# Rapid Automation Application Deployment Framework for Real Time Process and Industrial Automation Systems

S. Veera Ragavan, Velappa Ganapathy, and Ibrahim Kusumah Kusnanto

***Abstract*—Development of a Low Cost Industrial Automation and Control System Framework based on Service-Oriented Architecture (SOA) using OSGi Framework for rapid application deployment has been attempted here. Objective is to develop a user-friendly, Platform and Hardware Independent Industrial Automation System using Free and Open Source Software (FOSS) and Completely-off –the-shelf (COTS) Hardware. The developed application prototype successfully integrates a service enabled SCADA system and its real time data acquisition and control of various industrial applications using Extensible Messaging and Presence Protocol (XMPP). The prototype also provides information through a user friendly Graphical User Interface (GUI) for process visualization and remote controllability. An Alarm Event Handling Service with integrated Short Message Service (SMS) and Remote Service Invocation abilities are the salient features of the developed system. Implementation using popular industry standard SCADA software (IGSS) proves that the work is scalable as required by real-world applications.**

***Index Terms*—Service-Oriented Architecture (SOA), open systems, software architecture, OSGi framework, distributed services, software systems, SCADA, industrial automation.**

## I. Introduction

The basic motivation for developing an Industrial Automation System is productivity enhancement through real time data acquisition, remote monitoring, remote maintenance, highly flexible service upgrades, and a low environmental impact. Since users expect to get improved Services at reasonable cost, there is a need for an integrated and low-cost system platform that allows intelligent features to be easily implemented [1].

Service-Oriented Architecture (SOA) provides for the development of a flexible and modular system. The main idea of SOA is to create "Services" that can be constructed together to build a system. Services are functions that are well-defined, self-contained, and independent of the context or state of other services. Services communicate with each other, and this involves data passing and activity coordination [2], [3]. Besides the key traits in SOA are Dynamism and Substitutability. Dynamism is the ability for Service Providers to offer and retract Services at any time and the ability of service user to bind available services at will. Substitutability is the fact that a service description represents a contract [4].

Manuscript received December 15, 2013; revised March 25, 2014.

The authors are with School of Engineering, Monash University, Bandar Sunway, Malaysia (e-mail: veera.ragavan@.monash.edu, ibkusnanto@gmail.com).

OSGi is a lightweight SOA based application deployment framework [4] using Java programming language. The OSGi [5] specifications benefits from key traits in SOA where services and their associates can be added or removed at any time, service providers may offer or retract services tailored to the networked environment's connected devices and also the ability of service providers to offer services intended to be open and inclusive to promote interoperability [4]. OSGi applications are Platform and Hardware Independent as they run on Java. A middleware layer on top of OSGi called Remoting-OSGi (R-OSGi) [6] provides the ability of services to be used remotely.

Using OSGi framework, this work aims to develop and implement a SOA based Industrial Automation System that is Service based, Scalable, Dynamic, Open Source and a Low cost a real world application.

## II. Methodology

A SOA based Automation System implementation using OSGi framework, on Java using XMPP [7] has been attempted. The OSGi framework employs the service-oriented approach and the java class-loader architecture for the runtime service deployment that are well suited for dynamic environments [8]. Bundles are regular Java JAR files containing class files, other resources (images, icons, required APIs etc), and also a manifest, which is used to declare static information about the bundle, such as the packages the bundle import and export. Further, bundles may also provide services to other bundles. In the OSGi architecture, a service is a standard Java object that is registered using one or more interface types and properties (that are used to locate the service) [9]. Two types of Bundles will be developed: Service Bundles and Application Bundles.

Service Bundle provides the services for the Application Bundles to use. Service Bundles to be created must provide the capability to communicate with IGSS [10] using XMPP for real-time data acquisition, alarm event facilitation and handling. These Service Bundles needs to be loosely coupled to minimize dependency between the services coupling bundles independently implemented by different developers (reusability and flexibility) [9]. Service Bundles interact with the Application Bundles through Service Interfaces (SI). The design of SI's is unique and specific to the intended Service that the Service Bundle offers. The interfaces are collections of methods (services) which are offered by the Service Bundles to be used by Service Consumers (Application Bundle).

