Identifying Performance Objectives to Guide Service Oriented Architecture Layers

Tehreem Masood(✉), Chantal Bonner Cherifi, and Néjib Moalla

Decision and Information Sciences for Production Systems (DISP),
Université Lumière Lyon 2, Lyon, France
{tehreem.masood, chantal.bonnercherifi,
nejib.moalla}@univ-lyon2.fr

Abstract. Service oriented architecture is emerging as a powerful paradigm for organizations that need to integrate their applications within and across organizational boundaries. Organizations need to take decisions more quickly and need to change those decisions dynamicaly. Delivering an adequate level of performance is a critical and significant challenge that requires monitoring along the different layers of service oriented architecture. Current monitoring systems are designed to support specific layers but do not fulfil the requirements of all the layers of service oriented architecture. Ontologies on the semantic web standardize and formalize the concepts and store domain knowledge for effective decision making. In this paper, we propose performance monitoring framework for various layers of service oriented architecture. It integrates various ontologies to monitor the performance at the service oriented layers in order to ensure their sustainability. We design a Service Performance Ontology that captures all the information about the service domain. Along with that we design ontologies for ensuring performance at service level, binding level, composition level and server level. We conduct a performance evaluation over real web services using suitable estimators for response time, delay, loss and more.

Keywords: Web services · Service Oriented Architecture (SOA) · Performance · Decision making · Ontology

1 Introduction

Web Services is a software system designed to support interoperable machine to machine interaction over a network [1]. It uses open standards like XML [2], SOAP [3], WSDL [4]. Web-service based applications are applications built by using web services provided by third-parties. The SOA Reference Architecture (SOA RA) has nine layers that emerge in the process of designing an SOA solution or defining an enterprise architecture standard [5]. SOA RA is shown in Fig. 1.

The component services access the Data abstraction layer to fetch and retrieve data. Messaging through SOAP provides the ability to perform the necessary message transformation to connect the service requestor to the service provider and to publish and subscribe messages and events asynchronously [6]. In this way services are published in the service layer. On the top level layer, BP applications provide process

S. Hammoudi et al. (Eds): MODELSWARD 2016, CCIS 692, pp. 216–226, 2017.
DOI: 10.1007/978-3-319-66302-9_11

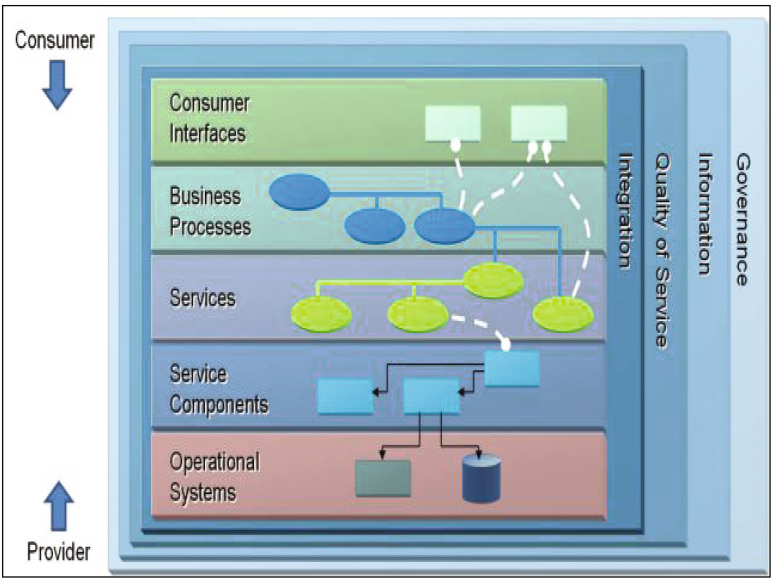

Fig. 1. SOA ref Layers [5].

orchestration mechanism to execute enterprise business processes. BP application use Business Process Modeling Notation (BPMN) [7] to design their business processes. Governance rules are the set of policies like service will be available for one year etc. Security is used to provide some integrity to the system like authentication with the help of user name and password. Quality of service layer is used to provide the agreed quality as defined in service level agreements.

