, developers can compile and deploy many web services on production servers.

Figure 2 presents the life cycle of a web service [8] project:

- create project – a project can be created by importing from Redmine deployed at `https://projects.os.niwa.gda.pl` or by using the "Create Project" form from the left menu of the Wiki-WS website. While creating a project the user should provide a unique name, a description and choose the web service technology, the deployment server and the subversion server. They also should specify, if the project is publicly available;
- create repository – when the project creation process ends a repository on the chosen subversion server is created automatically. The address to the repository

MOST WIEDZY Downloaded from mostwiedzy.pl

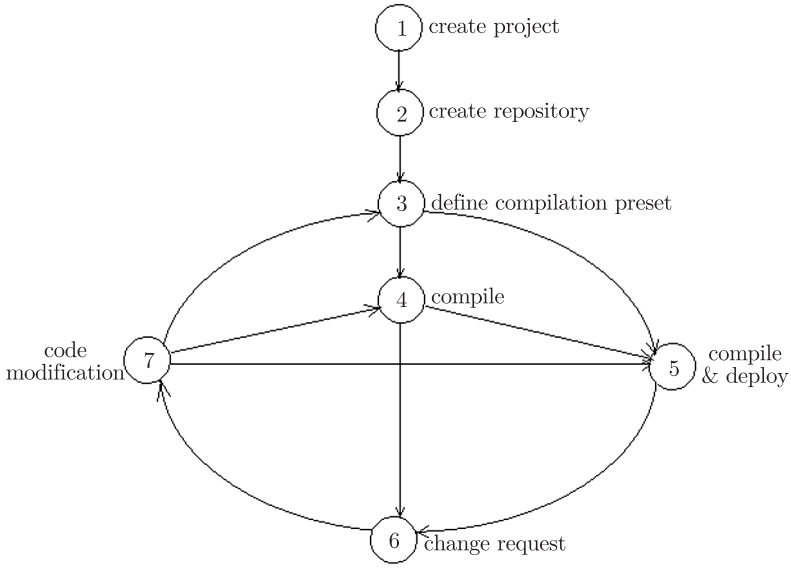

Figure 2. Web service life cycle on Wiki-WS platform

where the user should commit the first version of the code is given in the project details. Connection to the subversion server can be made by commonly available SVN clients;

- define compilation preset – provide compilation information such as the main class, directory where the code exists;
- compile – compile the code giving a revision version of the code and compilation preset, after successful compilation the web service is available to publish on the production server. When compilation errors occur they are listed in the Wiki-WS interface near the build;
- compile & deploy – if the user is sure of the web service, he/she can compile the code and publish the build directly on the server, he/she can use it by simple checking the "publish on server" check box in the Wiki-WS interface;
- change request – should flow from Redmine or be made directly by developers;
- code modification – can be done by a SVN client or by the Wiki-WS Web Editor. Changes are committed to the repository giving a new code revision to compile and deploy.

A project compiled on the Wiki-WS platform is called a build. A project can have many builds which can be deployed on the production server. Every deployed build has its own unique URL format `https://wikiws.os.niwa.gda.pl/service/project_name/build_name`. Publishing a new build does not overlap the old build, so applications using the old build can work unchanged. For every build there are copies of libraries stored in build directories on the deployment server. Every build has its own libraries and a new build does not get involved in any previous build. Application developers are informed of changes in web services

MOST WIEDZY Downloaded from mostwiedzy.pl

and publishing a new build using an internal messaging system so that they can choose the build version appropriate for their applications.

3. Wiki-WS platform architecture

Wiki-WS has a distributed architecture presented in Figure 3 which is composed of:

- Wiki-WS Website server – the entry point where the user can manage and publish his/her projects. The Windows 2012R2 server, IIS, the MSSQL 2008R2 Server and the MySQL server. It is also here that the statistics collection, the logging mechanisms and the proxy server that is responsible for passing requests to other servers are located;

