

Virtualization for Manufacturing Web Services: a WS-RF approach

Rajaram Shivram Kumar¹, Zhonghua Yang¹, Jing Bing Zhang², Liqun Zhuang²

¹Information Communication Institute of Singapore (ICIS)
School of Electrical and Electronics Engineering
Nanyang Technological University, Singapore 639798

²Singapore Institute of Manufacturing Technology
Singapore 638075

ezhyang@ntu.edu.sg

Abstract

Enterprise integration is a top priority for many organizations attempting to streamline business processes, coordinate disparate functions, improve corporate responsiveness to customers and partners and optimize supply chains in an effort to improve productivity. Web services orientation has appeared as a new paradigm for realizing enterprise integration, and fundamental to the service orientation is the virtualization. In the service orientation approach, every thing in a considered environment (devices, equipments, processes and applications) is required to be virtualized as a Web service. Via a virtualization, all manufacturing systems are exposed as Web services, and they can offer their services and functionality in a standard Web service environment. In this paper, we adopt Web Services Resource Framework (WSRF) as a major mechanism for virtualization and for illustration purpose, we use WSRF to virtualize an AGV. Different views of an AGV are defined via virtualization. WSRF.Net is then used to develop the prototype wherein the different views of the AGV are depicted as WS Resources. Different parameters of a view are accessed and operations are performed on the AGV through the web service.

Keyword: Service-orientation, Virtualization, Enterprise integration, Web Services Resource Framework (WS-RF)

I. Introduction

Enterprise integration is a top priority for many organizations attempting to streamline business processes, coordinate disparate functions, improve corporate responsiveness to customers and partners and optimize supply chains in an effort to improve productivity. The need to coordinate manufacturing resources that are not subject to centralized control, to use the standard and open protocols and to deliver nontrivial qualities of services have transformed the traditional network manufacturing architecture into the novel manufacturing Grid architecture. The service-oriented paradigm and Web services technologies are rapidly emerging as the most practical approach for integrating a wide array of manufacturing resources in the manufacturing Grid environment, and

efforts in Semantic Web standards and technologies open an attractive opportunity for automating the integration process. Web services orientation has appeared as a new paradigm for realizing enterprise integration.

The fundamental for realizing service-oriented enterprise integration is virtualization. To achieve service-orientation, every thing in a considered environment (devices, equipments, processes and applications) is required to be virtualized as a Web service. In other words, via virtualization, all manufacturing systems are exposed as Web services, and they can offer their services and functionality in a standard Web service environment. In this paper, we present the Web standard based technologies and approaches for virtualization; the focus is placed on the emerging WS Resource Framework (WS-RF) [6, 2]. Using WS-RF, manufacturing resources will be modeled as a WS-Resource (combination of the Web Service and the stateful resource) [5] (Figure 1). This paper proposes the preliminary mechanisms by which a view of the state of the resource is defined and associated with a Web services description. The resource is then allowed for access in a standard Web service environment. As a proof of concept, virtualization is done for an AGV (Automatic Guided Vehicle). Different views of an AGV are defined and each view is depicted as a WS Resource. Different parameters of a view are accessed and operations are performed on the AGV through the web service.

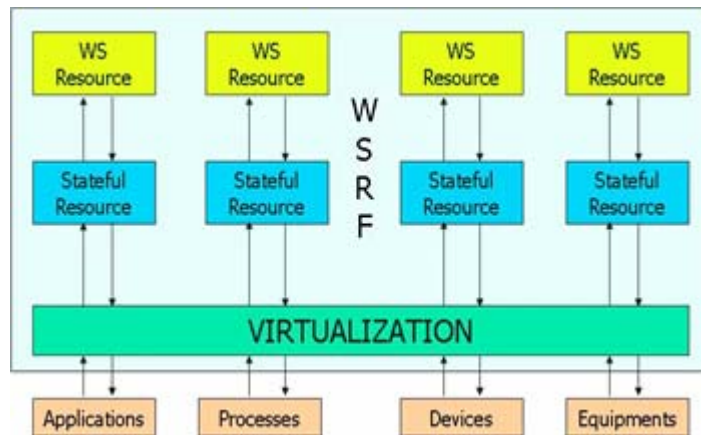


Figure 1: Virtualization approach via WS-RF

This paper is organized as follows. We first give a brief overview of WS-Resource Framework in the next section. The achieving virtualization using WS-RF is presented in section 3. The design and prototyping of an AGV virtualization is outlined in section 4. Finally section 5 concludes the paper.

