Enabling Dynamic Workflow for Disaster Monitoring and Relief Through Service-Oriented Sensor Networks

Lu Liu[1], David Webster[2], Jie Xu[2], Kaigui Wu[3]
[1]School of Engineering and Information Sciences, Middlesex University, London, NW4 4BT, UK
[2]School of Computing, University of Leeds, Leeds, LS2 9JT, U.K.
[3]College of Computer Science, Chongqing University, Chongqing, China
l.liu@mdx.ac.uk, {d.e.webster, j.xu, k.wu}@ leeds.ac.uk

Abstract

Natural and man-made disasters can significantly impact both people and environments. Sensor networks have the potential to revolutionize the capture, processing and communication of critical data for use of disaster rescue and relief [1]. In order to provide a dependable rescue capability through dynamically integrating newly developed and legacy sensor systems with other systems and computing, new methodologies are required for the dependable integration of services in heterogeneous environments. In this paper, we present a new architectural model which can proactively self-adapt to changes and evolution occurring in the provision of search and rescue capabilities in a dynamic environment. This approach has been demonstrated through developing and testing a demonstration system for a scenario of disaster area monitoring.

Keywords *– Service Oriented Architecture, sensor networks, disaster relief*

1. Introduction

Natural and man-made disasters can significantly impact both people and environments. For example, in the 2010 Haiti Earthquake, the Haitian Government reports that between 217,000 and 230,000 people had been identified as dead, an estimated 300,000 injured, and an estimated 1,000,000 homeless. Rescue efforts began in the immediate aftermath of the earthquake with great support from international communities. Real-time monitoring data and information is crucial for the success of disaster relief efforts from international communities.

In order to provide a dependable rescue capability through dynamically integrating systems and computing, new methodologies are required for the dependable integration of services in heterogeneous environments. A rescue capability is the operation of integrated services to fulfil a rescue mission objective. For example, in the analysis of floods, a number of different services provide information about - meteorology, topography, soil characteristics, vegetation, hydrology, settlements, infrastructure, transportation, population, socio-economics and material resources – that needs to be integrated to deliver a rescue capability for decision-making of disaster management agents.

Sensor networks have the potential to revolutionize the capture, processing and communication of critical data for use of disaster rescue and relief [1]. The functions of sensors need to be integrated to provide a joint service to meet different search and rescue requirements. For the provision of a dependable rescue capability in a dynamic and unpredictable disaster area, the networked sensors should have ability to autonomously support and co-operate with each other to quickly configure any services available on the disaster area to deliver a real-time capability, self-adaptability to modify their behaviours to deliver a sustainable capability according to environmental changes, and ability to share information, generate access and protect information throughout the network.

In this paper, we present a new architectural model which can proactively self-adapt to changes and evolution occurring in the provision of search and rescue capabilities in a dynamic environment. The rest of the paper is organised as follows. Section 2 discusses related work on service-oriented architecture (SOA) and service composition. The dependable dynamic service integration for delivery of rescue capability is discussed in Section 3. The demonstration system for disaster area monitoring to demonstrate the use of the architectural model for disaster relief is introduced in Section 4. In Section 5, conclusions are drawn and future work is described.

2. Related Work

The use of SOA has been motivated by many industries changing focus from product delivery to service-based delivery. The focus on service delivery has also been apparent in software, where networking has become faster, more reliable and more available through reduced cost. The approach to SOA in software enables business process integration that characterizes business functions as services, and integrates dynamically across departments and organizations. The SOA can be used to

CHINACOM 2010, August 25-27, Beijing, China
Copyright © 2011 ICST 973-963-9799-97-4
DOI 10.4108/chinacom.2010.129

integrate businesses, systems and computing at runtime [2] by using different levels of abstraction.