Ontologies play an important role in both semantic web applications and knowledge engineering systems [8]. Numerous tasks such as information storage, processing, retrieval, decision making etc. are done on the basis of ontologies.

Web service performance metrics are classified as client-side and server-side. Load generators are used together with profiling tools to measure and optimize the performance of web service based applications. Examples are JMeter [9], soapUI [10], Eclipse TPTP [11] and JProbe [12].

Software performance engineering is a systematic approach to construct software systems that meet performance objectives (PO's). As part of software performance engineering, we focus on establishing PO's. PO's can be monitored at service or process execution time. PO's that we have specified are related to availability, reliability, response time, throughput, resource utilization and bandwidth of SOA layers. It helps to determine that how the SOA layers performs under a particular workload. It demonstrates that the system meets performance criteria.

Companies need to know the performance of web services for a number of reasons. Some of the reasons are: Clients need to know and demand efficient PO's. Multiple service providers offering same or similar service take advantage. Also, It is required to know the resource demands of the system at different workloads.

The analysis of requirement phase shows the diversity of measurements that are applied in different service networks together with the demand for a common framework. It derives a list of performance objectives being of common interest. These performance objectives can be retrieved by using different business activity monitoring tools like Oracle Business Activity Monitoring [13], WSO2 server [14]. Performance objectives of primary interest for the users are related to the performance degradations that include availability, response time, delay, failure and loss of service.

Our objective in this research is to propose a decision-making model combining service/process performance objectives in order to generate validation arguments for the expansion of the service environment. Therefore in this paper, we propose a Performance Monitoring Framework based on the Service SOA. The decision support covers the unavailability of services, recommendation of services based on key performance indicators and the composition of existing ones.

In this paper, we introduce a performance monitoring framework for SOA layers. The framework consists of two major steps (i) Service monitoring ontology (ii) Ontologies for performance measurement of service oriented activities.

The remaining of the paper is organized as follows: Sect. 2 includes related work. Section 3 discusses our proposed performance monitoring framework. We conclude our work in the last section.

2 Requirements

Performance monitoring framework that is addressed in this paper should be able to fulfill the requirements of different user perspectives.

2.1 Service Network or Business Activity Users

In this case the framework shall provide a view of the service network which allows to easily track the failing service by analyzing the network of services. Service network involves service tasks and business processes. Service task issues a single business action for the most part. Its interactions involve the execution of a single business function or an inquiry against a single core business entity. Business Process is a sequence of tasks triggered by business events. It issues a series of business actions that could involve the invocation of one or more granular business services. This perspective is related to those users who are managing the network of services or business activities to identify and replace the faulty service. This perspective is important to address business activities are the core part of a value chain in the enterprise.

2.2 Performance Emergency Response Team (PERT)

It is important to gather the information from various sources to get clear view of performance degradations. It can be done by setting up performance objectives for monitoring along SOA layers. Performance objectives to analyze performance degradations are availability, response time, delay, loss and many others. There are different tools available to monitor these performance objectives. It is extremely important to

manage the performance problems by providing some alternate solutions. Framework shall provide a mechanism to recommend alternate solutions under these circumstances.

3 Related Work

This section is divided into four parts. The first part is related to the end user requirement in performance monitoring. The second part deals with Performance based monitoring projects related to SOA paradigm are evaluated. The third part deals with the decision making models to support SOA. The fourth part evaluates the Ontology based QoS Analysis Techniques.

3.1 Performance Based Monitoring Projects

INTERMON [15] project is inter-domain QoS monitoring. This project is based on abstractions for traffic, topology. It centralizes the collection of pre-defined measurements. It cannot be guaranteed that an entity has a complete control of other networks.

JRA1 [16] project is inter-domain QoS monitoring. This architecture uses authentication and authorization rules, schedule new types of measurements. It provides metrics concatenation and aggregation. The constraint is requesting data for day-to-day operation of the networks. Other constraints are to allow distributed policies among the different networks, for exchange of monitoring data and access to on-demand test tools.