II. WS-Resource Framework: A brief overview

Web Services have been here since 2000 along with the usage of XML-based technologies like SOAP, WSDL (Web Services Definition Language), UDDI (Universal, Description, Discovery and Integration). The W3C web services architecture working group defines web service as “a software system designed to support interoperable machine-to-machine interaction over a network, and it has an interface described in a machine-processable format (specially WSDL), and other systems interact with the web service in a manner prescribed by its description using SOAP-messages, typically conveyed using HTTP with an XML serialization in conjunction with other web-related standards” [11]. The simplicity derived from the XML technology results in the widespread adoption of Web Services in industry.

Although web services have been successful, they are mainly regarded as “stateless”, i.e. they are unable to remember the results of one invocation to the other. This limits the usage of web services from using the results of any previous invocation. The notion of “stateful web services” was first introduced in the Grid community [5]. It was argued that it is convenient and useful to extend the stateless Web services to stateful Web services. The concept of “Grid Services” as introduced is the one of stateful Web services. Grid services are defined by the OGSA [4], and stateful Web services were specified by the “Open Grid Services Infrastructure” (OGSI) [9]. The OGSI defines a set of conventions and extensions on the use of Web Service Definition Language (WSDL) and XML Schema to enable stateful Web services. It defines approaches for creating, naming, and managing the lifetime of instances of services; for declaring and inspecting service state data; for asynchronous notification of service state change; for representing and managing collections of service instances; and for common handling of service invocation faults [1]. The Grid service extends the Web service by including a service factory to create service instances, so that states can be encapsulated inside the instance.

Although OGSI was developed for the convergence of Grid and Web Services, it was quickly realized that OGSI was not readily accepted by the Web services community for the following reasons [1]:

- *Too much stuff in one specification:* OGSI did not have a clean separation (factoring) of functions to support incremental adoption. The large size and scope of the OGSI specification has made it hard for readers to understand its contents and to identify and refer to those components that are required for a specific task.
- *Does not work well with existing Web services and XML tooling:* OGSI v1.0 uses XML Schema aggressively, for example with substantial use of `xsd:any`, attributes and “document-oriented” WSDL operations. Unfortunately, this use of these standard XML Schema features has caused problems with some existing Web services toolkits, XML development tools, and standards (e.g., JAX-RPC).
- *Too object oriented:* OGSI v1.0 models a stateful resource as a Web service that encapsulates the resource’s state, with the identity and lifecycle of the service and resource state coupled. This approach was not accepted by the web services community as some web services purists argued that web services did not have state or instances. In addition, some Web services implementations do accommodate dynamic service creation and destruction.

To overcome all these limitations of OGSI, in January 2004, the WS-Resource Framework was proposed as a refactoring and evolution of OGSI aimed at exploiting new Web services standards, specifically WS-Addressing, and at evolving OGSI based on early implementation and application experiences. The WS-Resource Framework retains essentially all of the functional capabilities present in OGSI, while changing some of the syntax (e.g., to exploit WS-Addressing) and also adopting a different terminology in its presentation. WSRF is *a significant step by the Grid community to align with Web Services*. It brings about the convergence of the Grid and Web Services.