The main application in the Prototype is the Application Bundle which is the bundle that uses the Services provided

by Service Bundles to assemble fully functional Automation System with the functionality aimed in the objectives. The application has the ability to control IGSS Stations automatically or manually and trigger alarm handling and SMS notification for alarm event.

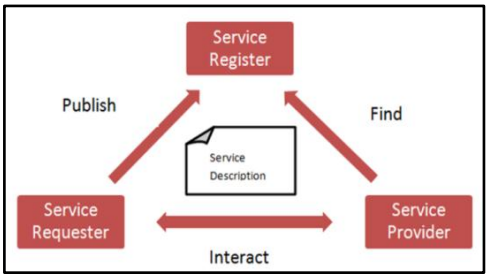


Fig. 1. A service oriented interaction pattern [4].

During runtime Service Bundles need to be started before Application Bundle, as Application Bundle consumes or depends on the services in Service Bundles [11]. Service Bundles offer services to Application Bundles, where service functionality is characterized in a Service Description. Service Bundles are Discovered using their Service Descriptions by Application Bundle querying a Service Registry, where Service Bundles Publish Service descriptions [4]. This service registry will be available in the Main Computer that has the Service Bundles and Application Bundle itself. A Remote Application Bundle that has the same functionality with the Application Bundle will be implemented on a Remote Computer to use the Service Bundles on the Main Computer.

As OSGi applications run on Java it makes the application to be Platform and Hardware Independent, as Java can run on any Operating System and machines [12].

All bundles development is done using Eclipse Rich Client Platform (RCP) [13] which is a multi-language software development environment for developing general purpose applications. Eclipse RCP is a pure-plug in platform and, hence, fully extensible by architectural design since it is based on Equinox, the Eclipse implementation of the OSGi specification. An adoption of Eclipse Communication Framework (ECF) in Eclipse RCP supports the development of distributed Eclipse-based tools and applications, which provides XMPP and R-OSGi [11].

XMPP connection from the IGSS Stations was done using the integrated Visual Basic interface on the IGSS software.

## III. Literature Review and Related Work

Similar deployments of application frameworks for Industrial Automation such as SIMOO-RT, CORFU, and Real-time Framework have been done. However they are not SOA based.

SIMOO-RT is an Object Oriented framework designed to support the whole development cycle of real-time industrial automation systems. It is based on the concept of distributed active objects, which are autonomous execution entities that have their own thread of control, and interact with each other by means of remote methods invocation [14].

CORFU is an Object Oriented framework to improve the engineering process of Industrial Process Measurement and Control Systems (IPMCSs) in terms of reliability, development time and degree of automation and embodies an abstract design that is capable of providing solutions for a family of distributed IPMCSs [15].

Real-time Framework is a package of software modules for building distributed real-time control systems in Robotics and Automation. The Real-time Framework covers the areas of client-server communication, control of program flow and modality through messaging and state machines, and low-level input/output. In addition, it contains a real-time utilities package and wrappers for operating system calls, which shield any operating system dependence from the application built on top of the Framework [16].

## IV. Architectural Overview

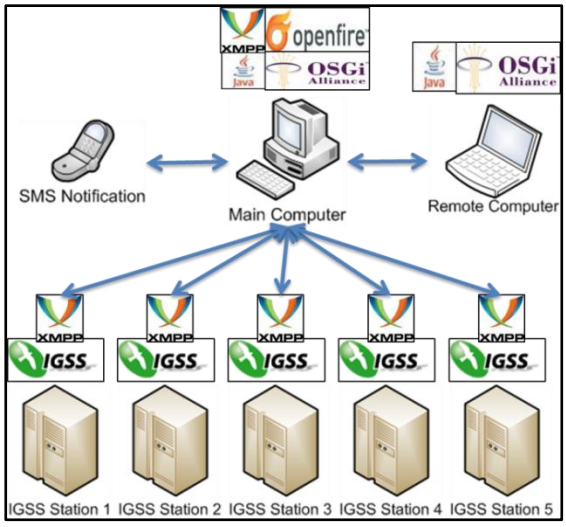


Fig. 2. Hardware architecture.

The Prototype system developed consists of a Main Computer, a Remote Computer and five IGSS Stations. These IGSS Stations are simulations of different types of typical Industrial Environments which provide data to the Application Bundle to the Main or Remote Computer using XMPP.