Web services are a de facto implementation of SOA. In the last decade, a number of Web Service composition frameworks and applications have been developed. Alonso et.al [3] described six different dimensions of web service composition models which can make different assumptions of the types of components considered. The disadvantages of composition models makes composition work more involved because of the heterogeneity of the components. The Web Services Composite Application Framework (WS-CAF) [4] is an open framework developed by the OASIS group. The purpose of the OASIS WS-CAF is to define a generic and open framework for applications that contain multiple services used in combination.

eFlow [5], developed by Hewlett Packard, is a system for the specification, enactment and management of composite services [6]. Composite services are modeled by a graph which defines the flow of service invocations. eFlow provides the dynamic features to cope with the rapidly evolving business environment where Web services are used. BPEL [7] is a standard business process execution language which forms the necessary technical foundation for multiple usage patterns including both the descriptions of the process interface for business protocols and executable process models.

3. Design for Evolution

Service-oriented architecture (SOA) is designed for changes. Loose coupling is one of the key architectural principles of SOA, and this enables services to maintain a relationship that minimises dependencies and only requires maintaining an awareness of each other. The loose coupling of SOA enables service implementations to be inter-changed and modified where integration interfaces are developed with minimal assumptions between the sending/receiving parties, thus reducing the risk that a change in one service will force a change in another service. However, service integration is dependent on service interface definitions and requires management of P2P workflow definitions to minimise impact on provision of rescue capability. The changes of requirements and workflow definitions could still affect the dependability and sustainability of provision for a rescue capability through service integration.

In order to ensure the reliability of provision of rescue capabilities, a systematic approach needs to be developed which would lead to flexible architectures for through-life evolution. Figure 1 illustrates service-oriented architecture for the delivery of rescue capabilities. In this architecture, each peer node provides a number of services, each service performs a set of functions, and these can be integrated to form a higher level of functionality to deliver a rescue capability. Dynamic binding allows common functions to be identified in different implementations. The architecture enables functions from different services across sensors to be integrated to provide rescue capabilities in a loosely coupled manner. For example, a remote sensing sensor on a rescue helicopter provides services, such as surveillance, target recognition and target tracking services. The surveillance service includes functions for metrological surveillance and situation surveillance. The metrological surveillance function may be combined with other functions in a workflow to form a higher level weather service that contributes to search-and-rescue missions.

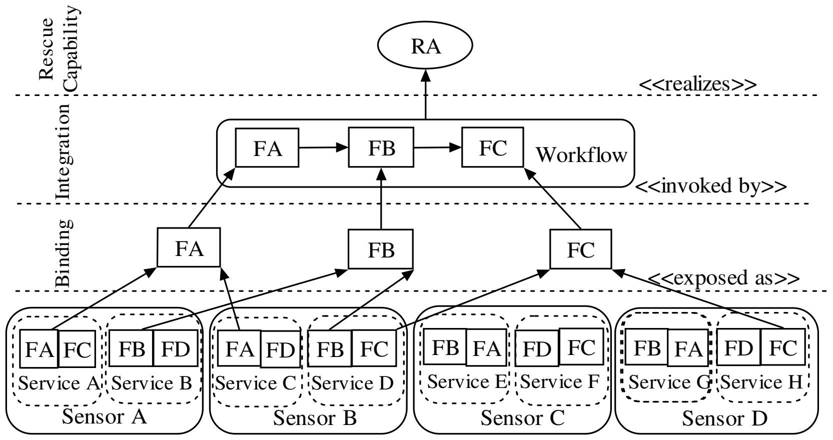


Figure 1. Redundant service binding

3.1 Workflows for Disaster Relief

In this section, a disaster area monitoring scenario is used as an exemplar to demonstrate the evolution of rescue capabilities and to analyse the potential impact of evolution.

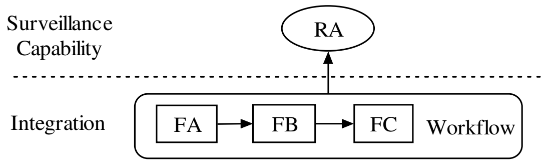


Figure 2. Workflow of service integration for delivery of rescue capabilities

In a network of sensors, a number of radar sensors supply data through services. The network of radar sensors is modelled conveniently as a dynamic network of services, facilitating ongoing changes. In the modelled system, a surveillance user can submit real-time requests to the system for information of Points of Interest (POIs) in a specified region. A sequence of services (such as "Get map information" (*FA*) and "Get sensor reading" (*FB*), "Display targets on map" (*FC*) can be operated in a workflow in order to provide a regional surveillance capability. The service integration can be abstracted by using workflow patterns as shown in Figure 2, where function *FA* represents the service of getting map information, *FB* represents the service of getting radar reading and *FC* represents the service of displaying targets on map.