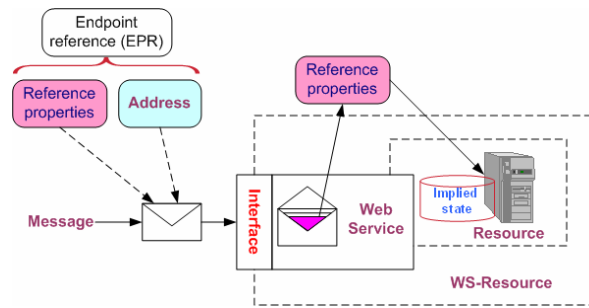


Figure 2: WS-Resource, its Endpoint and implied resource pattern

WSRF separates the state management from the Web service itself (Figure 2). It is a set of Web services specifications that define a rendering of the WS-Resource approach in terms of specific message exchanges and related XML definitions. A WS-Resource [5] is defined to have a specific set of state data expressible as an XML document, have a well-defined lifecycle, and be known to, and acted upon, by one or more Web services. The concerns expressed in OGSF were addressed as follows [1]:

- Addressing *Too much stuff in one specification*: The WS-Resource Framework and WS-Notification specifications split the OGSF v1.0 functionality into a family of separate specifications that allow for flexible composition. In fact, there are total six specifications that constitute the Web services resource framework (WS-Resource, WSResourceProperties (WSRF-RP), WS-ResourceLifetime (WSRF-RL), WS-ServiceGroup (WSRF-SG), WS-BaseFaults (WSRF-BF)), all of which except for WS-Addressing are under standardization by OASIS [12].
- Addressing *Does not work well with existing Web services and XML tooling*: The WS-Resource Framework tones down the usage of XML Schema and uses standard XML Schema mechanisms that are familiar to developers and are supported by existing tooling.
- Addressing *Too object oriented*: The WS-Resource Framework makes an explicit distinction between the "service" and the stateful entities acted upon by that service. The WS-Resource Framework defines the means by which a Web service and a stateful resource are composed. The WS-Resource Framework calls these compositions "WS-Resources," and introduces the "implied resource pattern" [5] to form the relationship between Web services and the stateful resources through a conventional use of WS-Addressing.

The WSRF has certain advantages over OGSF. It better exploits existing XML standards, as well as emerging Web services standards such as WS-Addressing. Thus, the WS-Resource Framework is easier to implement within existing and emerging Web services toolkits, and easier to exploit within the scope of Web services interfaces in definition [1].

To create a new stateful service, a WS-Resource factory is required. A WS-Resource factory is a web service to create instance of a stateful resource. This factory can be the stateless Web service itself, as long as it is capable of bringing a WS-Resource into existence. When a requestor makes a service request, an endpoint reference [10, 2] is returned to the requestor. The service requestor would use the endpoint reference to send messages to the identified Web service. The Web service then uses the identifier in the endpoint reference to access/modify the stateful resource (ref to Figure 2).

III. Virtualization using WS-Resource Framework

In order to investigate the applicability of WS-RF to service-orientation, we consider a physical AGV (Automatic Guided Vehicle) system. An Automated Guided Vehicle system is a material handling system in which driverless, battery-powered carts are moved by means of Laser, Optical or Electronic guidance. The vehicles themselves may be towing vehicles, fork lifts, pallet trucks, or unit load carriers. Control of AGVs in most manufacturing systems is accomplished by means of a central computer. The central computer often contains system wide information, and directs the vehicle where to go to pick up a load, where to take the load, and might even redirect the vehicle while it is in transit to perform another task. Depending upon the type of system, communication may occur only at specified locations, or in more advanced systems, at any vehicle position.

Virtualization of an AGV is done to achieve automatic integration of manufacturing and service systems. By depicting the AGV as a Web service, we can obtain different information of a material handling system. The exact positioning of an AGV can be obtained and we can know the different operational parameters of an AGV like speed, curve radius, battery voltage etc. Information regarding the physical specifications of an AGV can also be obtained as well as different performance parameters like Availability, resource utilization can be obtained. Finance people can get information regarding to the different costs associated with an AGV like production costs, operational costs etc as well as different types of savings done in the plant due to the provisioning of an AGVS.

When the AGV is virtualized as Web services, they offer different views of the AGV. The view represents an abstraction of the physical system. In the WS-RF, different views of an AGV are defined in WS-ResourceProperty [7]. In WS-RF, the resource properties document type associated with a Web service's WSDL 1.1 portType definition provides the declaration of the exposed resource properties of the WS-Resource [7]. It represents a particular composed structural view or projection of the resource properties of the WS-Resource, essentially exposing the stateful resource component within the WS-Resource composition.

