A Taxonomy-driven Approach for Performance Measurement and Modelling in Service Oriented Architectures

Ernest Sithole[1], Sally McClean[1], Bryan Scotney[1], Gerard Parr[1],
Adrian Moore[1], and Dave Bustard[1]
Stephen Dawson[2]

[1] School of Computing and Information Engineering, Faculty of Computing and Engineering, University of Ulster at Coleraine,
Coleraine - BT52 1SA, Co. Londonderry, Northern Ireland, UK
{e.sithole, si.mcclean, bw.scotney, gp.parr, aa.moore, dw.bustard}@ulster.ac.uk

[2] SAP Research CEC Belfast, TEIC Building, University of Ulster,
Jordanstown – BT37 0QB, Newtownabbey, UK
{stephen.dawson}@sap.com

Abstract. Performance analysis in traditional distributed computing systems has always been inherently complex given the combination and interplay of behaviours associated with various operational components such as application routines, operating systems, communication protocols, the network infrastructure, and CPU and storage hardware nodes. In Service Oriented Architecture- (SOA)-based distributed systems, where functional capabilities are abstracted and presented as service packages, accurately determining performance patterns is even more challenging. The increased complexity arises from additional characteristics of the middleware-based service framework. In this paper we propose a conceptual approach for modelling SOA performance, which is based on understanding (a) the behaviours manifested in the SOA-based infrastructure and (b) the properties associated with the underlying physical resource landscape. The parameters used in developing the SOA performance models are derived from the service taxonomy that we also present. Finally, an initial conceptual method for service performance modelling is developed.

Keywords: Service taxonomy, performance, compositions.

1 Introduction

A dominant trend in distributed computing is the current migration of Information Technology (IT) systems towards the Service Oriented Architecture (SOA) design. The principal idea behind the adoption of the SOA paradigm is to define and present the functional capabilities of IT infrastructures to user environments as addressable and reusable services. As shown in Figure 1, the SOA-based IT implementation

approach can be summarised into a layered organisation. Such a functional structure permits simple, incremental and safe enhancements on the IT infrastructure. Furthermore the layered organisation of the functional elements bridges the semantic gap between corporate business goals and technical IT functions. As a result, the flexibility inherent in the standards-based SOA implementations enables the automation of enterprise processes and measurable performance of business tasks.

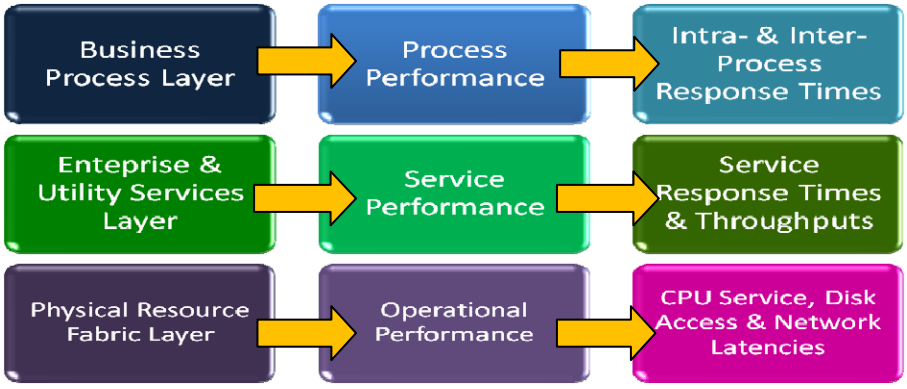

Fig. 1. Performance Components Associated with SOA Functional Layers.

The basic makeup of the SOA system implementation consists of the following functional layers shown in Figure 1: (a) Business Process Layer in which service compositions are defined (b) Enterprise Services Layer that provides specific service offerings for the needs of the enterprise domain as well as the virtualised Utility Services environment, in which the operations executed by physical resources are presented as accessible service capabilities and (c) the Physical Resource Fabric made up of distributed hardware and system platforms that perform the actual physical operations for service delivery.

Included in Figure 1 are the performance components associated with each of the layers of the SOA stack. Operational Performance is obtained from the runtime implementation of application routines as they execute in the hardware environment. Service Performance is derived from the lowest level of service provision, where the functionality of a single service is accessed through interfacing mechanisms. Process Performance is based on executions of integrated service packages that are brought together to achieve a concrete business result.

