Service-oriented architecture for ship automation and control system

Myo Min Swe

Saint- Petersburg State Marine Technical University

3, Lotsmanskaya Str., St.Petersburg, 190008

e-mail: deg@csa.ru, eltson@gmail.com

ABSTRACT

In this paper we discuss a Service-Oriented Architecture (SOA) for building ship automation and control, made out of loosely-coupled and distributed web services. The proposed architecture consists of three elementary tiers: the client tier that corresponds to any computing ship control system; the server tier that corresponds to the web services that deliver the basic functionalities for the client tier; and the middleware tier that corresponds to an enterprise service bus that promotes interoperability between all the interconnected entities. Ship automation and control system was simulated and experimented and it successfully exhibited the desired features of SOA. Future research can improve upon the proposed architecture so much so that it supports encryption for securing the exchange of data between the various communicating entities of the system.

Keywords

Service-Oriented Architecture (SOA), web services, enterprise service bus – ESB, ship automation & control system.

Introduction

Computing technologies are becoming more pervasive day after day, offering new potentials for automating tasks in many challenging applications. There I want to discuss about Service-Oriented Architecture (SOA) for building ship automation and control system using service software components. Such as, machinery management, navigation, maneuvering, power management, cargo management, propulsion control, ballast management, thruster control and alarm management. In effect, many of these ship components and equipment are highly computing intensive systems that use complex embedded software and algorithms to handle computational and data-intensive tasks.

Ship Automation & Control System

On a ship there are many parameters that needs to be controlled or monitored including: temperatures, pressure, level, viscosity, flow control, position of vessel, speed, torque control, voltage, current, machinery status (on/ off), and equipment status (open/ closed). A modern automation and control system is a fully integrated system covering many aspects of the ship operation that includes the propulsion plant operation, power management operation on the auxiliary engines, auxiliary machinery operation, cargo on-and-off-loading operation, navigation and administration of maintenance and purchasing of spares [1].

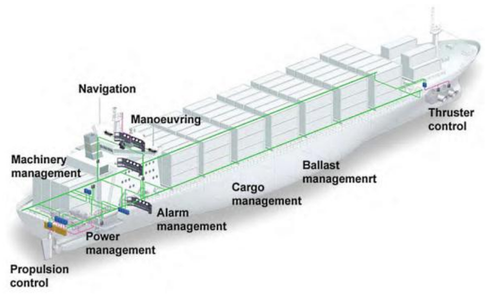

Figure 1: Monitoring and control system.

Service-oriented architecture

Service-Oriented Architecture or SOA for short is a model for system development based on loosely-integrated suite of services that can be used within multiple business domains [2]. SOA is also an approach and practice for building IT software systems using interoperable services. These services are loosely-coupled software components that encapsulate functionalities and are available to be remotely accessed by client applications over a network or Internet [3]. The backbone of SOA consists of web services and an Enterprise Service Bus (ESB).

Web Services

As defined by W3C, a web service is a software component designed to support interoperable machine-to-machine interaction over a network [5]. It uses the SOAP, an XML-based protocol to communicate over HTTP. Characteristically, web services have three key elements: Web Service Description Language (WSDL) which is an XML-based description of the operations and functionalities offered by the web service. It dictates the protocol bindings and the message formats required to connect to and interact with a given web service; Universal Description, Discovery and Integration (UDDI) which is a registry for storing web services' WSDLs and a mechanism to register and locate web services on the Internet; and the SOAP communication protocol which defines the structure and format of the messages being exchanged between the service requester represented by the client and the service provider represented by the actual web service. In fact, the service requester is a client application requesting a particular functionality from the service provider, and the service provider is usually a server that hosts and runs the actual web service. Other types or styles exist for web services. They include REST, RPC, RMI, .NET Remoting, CORBA, and Network Socket [4].

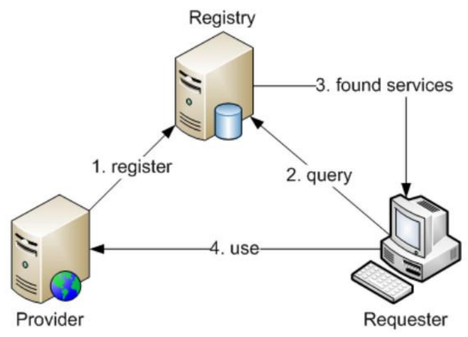

Figure 2: illustrates the infrastructure of a generic web service.

Enterprise Service Bus - ESB

In order to promote interoperability among its components, SOA often employs an Enterprise Service Bus or ESB. Fundamentally, an ESB is a piece of software that lies between the different components of an SOA, mainly between the service requester and the service provider to enable a transparent and seamless communication among them [6]. It, in fact, acts as a middleware and a message broker between the different communicating parties in SOA architecture. The primary task of ESB is to support message routing and ensure a better orchestration and interoperability between the various interconnected web services possibly built using different technologies, platforms, standards, and programming languages. Figure 3 shows an ESB connecting incompatible consumers and producers built using different technologies.