For the case of an AGV, different views can be classified into the following areas [8, 3]:

- Physical
- Operational
- Performance
- Financial

We now present how these different views are abstracted and defined in WS-RF. For the physical view of the AGV, the Resource properties document is defined below.

```
<wsdl:definitions ...xmlns:tns="http://localhost/agvviews" ...>
.....
  <wsdl:types>
    <xsd:schema targetNamespace="http://localhost/agvviews" ... >

      <!-- Resource properties document declaration -->
        <xsd:element name="PhysicalAGVViewProperties">
          <xsd:complexType>
```

```

        <xsd:sequence>
        <xsd:element ref="tns:AGVType"/>
        <xsd:element ref="tns:GuidanceType" />
        <xsd:element ref="tns:ModeOfLoadTransfer" />
        <xsd:element ref="tns:LoadType"/>
        <xsd:element ref="tns:LoadLength" />
        <xsd:element ref="tns:LoadWidth" />
        <xsd:element ref="tns:LoadHeight"/>
        <xsd:element ref="tns:LoadDiameter" />
        <xsd:element ref="tns:MaximumLoad" />
        <xsd:element ref="tns:MinimumLoad"/>
        <xsd:element ref="tns:LoadTransferHeight" />
        <xsd:element ref="tns:NumberOfPickUpLocations" />
        <xsd:element ref="tns:NumberOfDeliveryLocations"/>
        <xsd:element ref="tns:BatteryType" />
        <xsd:element ref="tns:ModeOfBatteryCharging" />
        <xsd:element ref="tns:GuidePathLength" />
        <xsd:element ref="tns:WeightOfVehicle" />
        <xsd:element ref="tns:SafetySensorType" />
        <xsd:any minOccurs="0" maxOccurs="unbounded" />
        </xsd:sequence>
        </xsd:complexType>
        </xsd:element>
.....
</xsd:schema>
    </wsdl:types>
.....
</wsdl:definitions>

```

where the relevant property elements are defined as follows.

```

<wsdl:definitions ... xmlns:tns="http://localhost/agvviews" ...>
.....
    <wsdl:types>
        <xsd:schema targetNamespace="http://localhost/agvviews" ... >
            <!-- Resource property element declarations -->
            <xsd:element name="AGVType" type="xsd:string"/>

```

```
        <xsd:element name="GuidanceType" type="xsd:string" />
        <xsd:element name="ModeOfLoadTransfer" type="xsd:string" />
        <xsd:element name="LoadType" type="xsd:string" />
        <xsd:element name="LoadLength" type="xsd:integer" />
        <xsd:element name="LoadWidth" type="xsd:integer" />
        <xsd:element name="LoadHeight" type="xsd:integer" />
        <xsd:element name="LoadDiameter" type="xsd:integer" />
        <xsd:element name="MaximumLoad" type="xsd:integer" />
        <xsd:element name="MinimumLoad" type="xsd:integer" />
        <xsd:element name="LoadTransferHeight" type="xsd:integer" />
        <xsd:element name="NumberOfPickUpLocations" type="xsd:integer"
/>
    <xsd:element
        name="NumberOfDeliveryLocations"
type="xsd:integer" />
    <xsd:element name="BatteryType" type="xsd:string" />
    <xsd:element name="ModeOfBatteryCharging" type="xsd:string" />
    <xsd:element name="GuidePathLength" type="xsd:integer" />
    <xsd:element name="WeightOfVehicle" type="xsd:integer" />
    <xsd:element name="SafetySensorType" type="xsd:string" />
    ...
</xsd:schema>
</wsdl:types>
.....
</wsdl:definitions>
```

Now we can establish the association of the resource properties document with the port-Type as defined in the WSDL. This association defines the type of the WS-Resource as follows:

```
<!-- Association of resource properties document to a portType -->
<wsdl:portType name="PhysicalAGVView"
wsrp:ResourceProperties="tns:PhysicalAGVViewProperties" >
...
</wsdl:portType>
```