A Main Computer installed with OSGi Framework provides the Service Bundles. It is also has Application Bundles. The Open fire installed at the Main Computer functions as the XMPP Server that provides data transaction between the IGSS Stations and Main computer.SMS notification sent utilizing GSM modem at the Main Computer. Main computer runs a Windows OS.

Remote Computer installed with OSGi Framework can access the Service Bundles at The Main Computer through R-OSGi. Remote Computer Runs using Ubuntu OS.

Equinox OSGi, which is an implementation of the OSGi framework [17] has been used. All the Service Bundles developed will be available in the Main Computer. Service Bundles register their Services during runtime at the Service Registry on the Main Computer itself. Application Bundle on the Main Computer uses the Service Bundles from the Service Registry to build a SCADA system.

A Remote Application Bundle on the Remote Computer access the Service Registry on the Main Computer using R-OSGi, as the results the Remote Application Bundle is able

to use the Service Bundles on the main computer as if they were local services [11]. The Remote Application Bundles have the same functionality as the Application Bundle in the Main Computer.

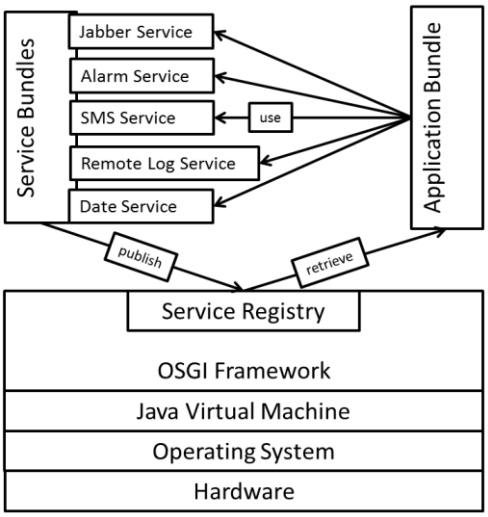


Fig. 3. Software architecture.

## V. Service and Application Bundles

### A. Service Bundles Deployed

#### 1) Jabber service

Jabber Service provides the communication for real-time data acquisition and control of the IGSS Stations. This real-time communication between the Application and IGSS is done using XMPP [7]. XMPP is an open technology for real-time communication between an XMPP Client and an XMPP Server. The XMPP Clients in this context are the Application and the IGSS Stations. These XMPP Clients exchange data by sending Extensible Mark-up Language (XML) to the XMPP Server and will be received by the XMPP Client destined [7]. Jabber Service also provides the information of availability of XMPP Clients. XMPP can be used to exchange any data that can be represented in XML, enabling development of a wide variety of applications. XMPP is much more than Instant Messaging. The semantics of the "push" mechanism," publish-subscribe" mechanism and request-response mechanism stanzas provide a generalized communication layer, making it possible to develop and deploy a wide range of presence-enabled applications [18].

#### 2) Alarm service

Alarm Service provides alarm event handling. When an alarm occurs Alarm Service provides information about the cause of the alarm and the current alarm state, which helps the operator to handle and solve an alarm event, also to log the alarm event occurred. The application uses the SMS Service for SMS notification and acknowledgement of alarm events

#### 3) Communication service -SMS service

SMS Service provides sending and receiving SMS to the application. The SMS Service in the work utilizes Falcom SAMBA75[19] Global System for Mobile Communications (GSM) modem. Communication Services such as Blue tooth, Field Busses, LAN etc can also be easily implemented.

#### 4) Remote log service

Remote Log Service provides a simple logging when the Service Bundles are used remotely by recording the name of Remote Computer using the Remote Services on the Eclipse console. It can be extended easily for more detailed logging.

#### 5) Date service

Date Service provides the current date and time vital for synchronization and messaging

### B. Application Bundle Deployed

The Application Bundle aggregate the Services to construct SCADA system. Using Jabber Service it will receive data in XML format a popular standard for data representation and exchange [20]. This XML data is then parsed to display the reading from the IGSS Stations and to send the processed data back to the IGSS Station. This data exchange allows the Application Bundle to acquire remote data and to Remotely Control the IGSS stations automatically or manually in real time. Using Jabber Service it can detect the availability of the IGSS Stations. Alarm Service uses alarm handling when the processed data triggers an alarm event. Alarm event triggers a SMS notification to be sent using SMS Service. Date, time and Remote Service User are also present using the Remote Log Service and Date Service. The Prototype has a GUI that provides graphical representations of the data from each IGSS Station, manual control interface, and alarm event log.