1.1 The Case for Service Taxonomy

The success of SOA implementations requires going beyond the technical breakthroughs (such as provision of standardised and flexible technologies) that have already been achieved in bringing together reusable functional components for service oriented IT systems. In order to plan for and review SOA implementations systematically, there is also a need for information-rich and structured formats that describe service attributes. Such descriptions should (a) permit the methodical

analysis of the SOA landscape's makeup and (b) enable systematic integrations of service components in the development of SOA solutions. According to the discussion presented in [1], the author argues for the presentation of service information in the form a taxonomy so that sound methodologies in formulating and evaluating SOA-based IT deployments can be supported. With sets of service information classified as a structured catalogue or taxonomy, SOA architects can have a useful tool in designing service-based IT solutions. Such a taxonomy format enables (a) consistency in the organisation of service categories, (b) improved service discoverability and (c) easier and wider federations of service domains leading to greater collaborations in the SOA cosmos.

While the taxonomies aimed at satisfying the above-mentioned requirements serve a useful purpose in planning for and evaluating SOA implementations, such classifications do not sufficiently address the complexities associated with performance analysis in distributed SOA systems. The functionality of the dispersed SOA frameworks is characterised by the interplay of multiple behaviours. These characteristics emanate from the physical resource fabric (made up of processing, communication, storage, application and system software components) and the behaviours associated with the individual service elements in the middleware-based service framework. Our proposals for a taxonomy set out to address the following aspects: (a) provision of service classifications for capturing key characteristics having a bearing on performance trends associated with both the service and physical resource landscapes and (b) presentation of performance-related characteristics so that the related functions of SLA/QoS guarantees, optimisations and fulfillment of performance goals are supported in SOA environments. Our approach of developing and using the service taxonomy in a performance-oriented way is in contrast to previous contributions, which have provided fairly general classifications that largely pay attention to service characteristics relating to functional capability.

2 Guidelines for Taxonomy and Key Service Classifications

Perhaps as the starting point in proposing our service taxonomy, it is worth considering other related research contributions on the subject. The discussion in [2] highlights the need for first taking into account the technical objectives that are common to most SOA implementations, and then apply those aims in developing service classifications. A common objective in most SOA solutions is the need for infrastructure designers to obtain a sound appreciation of the architecture the service landscape. Thus, for the service taxonomy to be useful in this regard, it should structure service properties in such a way that architecturally significant aspects of the SOA environment are easily conveyed. In this paper, we consider architectural aspects to be the collection of definitions and properties, which describe the *composition, organisation, coordination* and *interactions* of constituent service elements that are brought together to accomplish SOA-based IT solutions.

Another guiding objective which is becoming increasingly important to take into account when developing SOA-based IT systems is the need to design infrastructure deployments that meet specific performance targets. Therefore, the consideration of

how service categories can present information for supporting evaluation and planning of performance delivery over the SOA-based deployments is a further aspect to address in the choices of service classifications we adopt.

2.1 Principal Categories for Taxonomy

As the preceding discussion notes, considerable public discussion has been directed at the subject of service taxonomies, with most of the related work presenting different flavours of service classifications featuring functional capabilities [1][3][4][5]. In contrast, our taxonomy aims to provide a comprehensive set service of descriptions that capture essential dimensions of the SOA infrastructure. In turn, the dimensions of the taxonomy are used to support analysis and modelling of performance on SOA-based IT environments.

To present service characteristics in a performance-relevant way, we propose the adoption of the following as the main categories: (a) Service Functionality, (b) Relationships, (c) Interfacing and Runtime Properties, (d) Deployment and (e) Execution Strategies. It is worth pointing out that a substantial part of service classifications built into our taxonomy borrows from already existing classifications. To assist the readers appreciate the performance implications of the service categories that we present, full discussion on the subject is provided in the next section. The five major categories comprising our taxonomy are briefly considered from 2.2 to 2.5.

2.2 Classifying Service Functionality

Classifications of service functionality provide clear boundaries on the capability that individual services can and cannot offer. Such classification enables solution architects to determine the appropriate combinations of services required in a complementary fashion to meet the specific needs of SOA-enabled applications.

The discussion by Cohen in [5] provides two major categories of service classifications of functional capability; Business and Infrastructure Services. Business services are specific to the particular enterprise domain while Infrastructure services are common to all SOA-driven IT implementations. The Business functionality is expanded into 3 levels of service integration. The scope of integrations can be at Entity, Capability or Activity level. As shown in Figure 5, our taxonomy only factors the aspects of Business and Infrastructure operation, which we consider most important in conveying information about required functional features.