Redundant service binding is a technique to improve the reliability of the provision of rescue capabilities [8]. In order to deliver a reliable rescue capability, the required functions need to be provided by multiple services allocated to different peer nodes. The reconfiguration algorithm can switch to one of backup services in case of failure of initial service. The distributed recovery block (DRB) scheme [9] is applied to minimise the recovery time of integration. Figure 1 shows an example of redundant service binding. As shown in Figure 1, when a failure occurs in service *A*, the required function provided by the backup service *C* can still work for the provision of rescue capabilities.

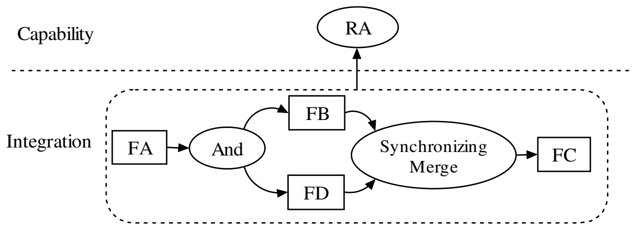


Figure 3. A possible evolution

Apart from failures of services, rescue capability evolution could cause potential problems affecting the reliability of provision of rescue capabilities. Requirements often change during a rescue and search operation, in accordance with changes of situation. The rescue capability could be evolved according to changes of environment and users' needs. A possible evolution is illustrated as an example: it is decided that some POIs are more important and a more aggressive monitoring service needs to be established. A number of rescue helicopters with remote sensing sensors are launched to provide a mixed monitoring service in conjunction with the deployed radars. The rescue capability is evolved with the additional requirement to a new version of capability defined with a parallel split workflow pattern and synchronizing merge workflow pattern [10] illustrated in Figure 3. The parallel split pattern represents a point in the workflow process where a single thread of control splits into multiple threads of control which can be executed in parallel, while synchronizing merge pattern represents a point in the workflow process where multiple paths converge into one single thread; Synchronization needs to take place if more than one path is taken [10]. The new capability could be delivered only in case that both the services: *FB* and *FD* are implemented successfully.

3.2 Self-adaptability

The reliability of provision of service integration could be affected by the evolution of workflows in the case mentioned above. The proposed reconfiguration algorithm is able to proactively self-diagnose the evolution, evaluate the impact of evolution and self-configure services to adapt to capability evolution. In order to theoretically investigate the impact of the evolution of capabilities, p is defined as the probability of failing to connect a service for integration. By configuring two services which are independent with each other for performing a required function as shown in Figure 1, the required function is delivered if either service is implemented successfully. The probability of failure of the delivery of a required function for service integration is p^2. In the original case without evolution, three functions are integrated in a workflow to deliver a rescue capability. Since all three functions are necessary for the provision of a rescue capability, the probability of successful service integration is $(1-p^2)^3$ in the original example.