The MonALISA project [17] also provide a framework for distributed monitoring. It consists of distributed servers which handle the metric monitoring for each configured host at their site and for WAN links to other MonALISA sites. This system relies on JINI for its discovery service.

The PlanetLab [18] is a huge distributed platform over 568 nodes, located in 271 different sites. It enables people being members of the group to access the platform to run networking experiments. PlanetLab infrastructure service is centralized and relies on a single database. Similar to INTERMON it can therefore not be applied as is to a multi-domain environment.

The perfSONAR framework [19] is a service-oriented monitoring architecture to monitor the performance in multi-domain networks. It is mainly used in large-scale environments and less suited for service monitoring where fine quality is important (such as for video quality monitoring). The focus of perfSonar is on troubleshooting across network boundaries.

Top-Hat [20] federate different measurement infrastructures to provide researchers with valuable information during the discovery and selection of resources for their experiments. It interconnects measurement systems to monitor performance in multi-domain networks. It provides experiment support. This tool is used to ping a large amount of hosts with the least possible resources.

3.2 Decision in SOA

Decision as a Service [21] separates the decisions logic from the application logic. The decision process is completely put into one service. It is a method used to describe decision logic in scenarios with many input parameters. It allows to use separate services for every step of the process locally.

SOA+d [22] creates an approach bridge for the gap between SOA and decision automation. They focus on intelligent design choice model of Simon to suggest this meta-model. They integrate elements into two dimensions: conceptual and methodological. New concepts like concept of decision, intelligence, design and the choice service. Meta model purpose is to define, organize and reuse knowledge about concepts involved in the business processes and their decisional aspect as well as their design and implementation based on services and the relationship between them.

3.3 Ontology Based QoS Analysis Techniques

MonONTO [23] provides ontology to propose recommendation for the advanced internet applications users. They have considered information concerning the application type, traffic generated and user profile along with network performance metrics. Their expert system monitors the performance of advanced internet applications. Their ontology serves as a support to a decision reference tool by providing high-level information to the user about the agreement of the network facing the service level demands. They have used a fixed list of network parameters. Therefore, it does not deal with the heterogeneity and extensibility issues. Implementations of web services have not been done by them. Additionally, it does not deal with QoS mapping.

Another technique named as Semantic Web Service Discovery Based on Agents and Ontologies [24] considers the fuzzy constraints. Their framework is modelled by adding semantics of QoS attributes with web service profiles. It describes the design and implementation of a web service matchmaking agent. Agent uses an OWL-S based ontology and an OWL reasoner to compare ontology based service descriptions. They have used fuzzy constraints increases the efficiency of the web service discovery approach by providing the customers the web services which are not actually satisfying the input QoS constraints, but are close to the QoS constraints specification.

4 Performance Monitoring Framework

This section presents our Performance Monitoring Framework as shown in Fig. 2. The first step of our framework is the requirement to use service or business process. This analysis is typically performed on user requirement or business goals in a certain time period to create a specification of requirements.

Next step is the identification, defining and collection of performance objectives. Performance requirements of SOA layers are monitored in terms of PO's. These PO's have target values that are required to achieve in a certain time period. We identify PO's at the service layer, orchestration layer, resource layer and binding level. We design performance based ontologies for service oriented architecture layers.

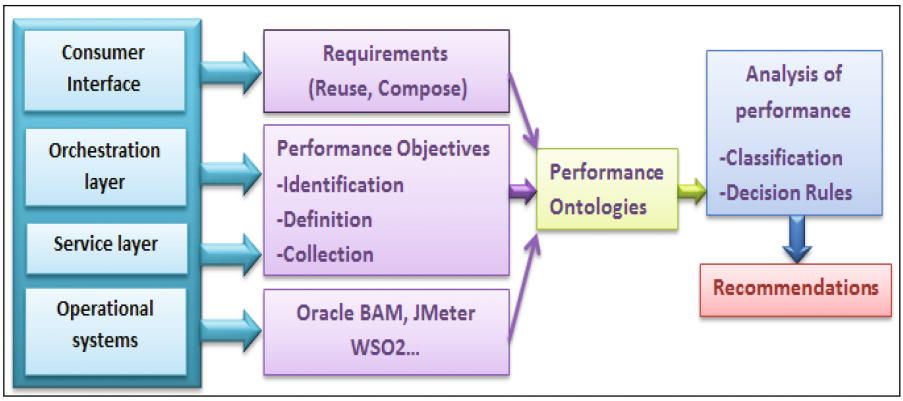

Fig. 2. Performance Monitoring Framework.

