GUJARAT TECHNOLOGICAL UNIVERSITY
(GTU)
INNOVATION COUNCIL (GIC)
Patent Search & Analysis Report
(PSAR)

Date of Submission : 25/09/2016

Dear Rathod Harshrajsinh Vijaysinh,

Studied Patent Number for generation of PSAR : **16BE7_130020107086_4**

PART 1: PATENT SEARCH DATABASE USED

1. Patent Search Database used : Google Patents

Web link of database : https://patents.google.com/

2. Keywords Used for Search : ERP,Website,Database,Backend

3. Search String Used : ERP Website And Database Backend design

4. Number of Results/Hits getting : 2499

PART 2: BASIC DATA OF PATENTED INVENTION /BIBLIOGRAPHIC DATA

5. Category/ Field of Invention : Computer/IT Engineering

6. Invention is Related to/Class of Invention : Related to generating secure Web service architectures

6 (a) : IPC class of the studied patent : G06Q 10/10 (20130101); H04L 67/16 (20130101); H04L 67/02 (20130101); H04L 63/20 (20130101); G06F 9/4

7. Title of Invention : System and method for generating secure Web service architectures using a Web Services security assessment methodology

8. Patent No. : US8346929B1

9. Application Number : 10642928

9 (a) : Web link of the studied patent : https://patents.google.com/patent/US8346929B1/en?q=erp&q=website&q=database&q=backend

10. Date of Filing/Application (DD/MM/YYYY) : 18/08/2003

11. Priority Date (DD/MM/YYYY) : 18/08/2003

12. Publication/Journal Number : US 709/226

13. Publication Date (DD/MM/YYYY) : 01/01/2013

14. First Filled Country : Albania : United States

15. Also Published as

Sr.No	Country Where Filled	Application No./Patent No.
1	United States	6738908
2	United States	6792605

16. Inventor/s Details.

Sr.No	Name of Inventor	Address/City/Country of Inventor
1	Lai Ray Y	Fremont, CA

17. Applicant/Assignee Details.

Sr.No	Name of Applicant/Assignee	Address/City/Country of Applicant
1	Oracle America Inc	Redwood City, CA

18. Applicant for Patent is : University

PART 3: TECHNICAL PART OF PATENTED INVENTION

19. Limitation of Prior Technology / Art

The Web Service system as recited in claim 1, wherein, to generate the Web Service architecture in accordance with the Web Services security assessment structured methodology, the Web Service architecture design system is configured to: determine one or more security components of the Web Service architecture according to one or more Use Case requirements for the Web Service architecture; determine one or more Web Service objects of the Web Service architecture to be protected; define an object relationship for security protection in the Web Service architecture; determine one or more associated trust domains, security policy and strategy, and one or more threat profiles for the Web Service architecture; determine one or more protection schemes and measures for the Web Services objects; and apply one or more Web Services design patterns including the one or more security design patterns to the Web Service architecture.

20. Specific Problem Solved / Objective of Invention

Web Services technologies enable the reuse of business functionality provided by mainframes and legacy systems. They help protect past investments of business functionality developed on legacy and proprietary platforms and ease building "killer" applications based on existing customer and account data kept by these legacy systems. "Killer" applications may create user stickiness by aggregating useful and timely customer and account information from different data sources that may run on legacy systems using Web Services as the technology enabler.

21. Brief about Invention

FIG. 1 illustrates the Web Services consumer-service provider relationship according to one embodiment.

FIG. 2 illustrates an exemplary complete Web Services application according to one embodiment.

FIG. 3 illustrates an exemplary Membership Award scenario according to one embodiment.

FIG. 4 illustrates business scenarios or use cases for the membership award processes according to one embodiment.

FIG. 5 illustrates an exemplary Membership Award Sequence Diagram according to one embodiment.

FIG. 6 illustrates an exemplary Business-to-Business Payment Services scenario according to one embodiment.

FIG. 7 illustrates business scenarios or use cases for the payment services according to one embodiment.

FIG. 8 is a Payment Services Sequence Diagram according to one embodiment.

FIG. 9 illustrates different layers of the Web Services technology stack according to one embodiment.

FIG. 10 presents a typical scenario for using Web Services according to one embodiment.

FIG. 11 illustrates Web Services use cases according to one embodiment.

FIG. 12 is a Web Services sequence diagram according to one embodiment.

FIG. 13 illustrates different areas of Web Services security according to one embodiment.