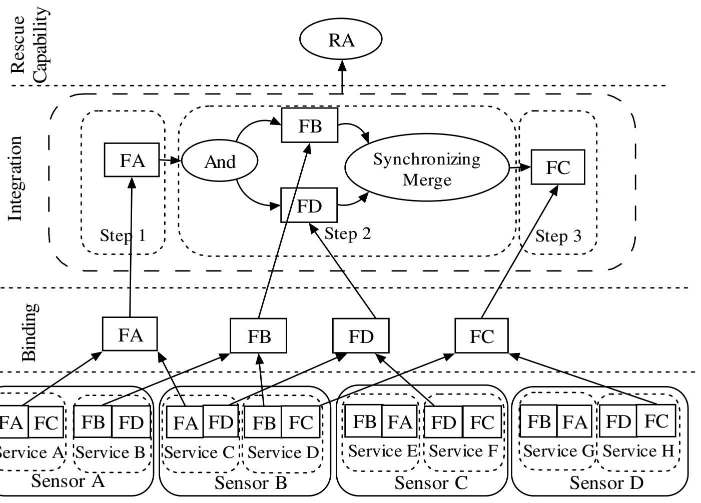


Figure 4. Evolution of rescue capability

In this scenario, the rescue capability could be delivered only when both *FB* and *FD* are available. The probability of successful provision of rescue capability is $(1-p^2)^4$. Since $(1-p^2)^4 \le (1-p^2)^3 \quad (0 \le p \le 1)$, the reliability of rescue capability is decreased. For a better understanding of impact of evolution, the workflow is divided into three steps as shown in Figure 4. The theoretical success probabilities of the three steps are: $1-p^2$, $(1-p^2)^2$, and $1-p^2$, respectively in this case. The step two is the weak point in the workflow. Its success probability is lower than the original success probability prior to evolution: $P_{evo} = (1-p^2)^2 \ge P_{org} = 1-p^2 \quad (0 \le p \le 1)$. To address this issue, the reconfiguration algorithm should have self-adaptability to proactively modify its behaviour to deliver a sustainable rescue capability according to the environmental changes. The redundancy of service binding that is the number of services providing a required function is a key parameter and designated as *R*. The redundancy needs to be dynamically self-justified, in order to adapt to capability evolution. As discussed above, the success probability of the second step is evaluated as $P_{evo}^R = (1-p^R)^2$.

For sustainable provision of rescue capability, more services ($R > 2$) need to be added and configured to provide each function *FB* and *FD* to enable its Cumulative Distribution Function (CDF) smaller than the original CDF prior to the evolution:

$$\int_0^1 (1-x^R)^2 dx \ge \int_0^1 (1-x^2) dx . \qquad (1)$$

The inequality (1) is satisfied only in the case of $R \ge 4$. The minimum value $R = 4$ is adopted to minimise the cost of the sustainable provision of rescue capability as illustrated in Figure 5.

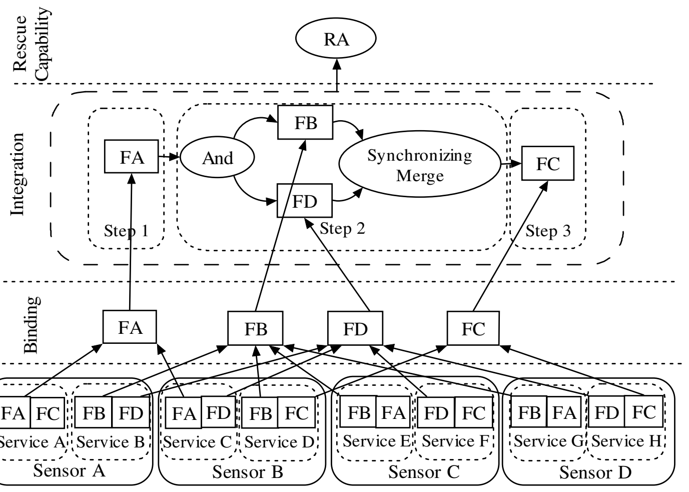


Figure 5. Evolution with adaptive service reconfiguration

4. Demonstration System Development

A demonstration system has been developed to demonstrate the use of the architectural model for the dynamic service integration of a network of sensors on a disaster area and to provide a search and rescue capability. The core of the approach is the process of mapping high-level requirements for capability onto the invocation of actual services, allowing the establishment of a dynamic workflow of service composition and integration, and dynamic search for services and on the fly planning through dynamic integration of services. The competitive advantage, such as timeliness, reliability and fault tolerance, can be achieved through the dynamic service discovery, composition and integration.