2.3 Classifying Service Deployments and Execution Approaches

The approaches taken in deploying and executing services provide further categories that we consider important to include in the taxonomy. As Figure 5 shows, Service Deployment determines in which part of the resource infrastructure service objects are installed to run (i.e. whether the intra- or extra-organizational arrangements can be made for accessing required services). The Service Execution choices determine the

approaches taken in the invocation of individual services that make up compound applications. The choice of marshalling strategy depends on the quantity of and dependencies between constituent services objects that are assembled into a SOA solution. Choreography-based techniques direct the execution of service routines through the operational logic residing inside the service components themselves. Orchestration approaches use external logic to fix the order and coordination of constituent service operations.

2.4 Classifying Service Interactions

As stated in 2.1 and also shown in Figure 5, we consider the interactions of services an important dimension that the service taxonomy must capture. The description of the interactions through service relationships summarises the respective roles that constituent services play in accomplishing SOA solutions, particularly in scenarios where composite services are used.

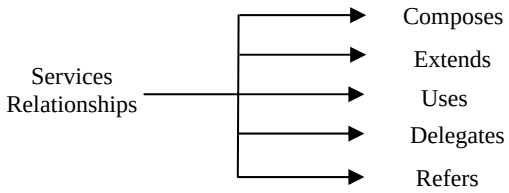

Fig. 2. Classifications of Service Relationships

Based on the set of relationships described in the OGSA document [8], the primary behaviours we adopt for our taxonomy are summarised as follows: (a) *Composes Relationship* encompassing situations where a service integrates a collection of underlying service capabilities in order to achieve a requested functionality, (b) *Extends Relationship* characterised by service entities that inherit and supplement the features of parent services, (c) *Uses Relationship* for service interactions whereby one service directs requests to other target services to handle, (d) *Delegates Relationship* for service responses that may optionally redirect requests to other services for execution and (e) *Refers Relationship* that describes responses involving prior consultations between associated services to establish or validate particular status attributes before the execution of received requests can proceed.

2.5 Classifying Service Runtime Properties

Each service element has attributes according to which its principal behaviours are characterised. The taxonomy according to [5] presents seven attributes for capturing key service behaviours. From that list our taxonomy adopts five service properties we consider as having strong bearing on performance at runtime; Interfacing Definitions, State Management, Transaction Handling, Error Handling, and Security Enforcement.

For detailed discussion on the impact which the selected runtime properties have on performance, reference can be made to 3.2

The *Interfacing* definitions provide protocol-based descriptions for exposing functions in a service. The *State Management* definitions determine how a service responds to messages having a bearing on the status of a running application. Provided by the *Transaction Handling* characteristic is a service's capability to receive, process, generate and transmit data in collaboration with other service objects that it has dependencies with. The *Error Handling* functions specify the corrective ability in a service to deal with operational inconsistencies of SOA-enabled applications. *Security Enforcement* determines the protection offered during service interactions.

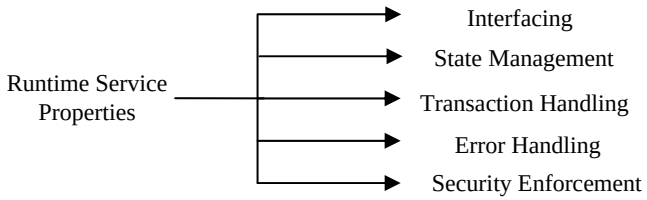

Fig. 3. Classification of Runtime Service Properties

3 Deriving Performance Components from the Service Taxonomy

As brought out in Section 1, this paper's focuses on developing a methodology for modelling and evaluating the performance of SOA implementations by considering how the service dimensions captured in the service taxonomy have an impact on performance. In the set of service classifications presented in the taxonomy, five broad categories are proposed as Figure 5 shows. From 3.1 to 3.3, we briefly consider the performance implications of the Functionality, Relationships, Deployment, Runtime Properties and Execution Strategies of service.

3.1 Service and Process Performance from Functionality and Relationships

The service taxonomy summarises the interactions of services through the service relationship category. The service relationships describe the respective roles that constituent services play in accomplishing SOA solutions, especially in scenarios involving use of multiple services. From the descriptions presented in 2.4, it can be appreciated that the nature of service relationships employed by SOA architects determines how services are joined up in accomplishing end-to-end process solutions. In turn, the integration of services as well as conditions governing their invocation can lengthen the overall response time of assembled business processes. In order to formally describe the phenomenon of service relationship in SOA performance models, Rud et al. in [6] [7] propose the use of correlation factors that denote the affinities between service parameters based on previous behavioural patterns. We

believe that the use of correlation coefficients has potential for describing the strength of service relationships involved in business process compositions.