We classify these performance objectives and define decision rules based on these ontological concepts to make recommendations like yes and no to reuse existing service.

Major steps for the creation of performance profile are:

- Specifying ontological concepts
- Management of common concepts
- Quality assurance on the performance profile

In the following sub sections we explain service performance ontology and ontologies at the service layer, orchestration layer, resource layer and binding level.

4.1 Service Performance Ontology

In this step we design Service Performance ontology (SPOnt) as a base infrastructure. It aggregates the main concepts and relationships between them. QoS requirements, service domain concepts, key performance indicators and performance levels are the major concepts. SPOnt is shown in Fig. 3.

Service
This concept has various data type properties to capture different attributes in SPOnt. It also has various object properties that links Service concept to other concepts. The details are as follows:

Performance Level
This concept conceptualizes the level where a service network can be monitored. It has various sub concepts, each for capturing the performance levels such as Domain Level, Node level, Server Level, Service Level, Operation Level, and Messaging Level.

QoS Level
Quality of service model is classified as metrics into time based, size based and combined (both time based and size based) metrics. Key performance indicator (KPI)

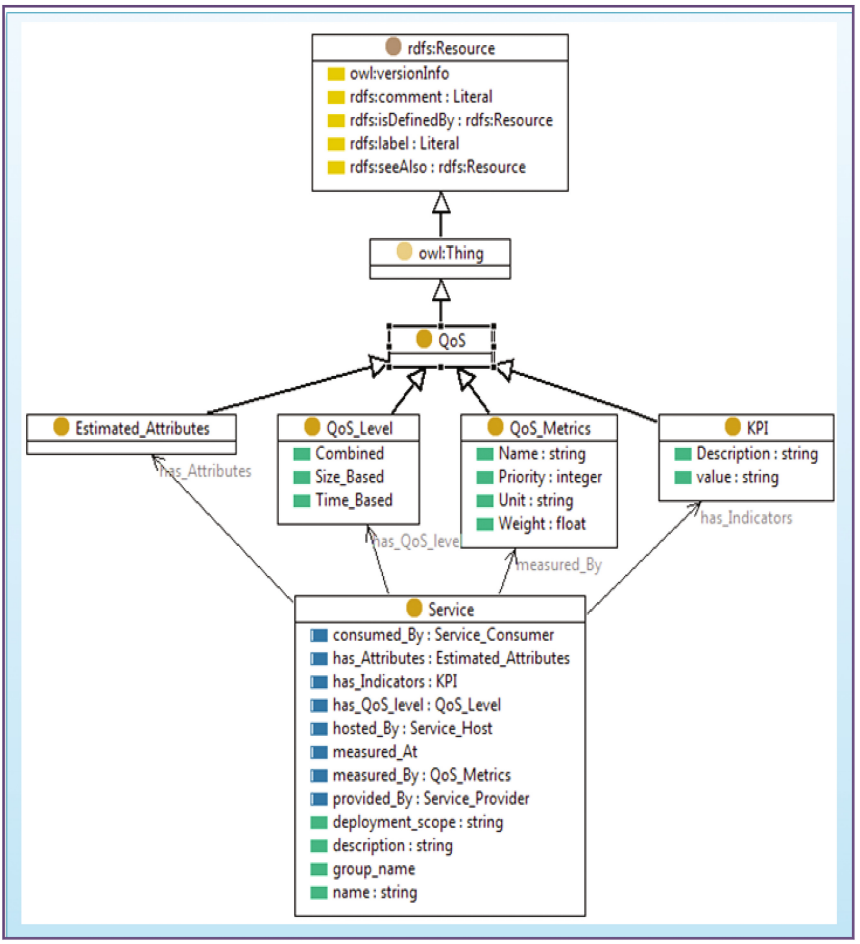

Fig. 3. Service Performance Ontology (SPOnt).

assessment model has classified the indicators as response time, delay, error, loss, SLA, number of operations per second and average data blocks per time unit.