The software demonstrator was developed in NetBeans 6.1 IDE on a GlassFish application server, which enabled our research group members to write, deploy, test and debug SOA applications using the Extensible Markup Language (XML), Business Process Execution Language (BPEL) and Java Web Services. The Software Demonstrator consists of the following main modules:

- Web Services to simulate capabilities of different systems;
- A dynamic workflow module for dynamic Web Services selection and composition for dependable provision of rescue capability;
- A client interface with sensor information display and user input.

The workflow dynamically discovers and integrates the sensor services and is implemented in a Java written Web Service.

The scenario aim of the software demonstrator was to model Region Surveillance using dynamic service integration of sensor networks on a disaster area. The intent is to demonstrate the architectural approach to engineering. The main concepts are:

- Use of SOA for disaster relief enhanced with other architectural styles and patterns;
- Integration of distributed systems in a dynamic environment;
- Coping with changes in availability of distributed components;

4.1 Exposing a Legacy Sensor Application to an SOA Network

As shown in Section 3.1, a legacy sensor application is exposed as a Web Service in order to allow it be integrated into an SOA enabled workflow for the provision of a regional surveillance capability. This system model can be summarized with UML as in Figure 6. In this model a service provider ("Sensor Service Provider") and client "Sensor Client" are defined. The Web Service ("Sensor Wrapper") server is hosted within the "Sensor Service Provider" along with the legacy application ("readsensor") that communicates with the physical sensor hardware. "Sensor Wrapper" acts as both an external facing Web Service and a wrapper to the "readsensor" application. The wrapping implementation of the `readsensor' application to a GlassFish hosted Web Service was shown in Program 1.

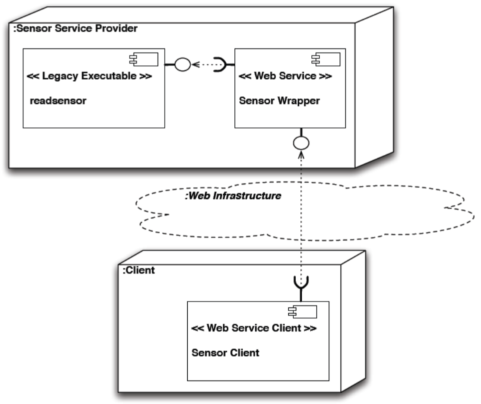


Figure 6. UML diagram of wrapped sensor System of Systems

Program 1 Simple Sensor Wrapper program in Java

```
/**
* Simple Sensor Wrapper
*/
@WebMethod(operationName = "getSensorImage")
public String getSensorImage()
{
    /**
    * Launch 'readsensor' program and read the output
    * represented by plain text with base64 encoding.
    */
    Runtime runtime = Runtime.getRuntime();
    Process proc;
    StringBuffer programOutput
    = new StringBuffer();
    try
    {
          proc = runtime.exec("readsensor");
          InputStream inputstream = proc.getInputStream();
          InputStreamReader inputstreamreader
            = new InputStreamReader(inputstream);
          BufferedReader bufferedreader
            = new BufferedReader(inputstreamreader);
          // read the program output
          String programOutputLine;
            while (
            (programOutputLine = bufferedreader.readLine())
            != null )
      {
      programOutput.append(programOutputLine);
      }
    } catch (Exception ex) { return null; }
    /**
    * Return the wrapped program output to the
    * Web Service client.
    */
    return programOutput.toString();
}
```

4.2 Dynamic Service Integration

The surveillance was based on Points of Interest, PoIs and physical features within a geographical area that are detected by a group of simulated sensors. The integration of sensors was achieved by using a workflow to contact a sensor service registry and to dynamically discover sensors for a given region.

The sensor data is then processed to eliminate duplicates and points outside the region of interest and the detected feature positions are displayed on a map. The workflow can be illustrated at a high level in Figure 7. This diagram can be directly mapped onto the SOA integration model shown in Figure 7 and illustrates the implementation technology used in the lower boxes. The workflow integrates Web Services which were the chosen demonstration implementation technology for SOA. The implementation of region surveillance used Google Maps to display the results from the feature detection workflow.