### C. Remote Application Bundle

Remote Application Bundle on the Remote Computer uses the Service Bundles on the Main Computer to construct Remote SCADA system that has the same functionality as the SCADA system at the Main Computer.

## VI. Service and Process Flow

A typical Industrial Process and Services Interaction pattern is as shown in Fig. 4 below.

### A. Initialization

During the initialization stage, when the system starts it will register all the Service Bundles on to the Service Registry, then the Application Bundle will start and check whether all the Service Bundles required for the Application Bundle available. If all the Service Bundles are available, then the Application Bundle will run until a stopped condition met. If some or all the Service Bundles required by the Application Bundle are not available, the system will return an error and is stopped.

### B. Application Run

When the Prototype runs it will trigger the GUI which consists of several tabs representing each IGSS Stations, IGSS Stations availability, and alarm event log. After triggering the GUI, the Prototype will then connect to the XMPP Server and once connected it will check the availability of the IGSS Stations. If one or more IGSS Stations are available it will start checking for new data feed from the IGSS Stations which will be processed and displayed as tags on the Application as well as to give

feedback data to the IGSS Stations.

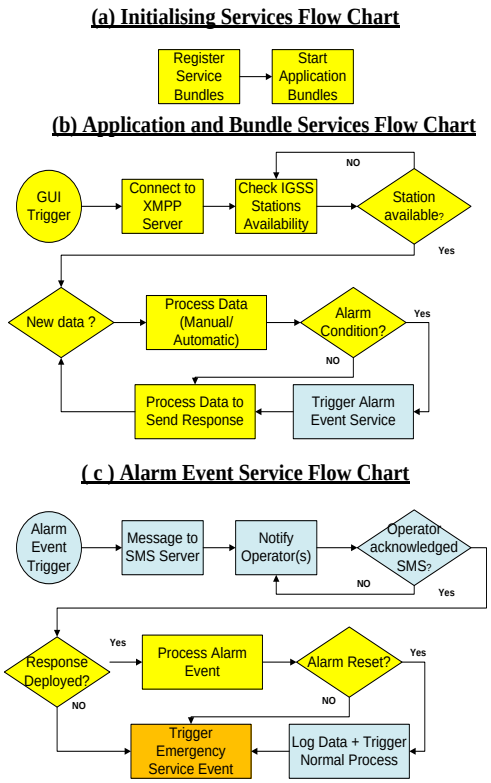

Fig. 4. Services and process flow.

With this data feedback, the control of some instruments on the IGSS Stations, are transferred for remote control which can be executed automatically or manually. The data process checks the data for, and based on preset criteria trigger alarm event, send notifications and transfers control before sending the processed / feedback data to the IGSS Stations.

### C. Alarm Event

When an alarm event occurs, an SMS notification will be sent to the operator of the IGSS Station. The operator acknowledges by sending an SMS. If the operator fails to acknowledge, another SMS notification will be sent to the operator and Supervisor until the operator acknowledges or triggers an Emergency. After acknowledging the SMS notification, the operator proceeds to acknowledge the alarm in the Plant itself showing that the operator already On Call to solve the problem. Again if the operator fails to acknowledge the alarm in the Plant, within a given time period, an SMS Notification will be sent again and the operator and the supervisor to acknowledges it. After the operator acknowledges the alarm on the Prototype, it will wait until the operator solves the problem and reset the alarm so that the process runs normally.

## VII. Results

### A. Real-Time Data Acquisition

Fig. 5 – Fig. 9 show the data read by the Application and its corresponding IGSS Stations. The Application can monitor and control multiple IGSS Stations on a single PC or Laptop, thus The Application provides the ease to monitor and control large scale Industrial Applications.

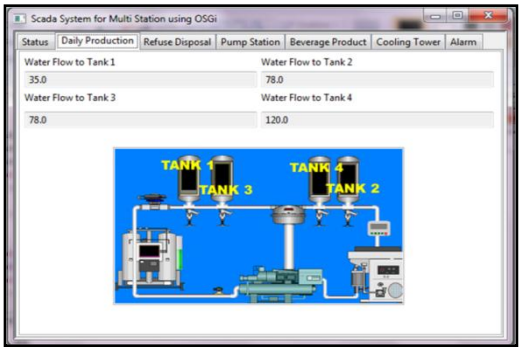

Fig. 5. Station 1 data control.