Time based

Time based classification includes all those indicators that can be measured in time units like availability, delay, response time. Availability is defined as the total down time per service. Delay is defined as downtime divided by uptime. Response time is also called latency. It is the time perceived by a client to obtain a reply for a request for a web service. It includes the transmission delays on the communication link. It is measured in time units.

Size based
Size based classification includes all those indicators that can be measured in size units. For example reliability. Reliability can be analyzed as loss or error of service. It is measured as number of successful invocations divided by total number of invocations.

Combined
Combined based classification includes all those indicators that can be measured by both time and size units like bandwidth and throughput. Bandwidth is defined as the tasks per time unit and average data blocks per time unit. Throughput is defined as the number of operations per second.

4.2 Performance Objectives for SOA Layers

In this step, we explain the ontological concepts of the performance at service layer, orchestration layer, resource layer and binding level. Ontologies for all these layers are shown below step by step. Figure 4 shows the ontology of performance at service layer. PO's at service layer are explained below.

Response Time: captures the response time of a service/operation. It has three sub concepts to record Maximum, Minimum and Average response time.
Request Count: shows the number of invocation of a service.
Response Count: shows the number of replies for an invocation of a service.
Fault Count: shows the number of invocations the service has not replied.
Deploy Time: shows when the service is deployed at the server
Up Time: shows the time period the service is available since its deployment

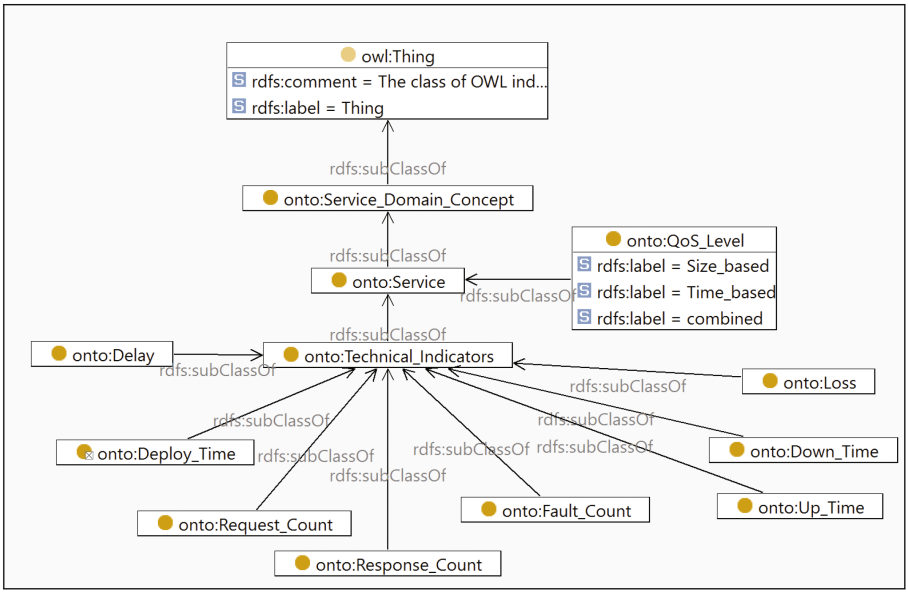

Fig. 4. Ontology of Performance at Service Layer.

Down Time: shows the time period of un-availability of a service since its deployment
Delay: shows the average response time of a service.
Loss: shows that the service is un-available (i.e., it cannot be invoked).

Figure 5 shows the ontology of performance at orchestration layer. PO's at orchestration layer are explained below.