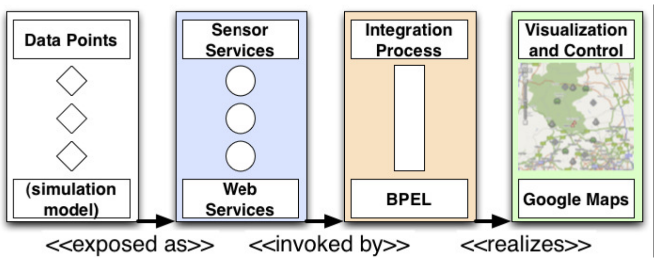
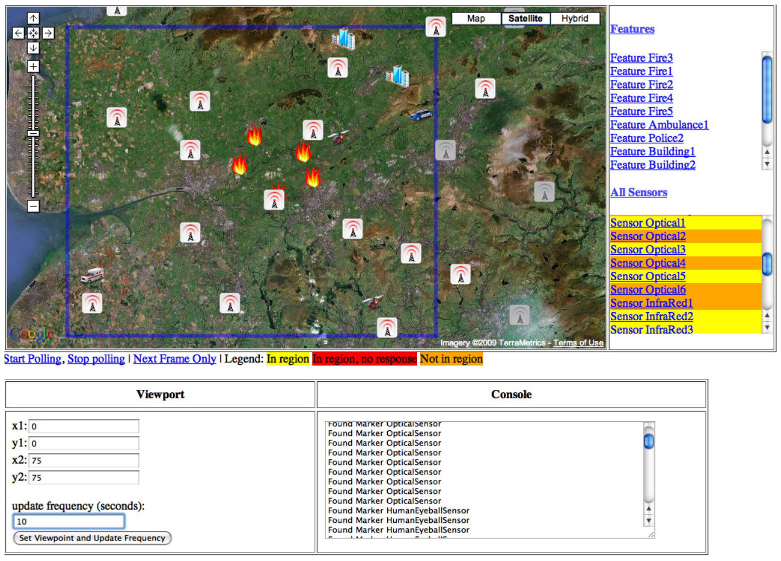


Figure 7. A high-level overview of the sensor integration workflow



Features

Feature Fire3
Feature Fire1
Feature Fire2
Feature Fire4
Feature Fire5
Feature Ambulance1
Feature Police2
Feature Building1
Feature Building2

All Sensors

Sensor Optical1
Sensor Optical2
Sensor Optical3
Sensor Optical4
Sensor Optical5
Sensor Optical6
Sensor InfraRed1
Sensor InfraRed2
Sensor InfraRed3

Start Polling, Stop polling | Next Frame Only | Legend: In region In region, no response Not in region

Viewport	Console
x1: 0 y1: 0 x2: 75 y2: 75 update frequency (seconds): 10 Set Viewpoint and Update Frequency	

```
Found Marker OpticalSensor
Found Marker OpticalSensor
Found Marker OpticalSensor
Found Marker OpticalSensor
Found Marker OpticalSensor
Found Marker OpticalSensor
Found Marker OpticalSensor
Found Marker OpticalSensor
Found Marker OpticalSensor
Found Marker HumanEyeballSensor
Found Marker HumanEyeballSensor
Found Marker HumanEyeballSensor
```

Figure 8. Disaster monitoring showing PoIs

A screenshot from this interface can be shown in Figure 8. In the system, a disaster relief volunteer can submit real-time requests for information of Points of interest, POIs, in a specified region. The system will return the related information about the POIs within that region, e.g., current locations of those POIs (Figure 8). The blue rectangle is the region of interest. The lists on the right of the image show the detected features and the sensors that have been accessed in the workflow. The system is built on a dynamic and changing environment, where sensor services in region may fail to respond with information about the POIs as shown in Figure 8. By using the approach proposed in Section 3, multiple sensor services are contacted to receive the data about POIs in the requested region.