Other views can similarly be defined based on how the system is to be abstracted and exposed. For example, for the operational view the Resource properties document of the AGV is defined as follows.

```
<wsdl:definitions ... xmlns:tns="http://localhost/agvviews" ...>
.....
```

```

<wsdl:types>
<xsd:schema targetNamespace="http://localhost/agvviews" ... >
  <!-- Resource property element declarations -->
  <xsd:element name="AGVSpeed" type="xsd:integer"/>
  <xsd:element name="AGVMaximumSpeed" type="xsd:integer" />
  <xsd:element name="AGVAcceleration" type="xsd:integer" />
  <xsd:element name="AGVDeceleration" type="xsd:integer" />
  <xsd:element name="AGVBatteryVoltage" type="xsd:integer" />
  <xsd:element name="DistanceTravelled" type="xsd:integer" />
  <xsd:element name="AGVPosition" type="xsd:string" />
  <xsd:element name="ModeOfOperation" type="xsd:string" />
  <xsd:element name="AGVCurveRadius" type="xsd:integer" />
  <xsd:element name="OperationTimePerShift" type="xsd:integer" />
  <xsd:element name="NumberOfShiftsPerDay" type="xsd:integer" />
  <xsd:element name="CurrentZone" type="xsd:string" />
  <xsd:element name="NextZone" type="xsd:string" />
  <xsd:element name="PreviousZone" type="xsd:string" />
  <xsd:element name="AGVPriority" type="xsd:integer" />
  <xsd:element name="JobPriority" type="xsd:integer" />
  <xsd:element name="RoutingType" type="xsd:string" />
  <xsd:element name="CriticalThroughput" type="xsd:integer" />

  <!-- Resource properties document declaration -->
  <xsd:element name="OperationalAGVViewProperties">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element ref="tns:AGVSpeed"/>
        <xsd:element ref="tns:AGVMaximumSpeed" />
        <xsd:element ref="tns:AGVAcceleration" />
        <xsd:element ref="tns:AGVDeceleration"/>
        <xsd:element ref="tns:AGVBatteryVoltage" />
        <xsd:element ref="tns:DistanceTravelled" />
        <xsd:element ref="tns:AGVPosition"/>
        <xsd:element ref="tns:AGVCurveRadius" />
        <xsd:element ref="tns:OperationTimePerShift"/>
        <xsd:element ref="tns:NumberOfShiftsPerDay" />
        <xsd:element ref="tns:CurrentZone" />

```

```

        <xsd:element ref="tns:NextZone"/>
        <xsd:element ref="tns:PreviousZone" />
        <xsd:element ref="tns:AGVPriority" />
        <xsd:element ref="tns:RoutingType" />
        <xsd:element ref="tns:CriticalThroughput" />
        <xsd:any minOccurs="0" maxOccurs="unbounded" />
    </xsd:sequence>
</xsd:complexType>
</xsd:element>
...
</xsd:schema>
</wsdl:types>
.....
<!-- Association of resource properties document to a portType -->
<wsdl:portType name="OperationalAGVView"
    wsrp:ResourceProperties="tns:OperationalAGVViewProperties" >
.....
</wsdl:portType>
.....
</wsdl:definitions>
    
```

With the minimum additional efforts, the resource property documents and Web service port type can be defined in a similar way for the Performance and Financial views of the AGVs.

To summarize, using WS-RF, a resource (physical or logical) can be exposed according to the required abstraction or virtualization, conveniently providing different views of the resource as shown in Figure 3.