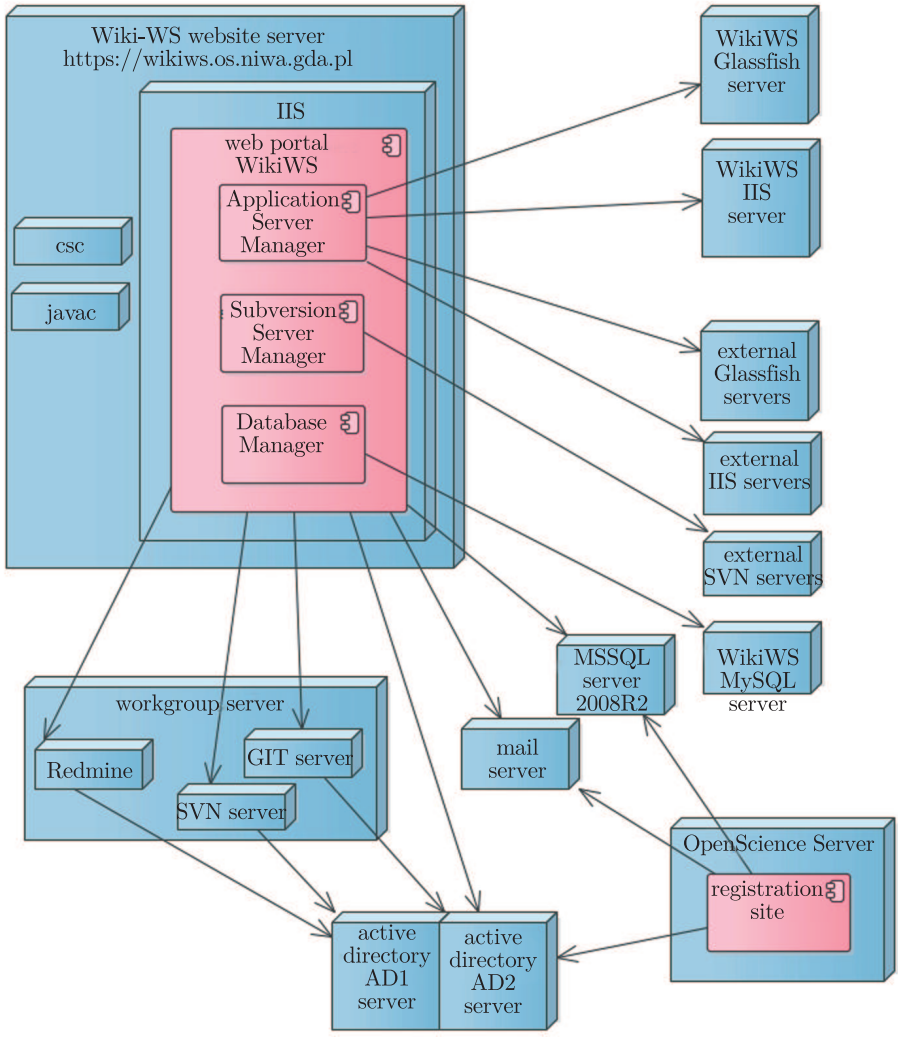

Figure 3. Wiki-WS platform architecture

MOST WIEDZY Downloaded from mostwiedzy.pl

- IIS server – a Windows Server where projects using the .NET technology are deployed and published. Windows 2012R2 with IIS configured for production purposes;
- Glassfish server – a Scientific Linux server with a Glassfish server where projects using the Java technology are deployed and published;
- Redmine server – a Scientific Linux server where Redmine is deployed;
- SVN server – a Scientific Linux server where the SVN server is deployed;
- Active Directory server – a double Windows Server storing user data and privileges.

In Figure 4 the Wiki-WS Website architecture is presented. It is composed of given subsystems:

- SubersionServerManager – contains connectors and subversion clients for managing SVN and Git servers used on the platform. In the simplest implementation it contains only one SVN server but it allows multiple servers to be connected;
- ProjectManager – responsible for giving users the ability to manage their projects, defining the SVN server and the application server for the project, defining compilation presets, producing and deploying builds. It is also responsible for collecting statistics information of the web service execution;
- ApplicationServerManager – responsible for managing remote Glassfish and remote IIS servers. In the simplest implementation it contains only one Glassfish and only one IIS server but it allows multiple servers to be connected. It is also ready to connect a Tomcat server;
- UserManager – responsible for managing the user session and managing user permissions for any operation in Wiki-WS. It allows managing the user privileges defined for application servers, subversion servers and projects;
- CompilationEngine – contains wrappers for csc.exe and javac compilers, responsible for web service code compilation and building deployment packages for the .NET and Java environment;
- NotificationEngine – provides a messaging interface for communication between users within Wiki-WS. Notifications can be stored in Wiki-WS or stored in Wiki-WS and sent by an e-mail server to the user,
- SearchEngine – allows searching for a web service deployed within Wiki-WS [9];
- RedmineConnector – responsible for communication between Wiki-WS and Redmine `https://projects.os.niwa.gda.pl`.