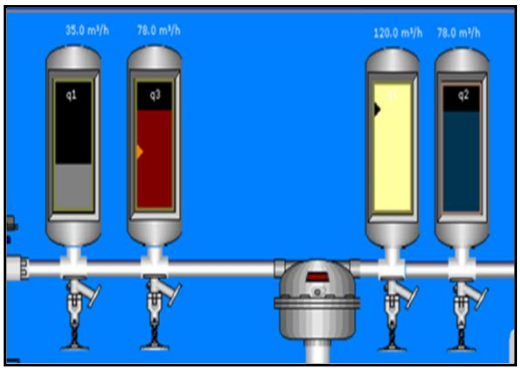

Fig. 6. IGSS station 1.

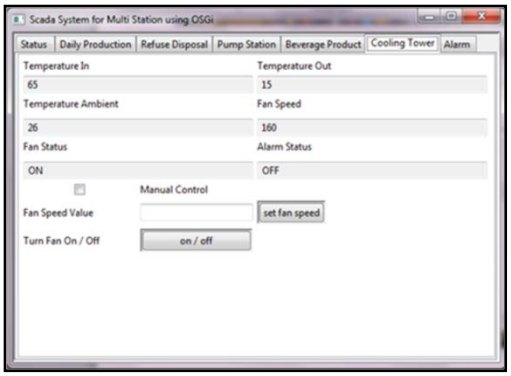

Fig. 7. Station 5 data control.

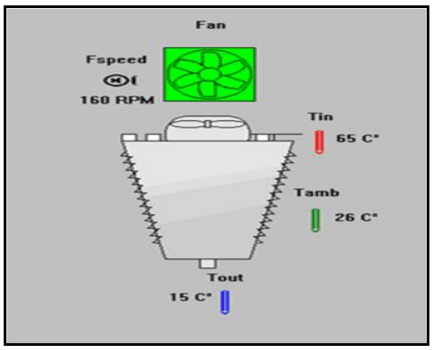

Fig. 8. IGSS station 5.

All the data read from the IGSS Stations are done in real-time by the Application, which benefits in alarm events or any other abnormal situations.

The Application provides a GUI for visualizing the process, data and status and availability of IGSS Stations using simple graphical representations as shown in Fig. 8.

Other than monitoring the IGSS Stations, the Application is capable of controlling IGSS Stations as well. An example of controlling of the IGSS Station shown in IGSS Station 5, where the Fan Speed of the Station is controlled automatically by the processed data from the Application.

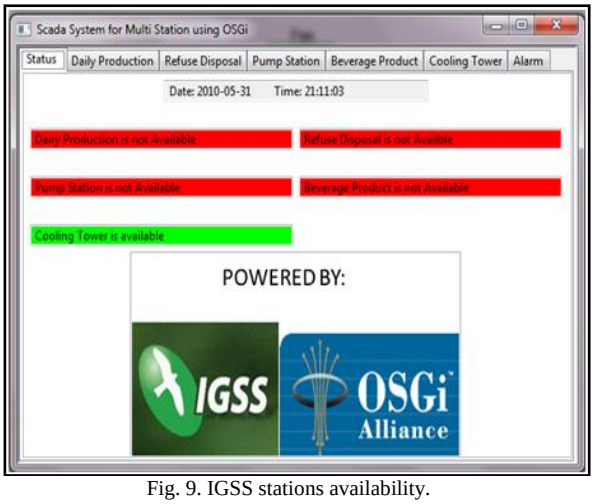


Fig. 9. IGSS stations availability.

### B. Alarm Event

The Application provides alarm events handling by notifying operator of the corresponding IGSS Station in abnormal situation using SMS. All the alarm events states are logged in the Alarm tab.