Process-Response-Time: captures the response time of a business process. It has three sub concepts to record Maximum, Minimum and Average response time.
Process-Up-Time: shows the time period the business process is available since its deployment
Process-Down-Time: shows the time period of un-availability of a business process.
Process-Delay: shows the average response time of a business process.
Process-Loss: shows that the business process is un-available (i.e., it cannot be invoked).
Process-Duration: shows the time duration of business process since it is deployed, executed and remained in process.

Some other PO's at service layer are also used in order to estimate their value at composition level like availability and service response time.

Figure 6 shows the ontology of performance at binding level. Binding level means at the messaging level. PO's at transport messaging level are binding-throughput,

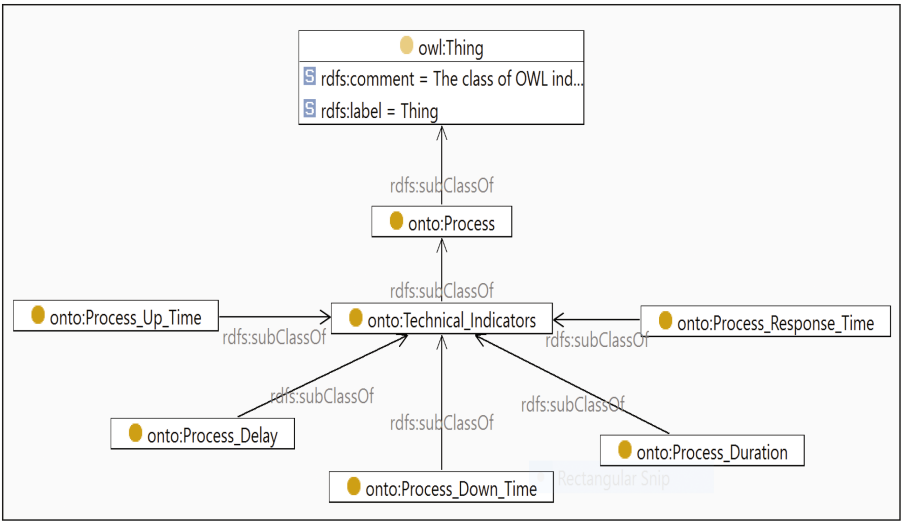

Fig. 5. Ontology of Performance at Orchestration Layer.

binding-reliable-messaging, binding-security (authentication, authorization, and encryption) and binding-bandwidth.

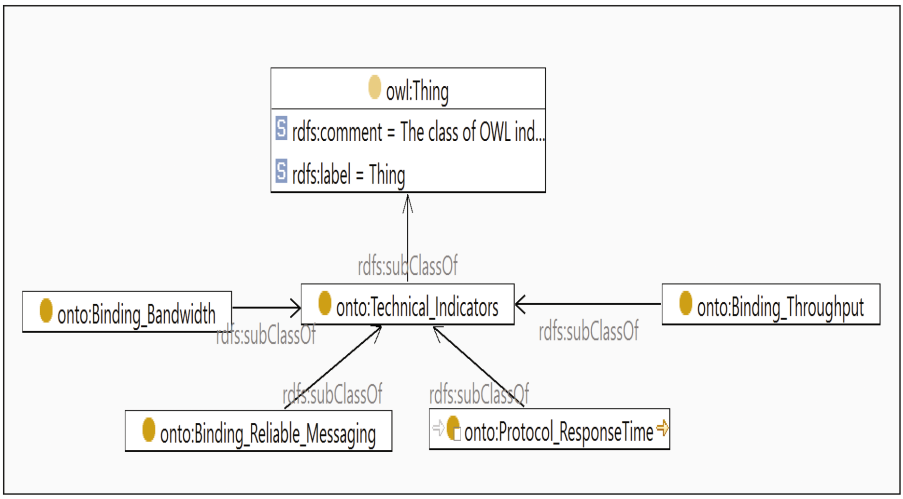

Fig. 6. Ontology of Performance at Integration Layer.