Wiki-WS is an environment ready to handle structured work groups composed of:

- developer – the programmer that develops the web service, has access to the subversion server where the code can be submitted, can define compilation presets. The programmer can also compile a build and deploy it on the application server;
- tester – can invoke a web service and report a change request with Redmine;

MOST WIEDZY Downloaded from mostwiedzy.pl

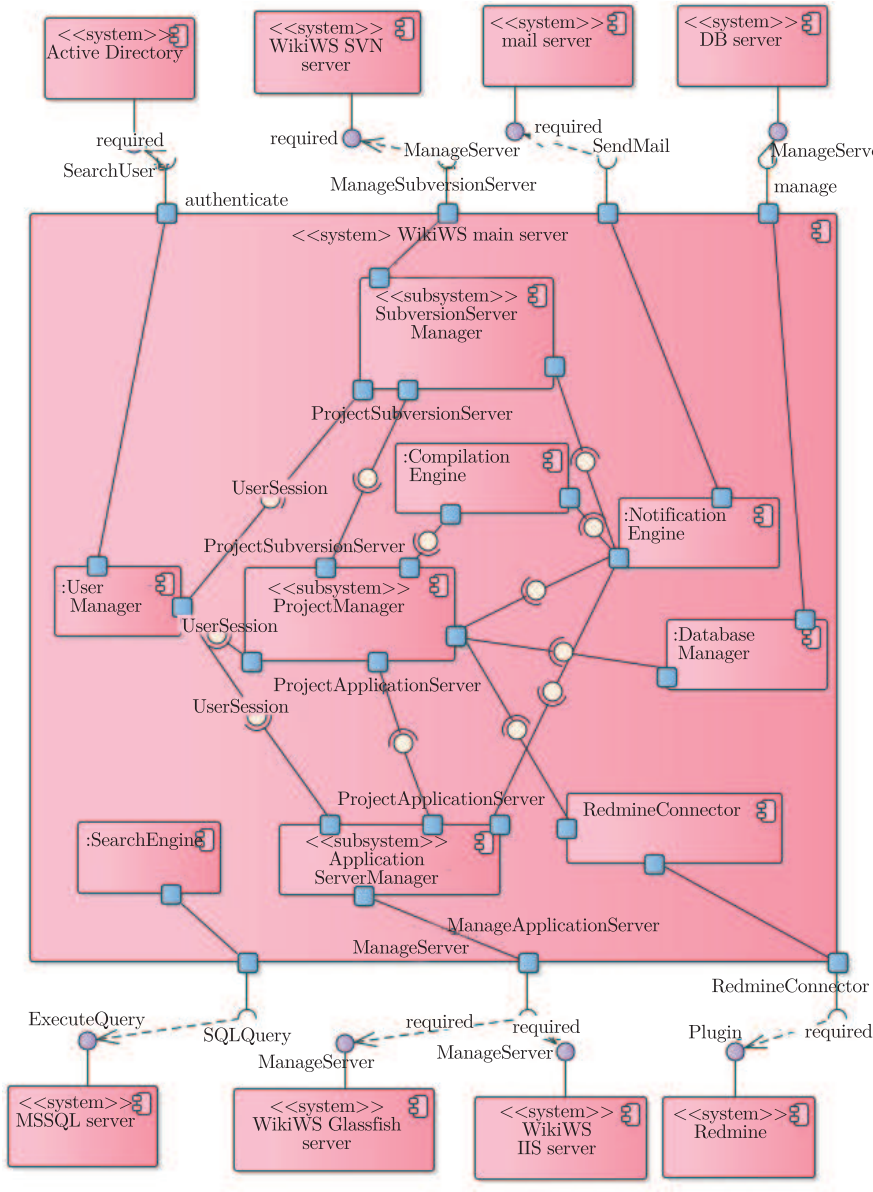

Figure 4. Wiki-WS Website programming structure

- manager – manages users participating in a project, defines descriptions and checks the project statistics, can report change requests with Redmine;
- subversion server owner – Wiki-WS has the ability for joining external subversion servers. Their owners can manage users and their privileges;
- application server owner – Wiki-WS has the ability to join external application servers. Owners of them can manage users and their privileges.

A hierarchy of actors is presented in Figure 5.