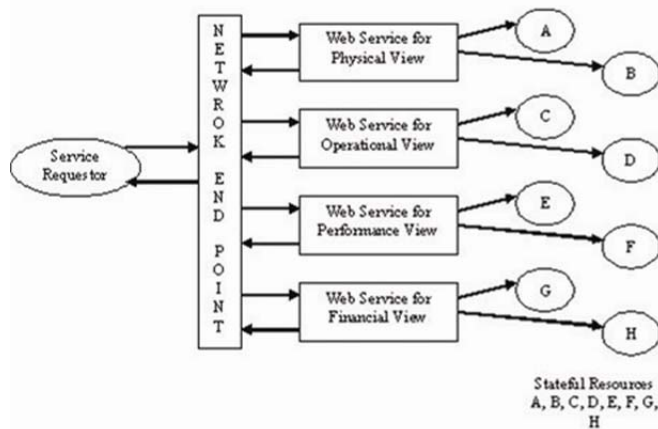


Figure 3: Virtualization provides different views

IV. Prototyping

WS-RF and related specifications do not dictate the means by which a service implements a resource properties document. As stated in [7], a given service implementation may choose to realize its implementation of the resource properties document as an actual XML instance document, stored in memory, in the file system, in a database or in some XML Repository. Other service implementations may dynamically construct the resource property elements and their values, from data held in programming language objects (such as a J2EE EJB Entity Bean) or by executing a command on a private communications channel to a physical resource. Yet another implementation possibility is a mapping layer to a standard management interface (such as CIM or SNMP).

In this section, we outline our design for the prototyping system which represents a virtualization of a physical GGV system. Each of the views mentioned in the previous section can be depicted as a WS Resource. A Web Service is created for each of the views and a WS Resource is created for each view for a particular AGV. The parameters of a view are depicted as ResourceProperties [7]. The Web Service itself can be used as a factory to create new WS Resources [2]. When the service requestor asks for a new WS Resource, a new WS Resource is created by the factory and the EPR (end point reference)[10] to the particular stateful resource i.e. the view for a specific AGV is returned to the service requestor. The service requestor can now access the different methods exposed by the web service operating on the stateful resource. Different methods exposed can be `getCurrentSpeed()`, `getCurrent-Position()`, `getPriority()`, `setPriority()`, `getBatteryVoltage()` etc. The service requestor can now get information on the state of the AGV as well as can perform some operations on the AGV by setting some parameters.

The prototype for virtualized AGV is implemented in the WSRF.Net environment.

V. Conclusion

Virtualization of manufacturing systems and exposing them as web services is a significant step in enterprise integration. Two Web standard technologies, OSGI and WSRF are explored to achieve virtualization. WSRF is chosen to virtualize an Automated Guided Vehicle (AGV) and expose it as a web service. Different views of an AGV are defined and the corresponding resource properties documents are generated. Each view is then exposed as a WS Resource. Virtualization enables the service requestor to access the different states of the manufacturing system and perform different operations on it in a standard web service environment.

VI. References

- [1] K. Czajkowski, D. Ferguson, I. Foster, J. Frey, , S. Graham, D. Snelling, and S. Tuecke. From Open Grid Services Infrastructure to Web Services Resource Framework: Refactoring and Evolution. Global Grid Forum. <http://www-106.ibm.com/developerworks/webservices/library/wsresource/grogsitowsrf.html>.
- [2] Karl Czajkowski, Donald F Ferguson, Ian Foster, Jeffrey Frey, Steve Graham, Igor Sedukhin, David Snelling, Steve Tuecke, and William Vambenepe. The WS-Resource Framework, 3 May 2004. White paper: <http://www-106.ibm.com/developerworks/library/ws-resource/ws-wsrf.pdf>.
- [3] Michael C Dempsey. Automated guided vehicle control systems: design application and selection. Society of Manufacturing Engineers Publication, 1986.