The demonstration system incorporates the following innovations to achieve competitive advantage:

- *Information-Rich Information Services*: provide description of services, composition templates with candidate composed services, application workflows, architectural patterns, application patterns, evaluation information [11].
- *Evolving Ontology*: ontology available for dependability, capability, system assessment [12].
- *Service Interoperability*: advanced techniques for dynamic authentication and run-time negotiation [13].
- *Optimisation for On-the-Fly Planning*: based on a tool [13] that supports the use of a variety of optimization techniques and their combination.

5. Conclusion

In this paper, an architectural approach that facilitates through-life system evolution has been presented, using the concepts of dynamic workflow management to enable dynamic service integration for reliable and sustainable provision of rescue capabilities. This approach is able to indentify evolution, evaluate the impact of evolution and self-configure services to adapt to evolution.

This approach has been demonstrated through developing and testing the demonstration system for a scenario of disaster area monitoring. The wrapping of legacy sensor systems to a SOA network has been discussed. The development of the demonstration system has been used to ascertain that the implementation the architecture is fit for use. Further development of the demonstration system will be used for further evaluation of disaster management systems.

Acknowledgment

The work reported in this paper has been supported by Major Program of the National Natural Science Foundation of China, under contact No.90818028.

References

[1] E. Cayirci and T. Coplu, "SENDROM: Sensor networks for disaster relief operations management," *Wireless Networks,* vol. 13, pp. 409-423, June 2007.

[2] W.-T. Tsai, X. Zhou, and Y. Chen, "PESOI: Process Embedded Service-Oriented Architecture," *Journal of Software,* vol. 17, p. 1470–1484, June 2006.

[3] G. Alonso, F. Casati, H. Kuno, and V. Machiraju, *Web Services Concepts, Architectures and Applications*: Springer Verlag, 2004.

[4] OASIS, "Web Services Composite Application Framework (WS-CAF)," [Accessed: Available from: http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=ws-caf.

[5] F. Casati, S. Ilnicki, LiJie Jin, V. Krishnamoorthy, and M.-C. Shan, "Adaptive and dynamic service composition in eFlow," in *12th International Conference on Advanced Information Systems Engineering(CAiSE)*, 2000.

[6] F. Casati and M.-C. Sha, "Dynamic and adaptive composition of e-services," in *12th International Conference on Advanced Information Systems Engineering (CAiSE 00)*, 2001.

[7] T. Andrews, F. Curbera, H. Dholakia, J. Klein, F. Leymann, K. Liu, D. Roller, D. Smith, S. Thatte, I. Trickovic, and S. Weerawarana, "Specification: Business Process Execution Language for Web Services Version 1.1," [Accessed: Available from:http://www-106.ibm.com/developerworks/library/ws-bpel/.

[8] L. Liu, D. Russell, D. Webster, Z. Luo, C. Venters, J. Xu, and J. Davies, "Delivering Sustainable Capability on Evolutionary Service-oriented Architecture," in *IEEE International Symposium on Object/component/service-oriented Real-time distributed Computing (ISORC 2009)*, Tokyo, Japan, 2009.

[9] K. H. Kim and H. Welch, "Distributed Execution of Recovery Blocks: An Approach for Uniform Treatment of Hardware and Software Faults in Real-Time Applications," *IEEE Transactions on Computers,* vol. 38, pp. 626-636, 1989.

[10] W. v. d. Aalst, A. t. Hofstede, B. Kiepuszewski, and A. Barros, "Workflow Pattern," *Distributed and Parallel Databases,* vol. 14, pp. 5-51, 2003.

[11] W.-T. Tsai, X. Zhou, Y. Chen, and X. Bai, "On Testing and Evaluating Service-Oriented Software," *IEEE Computer,* vol. 41, pp. 40-46, 2008.

[12] D. Webster, N. Looker, D. Russell, L. Liu, and J. Xu, "An Ontology for Evaluation of Network Enabled Capability," in *Realising Network Enabled Capability (RNEC'08)*, Leeds, United Kingdom, 2008.

[13] P. Townend, J. Huai, J. Xu, N. Looker, D. Zhang, J. Li, and L. Zhong, "CROWN-C: A High-Assurance Service-Oriented Grid Middleware System," *IEEE Computer,* vol. 41, pp. 33-38, 2008.