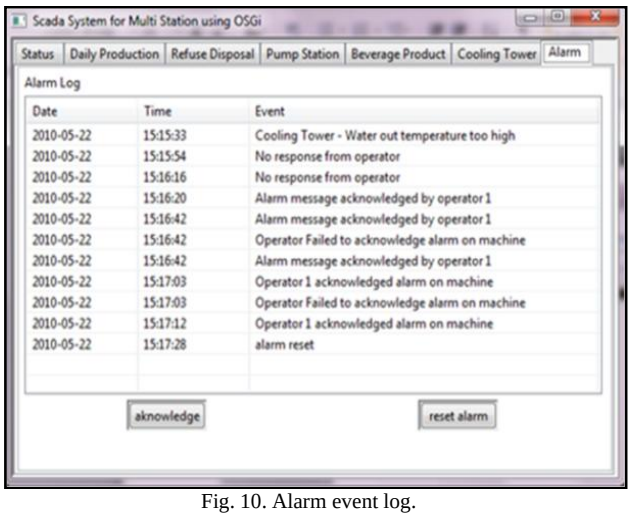


Fig. 10. Alarm event log.

Figure above shows a scenario of an alarm event. The alarm event was triggered because the Water-Out Temperature from the cooling tower (IGSS Station 5) was above the high limit Set. An SMS notification was then sent to the operator of the IGSS Station. The operator acknowledges the SMS Notification as well as the alarm in the Application console (Plant) by pressing the "Acknowledge" Button on the Alarm Tab. The informative GUI helps the operator find out that the problem caused by the Fan turned OFF. Using manual control the operator turned the Fan on then resets the alarm.

### C. Manual Control

Application provides a manual control for IGSS Station 5. Parameters that can be controlled are Fan Speed and turning Fan ON or OFF. As shown from the figures, the remote application controls the Fan Speed through safety interlocks provided.

Manual controls of the IGSS Stations are limited only controlling the parameters of the IGSS Stations, as full remote controlling of IGSS Stations such as turning ON or OFF the IGSS Stations remotely can be dangerous.

### D. Remote Application

Using same GUI's the Remote Application replicates and has the same extended functionally as that of the Application on the Main Computer. In this work the Remote Application was implemented on a Local Area Network. It can also be quickly extended to achieve Remote Application control over the internet.

## VIII. Evaluation

### A. Extensibility

Service Registry in the OSGi Framework provides the ability to add new Services without stopping and starting the Application.

### B. Modular

Bundles made are independent of each other. Thus it is possible to build different applications using some of the Service Bundles or all of it.

### C. Dynamism

Service Registry in the OSGi Framework provides Services Bundles on offer and can retract Services at any time. Application Bundle is able to bind to any available service at will.

### D. Substitutability

Service Bundles that are offered are loosely-coupled, which gives the suitability for any Service Bundle to be imported from another bundle.

### E. Platform/OS Independent

As OSGi Framework runs on top of JVM, the Application can be said to be OS and Platform independent.

## IX. Conclusion and Future Work

We accomplished Industrial Automation System implementation using OSGi framework. The Prototype was built using Service Bundles that were made to be flexible and modular.

Prototype system provides data acquisition and control which are done in real-time and it provides near real time alarm event handling. GUI's that facilitate the ease of use was implemented.

As OSGi Applications run on Java, the Application Prototype is Platform and Hardware Independent. This will save the cost of obtaining Legacy Industrial Controllers for an Industrial Automation System.

Prototype is scalable as it can be integrated with IGSS, which is an industry standard SCADA software used world-wide.

Future work that has been planned are:

- Further Testing of the framework for reliability and reusability are needed. Testing of SOA itself is a non-trivial and complex task [21].
- Implementation of a Security Layer for Bundles access [22] to prevent hacking.
- Implementation of the prototype framework in a soft real-time Industrial Automation scenario with other

Industrial systems, services and devices (such as IP cameras), etc.

## REFERENCES


[1] T. Tommila, O. Ventä, and K. Koskinen, "Next generation industrial automation–needs and opportunities," *Automation Technology Review*, pp. 34-41, 2001.

[2] D. K. Barry. (2010). Service-Oriented Architecture (SOA) definition. [Online]. Available: http://www.service-architecture.com/web-services/articles/service-oriented_architecture_soa_definition.html

[3] D. K. Barry, *Web Services, Service-oriented Architectures, and Cloud Computing: The Savvy Manager's Guid*, Newnes, 2012.

[4] N. Series, "Challenges in building service-oriented applications for OSGi," *IEEE Communications Magazine*, p. 145, 2004.

[5] O. Alliance, *OSGi-the Dynamic Module System for Java*, vol. 25, *May* 2009.