FIG. 14 illustrates a process for bringing together the various technologies described so far in order to build a workable Web Services solution according to one embodiment.

FIG. 15 illustrates a process for Web Services-enabling an application or applications according to one embodiment.

FIG. 16 illustrates an exemplary Web Services scenario according to one embodiment.

FIG. 17 illustrates an exemplary Web Services architecture using Sun ONE Framework according to one embodiment.

FIG. 18 illustrates an exemplary detailed Web Services architecture according to one embodiment.

FIG. 19 illustrates an example of a Web Services development life cycle using the Unified Process development methodology.

FIG. 20 illustrates a server-level architecture view of a securities trading (or brokerage) firm that adopts Web Services technology according to one embodiment.

FIG. 21 elaborates on the architecture diagram in FIG. 20 and depicts the logical components in each server according to one embodiment.

FIG. 22 is a table that shows an exemplary tiers vs. platform layers analysis, according to one embodiment.

FIG. 23 is a Quality of Services analysis matrix, according to one embodiment.

FIG. 24 illustrates the logical process of SOAP cache according to one embodiment.

FIG. 25 illustrates four Use Cases for managing a SOAP cache according to one embodiment.

FIG. 26 is a SOAP cache sequence diagram.

FIG. 27 illustrates an exemplary case of an investment manager placing a trade order with a brokerage firm.

FIG. 28 illustrates five business scenarios or business cases according to one embodiment.

FIG. 29 is a JMS Bridge sequence diagram according to one embodiment.

FIG. 30 illustrates an exemplary scenario with four instances of SOAP servers, each of which uses a separate IP port number, according to one embodiment.

FIG. 31 illustrates an exemplary scenario using three SOAP server machines connected to a HTTP load balancer according to one embodiment.

FIG. 32 illustrates exemplary State Management using RPC-based Web Services calls according to one embodiment.

FIG. 33 illustrates six business scenarios or Use Cases according to one embodiment.

FIG. 34 is a State Management Sequence Diagram according to one embodiment.

FIG. 35 illustrates an exemplary scenario where the SOAP server (SOAP reply) generates a logging event before it initiates a SOAP-RPC call or a document-based Web Services call according to one embodiment.

FIG. 36 illustrates four Use Cases for transaction logging according to one embodiment.

FIG. 37 is a SOAP Logger Sequence Diagram according to one embodiment.

FIG. 38 illustrates an example of clustering the hardware platform of multiple Service Registries according to one embodiment.

FIG. 39 illustrates deployment scenarios for both a public UDDI Service

Registry and a private UDDI Service Registry according to one embodiment.

FIG. 40 is an example of a staging Service Registry according to one embodiment.

FIG. 41 illustrates an exemplary design of a CTG running on the same platform with CICS and the Web server according to one embodiment.

FIG. 42 illustrates an exemplary design of a CTG running on a different host that communicates with CICS applications on a z/OS host according to one embodiment.

FIG. 43 illustrates an exemplary design of a remote CTG according to one embodiment.

FIG. 44 illustrates some design configurations that may be used when using CWS according to one embodiment.

FIG. 45 illustrates CWS Direct Connection according to one embodiment.

FIG. 46 illustrates the interaction process between components using the CICS Web Server Plug-in according to one embodiment.

FIG. 47 illustrates the interaction process between components using the 3270 Web Bridge according to one embodiment.

FIG. 48 illustrates CICS EJB Support according to one embodiment.

FIG. 49 illustrates an exemplary high-level application architecture for a SOAP Proxy on a Mainframe according to one embodiment.

FIG. 50 is a table of Integration Points for Mainframe Interoperability, according to one embodiment.

22. Key learning Points

Embodiments of a generic Web Services architecture may provide a repeatable and consistent way to design and deploy scalable, reliable Web Services, independent of the underlying vendor products. Embodiments may provide a vendor-independent architecture framework to design Web Services and to bring different technology pieces together in a big, complete picture. Embodiments may include best practices of delivering Web Services solutions with Quality of Services.

Web Services design patterns and when-to-use architecture principles are described. The Web Services design patterns and best approaches address the different needs of infrastructure architects, J2EE developers, security architects, and integration architects. In one embodiment, Web Services design patterns may be designed based on Quality of Service principles. Embodiments may be used in designing and implementing Quality of Services (the so-called "ilities") for reliable, available, and scalable Web Services. One embodiment may provide a Business-to-Business Integration (B2Bi) integration framework for Web Services. In this embodiment, one or more of the design patterns may be extended to B2Bi.