3.2 Service Performance from Interfacing Definitions and Runtime Properties

Interfacing descriptions are a key component of Service Properties as presented in the taxonomy [5]. The descriptions specify the formatting of messages, protocol-based exchanges and validations so that users can access and use service capabilities. In terms of the actual sequence of service execution, interfacing determines how the procedures of service discovery, allocation and invocation are performed. Figure 4 shows service interfacing features, which can be HTTP/SOAP-, REST-, CORBA-, or RPC-based. Since service functionalities cannot be invoked without the initial exchange of SOA-based protocol messages, overheads in terms of time delays are inevitable for SOA-based IT implementations. These overheads directly impact on overall service response times, and also on throughput levels for workloads associated with high rates of service requests.

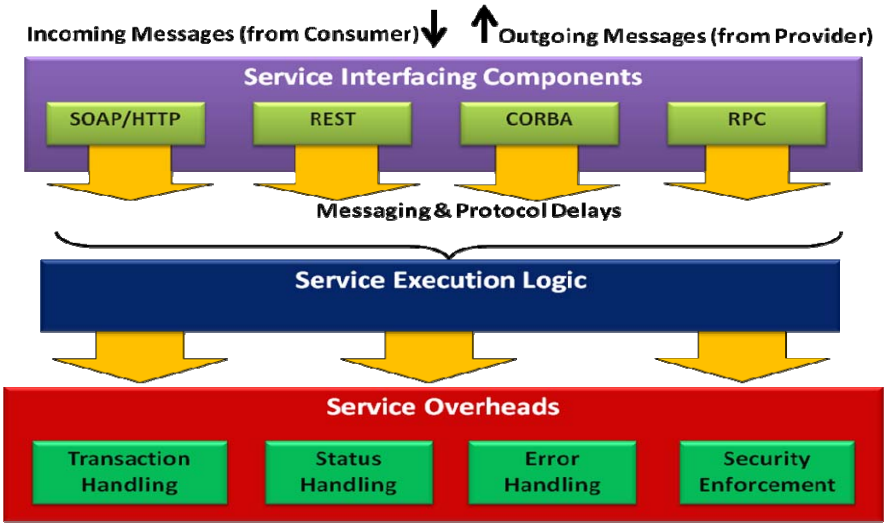

Fig. 4. Performance Components from Interfacing and Runtime Definitions

Besides the delays occurring prior to service execution, properties of state management, transaction handling, security enforcement and error correction contribute to additional overheads during runtime as Figure 4 shows.

3.3 Process Performance from Service Location and Execution Strategy

We have highlighted that physical deployments of individual services determine how readily they are accessed from user environments. For local deployments of services, performance evaluation would consider host system settings such as CPU speed, Caching Techniques, Page Fault levels and Disk access mechanisms. In most end-to-

end process integrations, significant increase in overall process response times is experienced when component services are scattered over a wide geographic area. For distributed SOA systems, additional overheads due to network latencies such as bandwidth, congestion and propagation delays need to be considered.

4 Definitions for Preliminary Model

In Figure 5, an all-encompassing description of the service taxonomy that summarises both the key dimensions of service characteristics and their impact on performance as discussed in Section 3 is shown. Using the set of service classifications shown in Figure 5, four important steps, which will provide a taxonomy-driven template for SOA performance modelling are derived.