- [4] I. Foster, D. Berry, A. Djaoui, A. Grimshaw, B. Horn, H. Kishimoto, F. Maciel, A. Savva, F. Siebenlist, R. Subramaniam, J. Treadwell, and J. Von Reich. The Open Grid Services Architecture, Version 1.0. Global Grid Forum, 12 July 2004.
- [5] Ian Foster, Jeffrey Frey, Steve Graham, Steve Tuecke, Karl Czajkowski, Don Ferguson, Frank Leymann, Martin Nally, Igor Sedukhin, David Snelling, Tony Storey, William Vambenepe, and Sanjiva Weerawarana. Modeling stateful resource with Web services, V1.1, 3 May 2004. White paper. <http://www-106.ibm.com/developerworks/library/ws-resource/ws-modelingresources.html>.
- [6] Steve Graham, Anish Karmarkar, Jeff Mischkinsky, Ian Robinson, and Igor Sedukhin. Web Services Resource (WS-Resource) V1.2. OASIS, working draft 02 edition, 9 December 2004.
- [7] Steve Graham and Jem Treadwell (Eds.). Web Services Resource Properties (WSResourceProperties) v1.2. OASIS WSRF TC, 10 June 2004.
- [8] Gary Hammond. AGVS at Work: Automated Guided Vehicle Systems. IFS Publications, 1986.
- [9] S. Tuecke, K. Czajkowski, I. Foster, J. Frey, S. Graham, C. Kesselman, T. Maquire, T. Sandholm, D. Snelling, and P. Vanderbilt. Open Grid Services Infrastructure (OGSI), V 1.0. Global Grid Forum, 27 June 2003. GFD-R-P.15 (GGF Proposed Recommendation).
- [10] W3C. Web Services Addressing (WS-Addressing). World Wide Web Consortium, 10 August 2004. W3C Member Submission:<http://www.w3.org/Submission/2004/SUBM-ws-addressing-20040810/>.
- [11] W3C. Web Services Architecture. WorldWideWeb Consortium, 11 February 2004. W3CWorking Group Note.
- [12] OASIS Web Services Resource Framework (WSRF) TC, http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsrf.



Shivram Kumar Rajaram (shivram@pmail.ntu.edu.sg) is a graduating MSc Student at Nanyang Technological University, Singapore, doing his MSc in Communication Software & Networks. He received his B.E from Regional Engineering College, Trichy of India in 2002 in Instrumentation & Control. His research interests are in the areas of web services and grid computing.



Dr. Yang, Zhonghua is currently an associate professor at Information Communication Institute of Singapore, School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore. He has research interests in various aspects of Grid computing, Semantic Web and Semantic Grid, Autonomic and service-oriented computing, and software agents. He is a Programme Director (Grid Computing) in the School. Dr. Yang is involved primarily in teaching Web Services, Data Structures and Algorithms, Software Engineering, Software Systems and their development, and Distributed Computing.

Prior to the current appointment, Dr Yang had an extensive university and industry career which includes at Griffith University (Australia), University of Alberta (Canada), and Imperial College (London, UK), Distributed Systems Technology Center (DSTC), University of Queensland, Australia, Singapore Institute of Manufacturing Technology (SIMTech). Dr. Yang spent a significant part of his professional life with the Ministry of Aerospace of China, serving in various senior positions.



Dr. ZHANG, Jing Bing received his B.Eng degree from Tsinghua University (PR China) in 1985, PhD degree from Loughborough University (UK) in 1992, and MBA degree from Birmingham University (UK) in 2001. He is currently a Senior Scientist with Singapore Institute of Manufacturing Technology (SIMTech), and has assumed roles of research team leader, group manager, department manager, project leader/manger, and programme head since joined SIMTech in 1992.

His research covers networked control, intelligent performance management, condition-based maintenance, fault diagnosis/prognosis, and wireless sensor network. He has successfully led and managed numerous in-house and industrial R&D projects in above areas. He currently serves as editorial board member for International Journal of Services Operations and Informatics (IJSOI), virtual manufacturing committee (VGC) member of the National Grid Office (NGO) Singapore, vice-chair of IEEE IES TC on Integrated Manufacturing and Services Systems (IMSS), and Chairman of SIAA Academy, Singapore Industrial Automation Association. He has served as session organiser, session chair, technical committee member and international advisory panel member for many international conferences. In 2003, he received the IES Prestigious Engineering Achievement Award from The Institution of Engineers, Singapore



Liqun Zhuang (lqzhuang@simtech.a-star.edu.sg) is a senior research engineer in Singapore Institute of Manufacturing Technology (SIMTech). He received his B.E. from University of Science and Technology of China and M.E from Shanghai University of Technology in 1989 and 1992, respectively, in computer science. His research interests are in the areas of discrete event systems, distributed control and wireless sensor networks.