Embodiments may provide a Web Security framework. Embodiments may provide a security framework to design end-to-end Web Services security. Embodiments may address security at different levels, from network level, infrastructure level, message level, to application level, and may bring different security technologies together in the security framework.

Embodiments of a system and method for providing a structured methodology and design patterns for implementing Web Services may include one or more of, but are not limited to: Deployment (Quality of Service) Scalability design patterns--e.g. SOAP server farm (load balancing SOAP requests), SOAP cache, multiple servlet engines, proxy/gateway, etc. Reliability design patterns--e.g. session management, state management, SOAP logger, etc. Availability design patterns--e.g. redundant SOAP servers, high availability service registries, etc. Service Registry Service versioning and registry management. Registry deployment (e.g. centralized and federated). Publish, unpublish to registry--JAXR. Synchronization of registries (content management). Integration Application-to-application patterns. Standard build design pattern. EAI design pattern--e.g. hub-spoke, replication, federated replication, multi-step application integration, etc. Data exchange design patterns. Process integration design patterns--e.g. closed process, open process, etc. Broker integration design patterns--e.g. service consolidation broker, reverse auction broker, etc. Security Protecting Web Services objects. Cross-domain single sign-on.

In this document, design patterns are defined in structured pattern format (e.g. context, problem, force, and solution) and are further described using Unified Modeling Language (UML) notation (e.g. sequence diagrams).

Embodiments of the Web Services architecture are generally described herein using Sun's Web Services technology (for example, JWSDP and JAX) with a Sun ONE architecture and J2EE flavor. Note, however, that embodiments are not limited to these technologies, and may be implemented with other Web Services technologies.

23. Summary of Invention

Embodiments of a system and method for generating a generic, vendor-independent secure Web Services architecture incorporating a Web Services Security Assessment structured methodology and security design patterns for designing and implementing secure Web Services are described. Lifecycles of the Web Services Security Assessment structured methodology may include one or more of, but are not limited to: vision and strategy, architecture design, development, integration, and deployment. In one embodiment, in the Vision and Strategy Web Services life cycle, architects may collect user security requirements and technical security requirements, and encapsulate them into use case requirements using Use Case modeling techniques. Architects may identify a set of Web Services objects that need to be protected and secured, and their associated relationship in the context of the deployment infrastructure. In the Architecture Design life cycle, architects may define trust domains, define security policy, and identify potential security threats. In the Development life cycle, architects may develop protection measures or security application codes to protect the Web Services objects and components. If necessary or desired, architects may apply one or more Web Services security tools. In the Integration life cycle, architects may apply one or more Web Services security design patterns to integrate different Web Services components together. In the Deployment life cycle, architects may deploy the Web Services infrastructure in accordance with the generated secure Web Services architecture. Security of the deployed Web Service may then be assessed.

In one embodiment of a Web Services Security Assessment methodology design process, one or more security components may be identified and implemented based on one or more use case requirements. The Web Services objects or components that need to be protected may be identified. The object relationship for security protection may be defined, and the associated trust domains, security policy and strategy and threat profiles may be identified. A set of protection schemes and measures for these Web Services objects may be derived. One or more supporting Web Services (security) tools may be applied to complete the security protection schemes, if necessary. Web Services design patterns, including security design patterns may be applied wherever appropriate. In some cases, re-architecting or re-engineering may be desired or required. Upon deployment to production, the security levels may be assessed by tiers, e.g. host scan and host security health checking.

One embodiment may be implemented as a Secure Web Services architecture design mechanism. The secure Web Services architecture design mechanism may receive Web Services requirements as input and, using the input, assist a user in designing and generating a secure Web Services architecture using the Web Services Security Assessment methodology and design patterns including security design patterns. A Web Services infrastructure may then be deployed or implemented in accordance with the secure Web Services architecture.

24. Number of Claims	:	57
25. Patent Status	:	Published Application

26. How much this invention is related with your IDP/UDP?

< 70 %

27. Do you have any idea to do anything around the said invention to improve it? (Give short note in not more than 500 words)

The various methods as illustrated in the Figures and described herein represent exemplary embodiments of methods. The methods may be implemented in software, hardware, or a combination thereof. The order of method may be changed, and various elements may be added, reordered, combined, omitted, modified, etc.
Various modifications and changes may be made as would be obvious to a person skilled in the art having the benefit of this disclosure. It is intended that the invention embrace all such modifications and changes and, accordingly, the above description to be regarded in an illustrative rather than a restrictive sense.