[6] J. S. Rellermeyer, G. Alonso, and T. Roscoe, "R-OSGi: distributed applications through software modularization," in *Proc. the ACM/IFIP/USENIX 2007 International Conference on Middleware*, 2007, pp. 1-20.

[7] P. Saint-Andre, "Extensible messaging and presence protocol (xmpp): Instant messaging and presence," 2011.

[8] H. Ahn, H. Oh, and C. O. Sung, "Towards reliable osgi framework and applications," in *Proc. the 2006 ACM symposium on Applied computing*, 2006, pp. 1456-1461.

[9] A. L. Tavares and M. T. Valente, "A gentle introduction to OSGi," *ACM SIGSOFT Software Engineering Notes*, vol. 33, p. 8, 2008.

[10] What is IGSS? (2010). [Online]. Available: http://igss.schneider-electric.com/products/igss/product-information/what-is-igss.aspx

[11] J. S. Rellermeyer, G. Alonso, and T. Roscoe, "Building, deploying, and monitoring distributed applications with Eclipse and R-OSGI," in *Proc. the 2007 OOPSLA Workshop on Eclipse Technology Exchange*, 2007, pp. 50-54.

[12] R. Helaihel and K. Olukotun, "Java as a specification language for hardware-software systems," in *Proc. the 1997 IEEE/ACM International Conference on Computer-Aided Design*, 1997, pp. 690-697.

[13] D. Gruber, B. Hargrave, J. McAffer, P. Rapicault, and T. Watson, "The Eclipse 3.0 platform: adopting OSGi technology," *IBM Systems Journal*, vol. 44, pp. 289-299, 2005.

[14] L. B. Becker and C. E. Pereira, "SIMOO-RT-an object-oriented framework for the development of real-time industrial automation systems," *IEEE Transactions on, Robotics and Automation*, vol. 18, pp. 421-430, 2002.

[15] K. Thramboulidis, "Development of distributed industrial control applications: the CORFU framework," in *Proc. 4th IEEE International Workshop on Factory Communication Systems*, 2002, pp. 39-46.

[16] A. Traub and R. D. Schraft, "An object-oriented realtime framework for distributed control systems," in *Proc. 1999 IEEE International Conference on Robotics and Automation*, 1999, pp. 3115-3121.

[17] J. McAffer, P. V. Lei, and S. Archer, *OSGi and Equinox: Creating Highly Modular Java Systems*, Addison-Wesley Professional, 2010.

[18] P. Saint-Andre, "Streaming xml with jabber/xmpp," *Internet Computing, IEEE*, vol. 9, pp. 82-89, 2005.

[19] Falcom, "SAMBA75 - Integrated Quad Band GSM/GPRS/EDGE Engine.," 2010.

[20] E. Bertino and E. Ferrari, "XML and data integration," *Internet Computing, IEEE*, vol. 5, pp. 75-76, 2001.

[21] G. A. Lewis, E. Morris, S. Simanta, and L. Wrage, "Common misconceptions about service-oriented architecture," in *Proc. Sixth International IEEE Conference on Commercial-off-the-Shelf (COTS)-Based Software Systems*, 2007, pp. 123-130.

[22] P. Parrend and S. Frénot, "Supporting the secure deployment of osgi bundles," in *Proc. IEEE International Symposium on World of Wireless, Mobile and Multimedia Networks, WoWMoM*, 2007, pp. 1-6.


**Sampath Kumar Veera Ragavan** is currently a senior lecturer of mechatronics engineering at Monash University Malaysia Campus. Before joining Monash, he worked for several multinational companies in various capacities. He has more than 17 years of industrial experience in design and development of factory automation systems, fluid power automation, software for consumer electronics and industrial automation He has executed several projects from concept to commissioning. He is also a chartered engineer from Engineering council (UK) and a consultant in the field of mechatronics, telematics and industrial automation. His current research interests are physical modeling and simulation, design of mechatronic systems, robotics, industrial automation and distributed embedded systems.

**Ibrahim Kusumah Kusnanto** was born in Bali, Indonesia in 1988. He received his bachelor degree (Hons.) in the field of mechatronics engineering at Monash University Sunway Campus, Malaysia. He is currently a senior developer at Dropmysite Pte Ltd. Singapore.