5 Conclusion

In this paper, we propose a framework to identify the performance objectives along the service oriented architecture layers. The proposed research work accelerates the analysis of existing service networks in order to validate service reuse capabilities. First of all we design our service performance ontology SPOnt to show the relationships of all the concepts. Then we identify the performance objectives at the service layer, orchestration layer, resource layer and binding level. Then we use performance objectives of these aforementioned layers to provide decision making capabilities. We will enhance our work to cover the recommendation based on the decision rules for service reuse and service composition problem. We will implement our work by using real time case study.

References

1. Gottschalk, K., Graham, S., Kreger, H., Snell, J.: Introduction to web services architecture. IBM Syst. J. **41**(2), 170–177 (2002)
2. Extensible Markup Language (XML) 1.1 (2nd edn.) (2006). World Wide Web Consortium. http://www.w3.org/TR/xml11/
3. Simple Object Access Protocol (SOAP) 1.2, Part 0, Primer (2007). World Wide Web Consortium. http://www.w3.org/TR/soap12-part0/
4. Web Services Description Language (WSDL) 2.0, part 1: Core Language (2007). World Wide Web Consortium. http://www.w3c.org/TR/wsdl20/
5. https://www.opengroup.org/soa/source-book/soa_refarch/layers.htm
6. Tari, Z., Phan, A.K.A., Jayasinghe, M., Abhaya, V.G.: Benchmarking soap binding. In: On the Performance of Web Services, pp. 35–58. Springer (2011)

7. Documents Associated with Business Process Model and Notation (BPMN) Version 2.0 Release date. January 2011. http://www.omg.org/spec/BPMN/2.0/PDF
8. Fahad, M., Qadir, M.A.: A Framework for Ontology Evaluation. ICCS Suppl. **354**, 149–158 (2008)
9. http://jakarta.apache.org/jmeter/
10. http://www.soapui.org/
11. https://eclipse.org/tptp/
12. https://marketplace.eclipse.org/content/jprobe
13. http://www.oracle.com/technetwork/middleware/bam/downloads/index.html
14. http://wso2.com/products/application-server/
15. INTERMON project. http://www.intermon.org/
16. Joint Research Activity 4, Enabling Grids for E-SciencE (EGEE) project. http://egeejra4.web.cern.ch/EGEE-JRA4/
17. MONitoring Agents using a Large Integrated Services Architecture (MonALISA). California Institute of Technology. http://monalisa.caltech.edu/
18. PlanetLab project. http://www.planet-lab.org/
19. Hanemann, A., Boote, J., Boyd, E., Durand, J., Kudarimoti, L., Łapacz, R., Swany, D., Trocha, S., & Zurawski, J. PerfSONAR: a service oriented architecture for multidomain network monitoring. In: Proceedings of 3rd International Conference Service Oriented Computing (ICSOC 2005). Amsterdam, The Netherlands. doi:10.1007/11596141_19
20. Bourgeau, T., Augé, J., Friedman, T.: TopHat: supporting experiments through measurement infrastructure federation. In: Proceedings of the 6th International ICST Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TridentCom 2010)
21. Zarghami, A., Sapkota, B., Eslami, M. Z., van Sinderen, M.: Decision as a service: separating decision-making from application process logic. In: EDOC, pp. 103–112. IEEE (2012)
22. Boumahdi, F., Chalal, R., Guendouz, A., Gasmia, K.: SOA+d: a new way to design the decision in SOA-based on the new standard Decision Model and Notation (DMN). SOCA **10** (1), 35–53 (2016). http://link.springer.com/search?query=Boumahdi&search-within=Journal&facet-journal-id=11761
23. Moraes, P., Sampaio, L., Monteiro, J., Portnoi, M.: Mononto: A domain ontology for network monitoring and recommendation for advanced internet applications users. In: Network Operations and Management Symposium Workshops. IEEE (2008)
24. Benaboud, R., Maamri, R., Sahnoun, Z.: Semantic Web Service Discovery Basedon Agents and Ontologies. Int. J. Innov. Manage. Technol. **3**(4), 467–472 (2012)