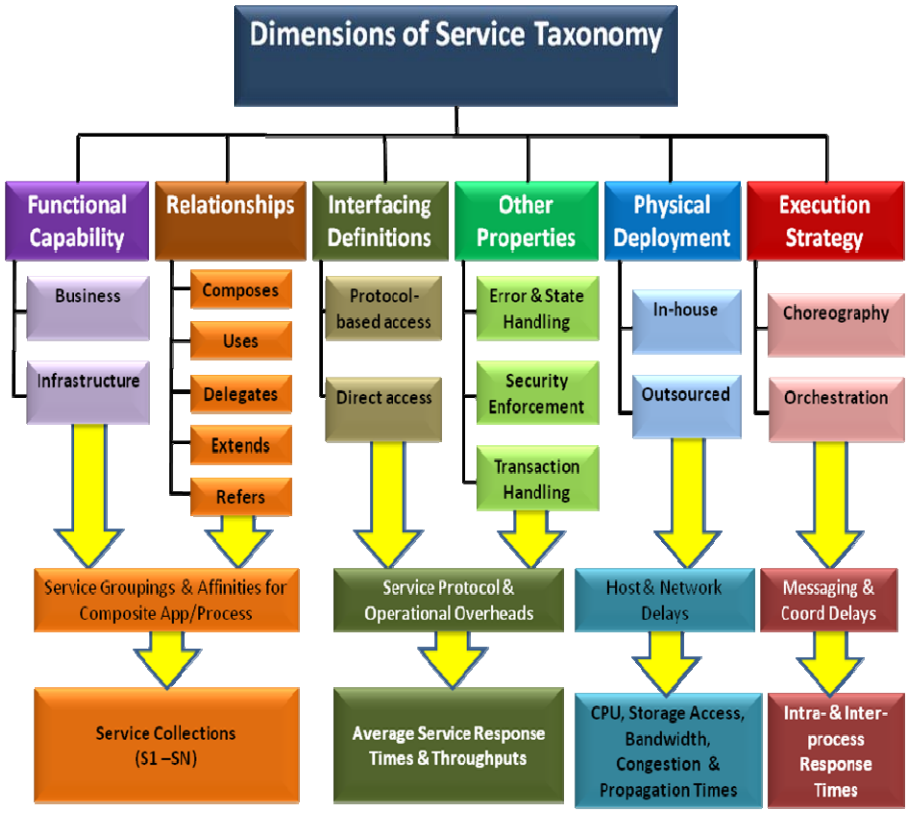

Fig. 5: Identifying Performance Components from Service Taxonomy

The taxonomy-based modelling approach is outlined as follows:

Stage 1: Based on the needs of the received service request, determine the required service groupings, S1 – SN, and their interactions based on existing functional capability and the choices of service relationships. In the real-world treatment of this

modelled stage, the considerations made would ensure that sufficient service-provision capacity is in place to fulfil the requirements of the accepted request.

Stage 2: From the choices of individual services brought together to provide a solution, we determine the overheads associated with protocol-based service accesses, $T_{Protocol}$, and operational overheads, T_{Secr}, $T_{Transac}$ and T_{Error}, during service execution.

Stage 3: Based on the physical deployment(s) of the individual services in use, we determine the hardware-based costs associated with host machines and the network infrastructure. For a local host system, service time, T_{CPU}, is a function of such settings as CPU speed, Caching and Virtual Memory techniques, and Storage Access mechanisms. For distributed SOA implementations, the network delays of bandwidth, propagation and congestion times (T_{BW}, T_{Prop} and T_{Cong}) also needs to be considered.

Stage 4: From the option that is chosen for the coordination of service executions, we establish the costs, $T_{Intraprocess}$ and $T_{Interprocess}$, associated with message exchanges both within and between constituent process and service elements.

4.1 An Example Taxonomy-based Performance Model

The initial model presented here is a simplified version of the design template we have proposed. Although Figure 6 shows parameters associated with a distributed implementation for completeness, we use a simplified construct based on standalone deployment i.e. generated requests access target services on the same machine.