MOST WIEDZY Downloaded from mostwiedzy.pl

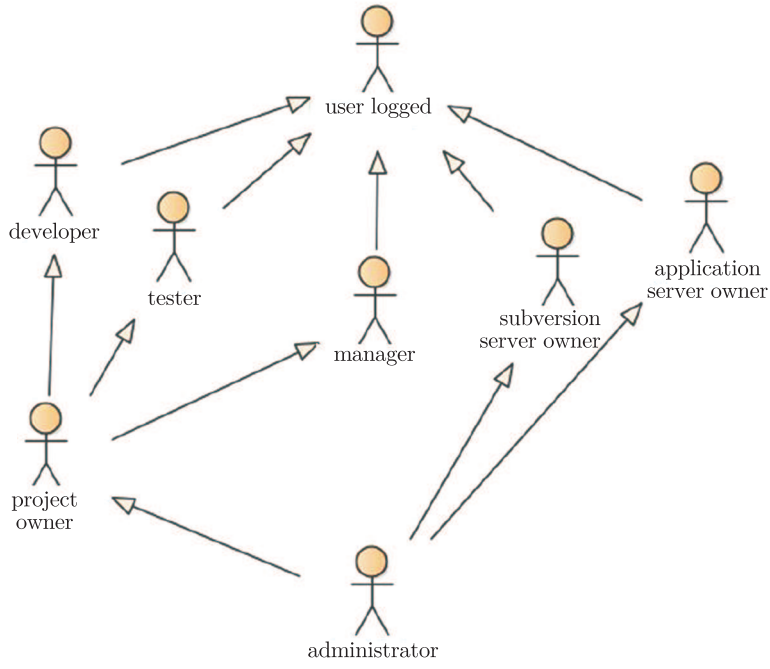

Figure 5. Hierarchy of Wiki-WS platform actors

4. Web service execution security

The Wiki-WS platform security implementation was a major challenge. It was done by implementing an advanced server infrastructure and security and safety policy of communication protocols used between servers.

Every .NET web service deployed on the IIS server has its own application pool that allows access to its own directories only. It is done by executing a special icacls command just after the application pool is created. icacls is responsible for modifying the discretionary access control list (DACLs) on specified files, and applies stored DACLs to files in specified directories [10]. The application pool is created during the first web service deployment and every build of this web service belongs to it. In this way there is no possibility to access another web service code or data which was successfully tested on the production server. The application pool is configured to throttle, if any web service belonging to it is consuming the CPU over a given limit which is also done if the web service is consuming much memory. Application pools between the Wiki-WS Website and the IIS deployment server are managed by the ServerManager object from the Microsoft.Web.Administration library using impersonation .NET mechanisms. The publishing project is made by FTPEngine using a secure FTP. Every pool has a shutdown limit of 30 sec and a startup limit of 30 sec.

Every Java web service deployed on the Glassfish server has its own secured domain. Managing the domain is done by SshGlassfishConnector through the ssh tunnel using the Glassfish asadmin script. Every build of the web service is

MOST WIEDZY Downloaded from mostwiedzy.pl

assigned to the project domain. Every domain has an admin port for managing and a service port for calling the web service. Application Pools on IIS and domains on Glassfish are created at the time of the project acceptance by the Wiki-WS admin.

Every server in the Wiki-WS platform stands behind the nginx server which is a proxy that filters and forwards user requests to a specific server. It rewrites WSDLs (Web Services Description Language) of the web services deployed on servers hiding their IP address. Thus, every request is passed through the nginx server where logging and statistics mechanisms are also implemented.

5. Test results and future work

Real world tests were performed in 8 groups of developers composed of 12–15 students each. The task was to develop and deploy web services on Wiki-WS Glassfish and the IIS server and implement an application using them. The compile and deploy process of web services on application servers oscillated between 3 and 200 seconds and no major bugs were found. More statistics will be provided when more web services are deployed and more users using them registered. For the future work we expect to introduce support for RESTful (Representational State Transfer) web services which have less complexity and are easier to implement than classic SOAP (Simple Object Access Protocol) web services. We will also provide mechanisms that could allow applying fees for the use of web services.

References

[1] XMethods Website 2015, http://www.xmethods.net
[2] WebserviceX.NET Website 2015, http://www.webservicex.net
[3] GeoNames Website 2015, http://www.geonames.org
[4] Xugbute Website 2015, http://www.xignite.com/Products/Catalog.aspx
[5] StrikeIron Website 2015, http://www.strikeiron.com
[6] CDYNE Professional REST and SOAP API Provider Website 2015, http://www.cdyne.com
[7] Krawczyk H and Downar M 2012 *Commonly Accessible Web Service Platform – Wiki-WS, Intelligent Tools for Building a Scientific Information Platform, Studies In Computational Intelligence*, Springer 251
[8] 2012 *Wiki-WS – Source Code Repository and Execution Environment for Web Services*, *ICT Young* 607
[9] Sobecki A and Downar M 2012 *Web Component for Automatic Extraction of Ontological Information from Informal Description of Web Services*, *ISAT*
[10] WindowsServer Website 2015, https://technet.microsoft.com/en-us/library/cc753525.aspx

MOST WIEDZY Downloaded from mostwiedzy.pl

MOST WIEDZY Downloaded from mostwiedzy.pl