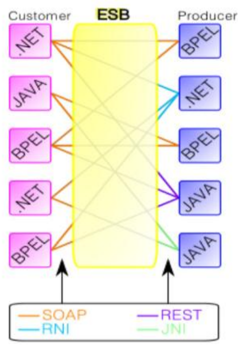

Figure 3: Architecture of an enterprise service bus

Proposed Architecture

This paper proposes a Service-Oriented Architecture (SOA) for building ship automation and control system using service software components. It is a distributed model made out of loosely-coupled interoperable web services and a central Enterprise Service Bus (ESB) not located inside the actual ship equipment but in an isolated location, possibly operation control center or space stations in low earth orbit. The communication between the all of ship control systems and web services is bi-directional and is done in a remote fashion using the HTTP protocol with help of the ESB acting as a middleware. The employed communication style is method invocation in which ship control system equipment can remotely call or invoke the different procedures of the existing web services to execute on the hosting system and return results to the equipment. These procedures also known as methods or functions contain the logic and the programming instructions that deliver the basic functionalities for the ship control system equipment. Essentially, the proposed architecture is composed of three basic tiers:

The first tier is the client represented by the ship system or any control system supporting computation, which invokes the different exposed methods of web services to perform a wide range of operations such as weather forecasting, communication, ballast management, machinery management, alarms and monitoring, tank gauging, cargo monitoring, control and handing, propulsion control, thruster control, auxiliary machinery control, power/energy monitoring and vessel performance optimization.

The second tier is the server represented by web services which are decoupled from the ship equipment and hosted and executed on sever machines located in control centers. The web services provide the actual code and logic for the different operations and functionalities. They contain the algorithm, implementation, and programming instructions necessary to provide the various computing machineries their basic maneuvers and functionalities.

The third tier is the middleware represented by the Enterprise Service Bus which offers a standard interface and a unified data-path for both the client and the server tiers to interoperate efficiently and exchange data regardless of their incompatible platforms and implementation technologies, for instance, technologies such as SOAP, REST, RPC or others. Figure 4 illustrates the proposed SOA architecture and its different tiers.

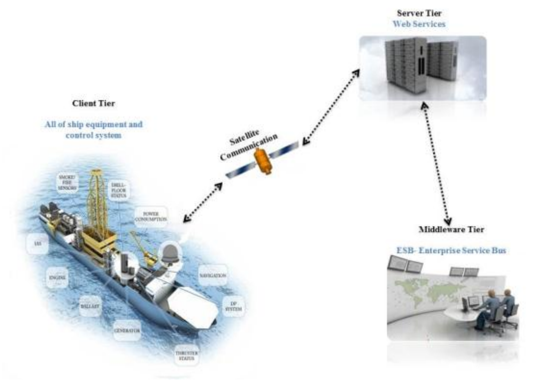

Figure 4: Different tiers of the proposed SOA architecture

Validation of the Proposed Architecture

The SOA approach proved to be very effective in all the different executed scenarios. The interoperability of the system allows the collaboration between various entities regardless of their underlying technologies and implementation details. The scalability of the system allows to easily and quickly alter and add functionalities to any ship system without having access to them. The maintainability of the system allows fixing and replacing out of order services while the system is running with no or minimal operation interruption. The reusability of the system allows building and deriving new web services from existing ones with the least amount of development time, cost and safe.

As a proof of concept, a client simulation idea, representing a ship and all of ship system, was built and is capable of sending requests to and reading results from ESB using the

SOAP protocol. The Client Tier - Actually, the client is any ship system, equipment, device, machine, communication system, infrastructure, and computer use in field. They contain an onboard computer able to discover the different remote web services through the ESB interface which describes the different functions encapsulated within the connected web services; The Middleware Tier - The ESB or Enterprise Service Bus provides a data-path for data to travel between all of ship systems and the web services. It constitutes a data transmission medium, emulating a messaging middleware that links between the different system in the ship and the different distributed web services to allow them to send and receive data to each other; The Server Tier - The server tier is where the web services are hosted. It mainly consists of several mainframe computer servers often located in ship control centers. These servers define the execution of the web services, process any ship's requests from the fleet, execute business logic, and perform intensive calculations on behalf of the system. The web services can be of any type, protocol, or version and they interact with the ESB through its multi-platform end-point adapters. Each time a new web service is integrated into the system, it publishes its WSDL to the ESB which saves it inside an internal registry along with other important details. The ESB then exposes the WSDL to all ship system allowing them to call remotely all available functions. Web services can provide any type of functionalities including GSM to receive and transmit telemetry data between the different ship system using SMS or other communication technologies; navigation to monitor and control the movement of ship in fleet and determine their positions using radars, sensors, and satellites.

Conclusions and Future Work

This paper presented a service-oriented architecture for building ship automation and control system using distributed software components called web services. The proposed architecture consists of three tiers: the client tier corresponding to any sort of computing ship equipment that require executing some functionalities; the server tier corresponding to the web services that deliver the basic functionalities and operations for the ship control system; and the ESB acting as a middleware that coordinates and shields the complexity and heterogeneity of communication among the different entities of the system. Experiments conducted showed a robust, reliable, scalable, interoperable, reusable, and a maintainable architecture that can adapt itself to the unforeseen circumstances and cope with the various obstacles that might be encountered during ship travelling missions. As future work, an encryption layer is to be added to the proposed SOA architecture so as to protect and conceal the exchange of messages and data communication between the various entities of the system.

References

1. http://www.shippipedia.com/ship-automation-control-system/
2. Erl, T, Service-Oriented Architecture: Concepts, Technology, and Design, Prentice Hall, 2005.
3. Josuttis, N., SOA in Practice, O'Reilly, 2007.
4. Fielding, R., Taylor, R., Principled Design of the Modern Web Architecture, ACM Transactions on Internet Technology (TOIT), Vol. 2, No. 2, pp. 115–150, 2002.
5. Web Services Glossary, W3C, 2004, [online]http://www.w3.org/TR/2004/NOTE-ws-gloss-20040211
6. David Chappell, Enterprise Service Bus, O'Reilly, 2004.
7. KONGSBERG Information Management System.
8. Innovations in Integrated Control Systems, Halvard Foss, Kongsberg Maritime AS – Kongsberg, Norway. October 17-18, 2006.
9. David Booth, Hugo Haas, and Francis McCabe, *Web Services Architecture*, W3C Working Group, 2004, [online] http://www.w3.org/TR/ws-arch/