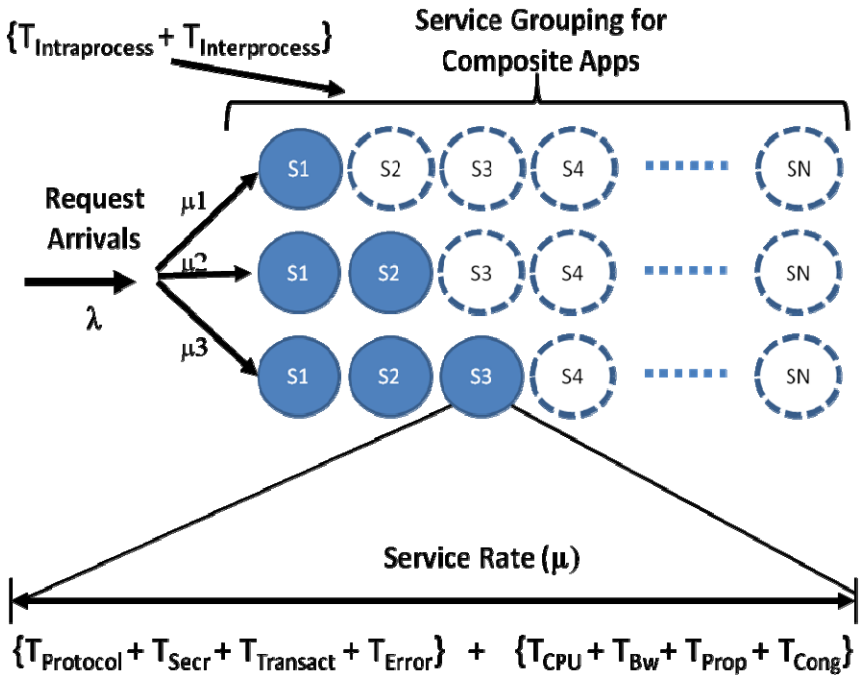

Fig. 6: Initial Model for Service Performance based on Taxonomy Descriptions

The collection of service elements, S1 – SN, in Figure 6, is derived from the application needs and dependencies between constituent services. The service execution time is a function the local machine's system settings. Because sequential operation of services is assumed, execution strategies for messaging and coordination of service functions are treated as embedded features inside each constituent service.

We now present the analytical description of the simplified model, where there are s services. Each service has exponential duration with rate μ and, with probability p_i, an application requires i services, for i=1,…,k. So the total execution time of an application which requires i services, denoted by X_i, is the convolution of i independently and identically distributed (i.i.d.) exponential random variables (r.v.s), each with rate μ. The distribution of of X_i is then special Erlang with probability density function (p.d.f.) given by:

$$f_i(x) = \frac{\mu^i x^{i-1} e^{-\mu x}}{(i-1)!} \quad x>0,$$

with corresponding mean i/μ and variance i/μ^2.

For any application, the p.d.f. of completion time is therefore:

$$f(x) = \sum_{i=1}^{s} \frac{p_i \mu^i x^{i-1}}{(i-1)!},$$

with corresponding mean

$$m = \sum_{i=1}^{s} \frac{p_i i}{\mu}.$$

The survival function $F(x)$ = Prob(X>x) is given by:

$$F(x) = \sum_{r=1}^{s} \sum_{i=1}^{r-1} \frac{p_r (\mu x)^i e^{-\mu x}}{i!},$$

where this function tells us how likely it is that a given application request exceeds a QoS threshold.

Table 1. Parameters for Preliminary Service Model Example.

Parameter	Description	Distribution
λ	Request Arrival Rate	Poisson (5)
$C_{Service}$	Number of Service instances	Uniform (1 - 10)
T_{CPU}	CPU time for Service operation	Exponential (0.5)

For the invocation of composite services, we model the instances of the internal services, $C_{Services}$, as a variable parameter that ranges between 1 and 10 component services, S1 – S6, according to the uniform distribution characteristic. We characterise the CPU execution time, $T_{CPU,}$ according to the exponential distribution

with mean of 0.5 seconds. In Table 1 we present the list of parameters that were defined for the preliminary model.

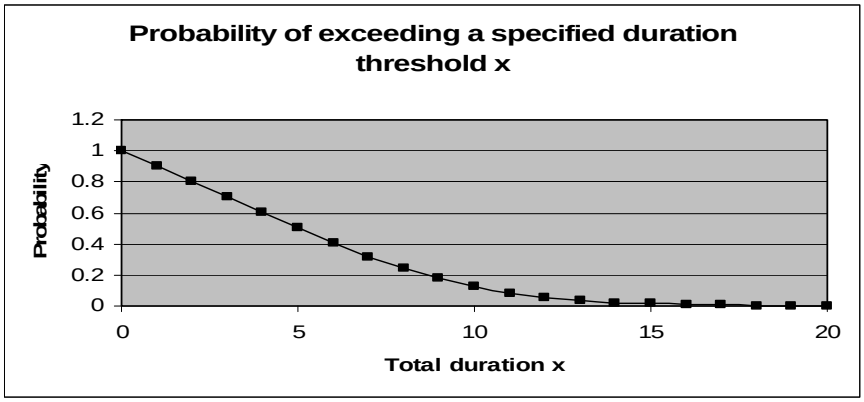

Fig. 7. Probability of exceeding a given duration threshold x

*Example*1: We consider the case when s=10, μ=1 and p_i =1/10 for i=1,…,10. In this case, m =5.5. In Figure 7 we illustrate the probability of exceeding a specified threshold x as a function of x. Thus, we can see that, for these data, if we have a QoS contract to execute 95% of requests in less that 10 time units, the current system will not be able to comply since the probability of exceeding 10 time units is 0.125, i.e. only 87.5% jobs will meet the QoS specification.

5 Conclusions and Future Directions

The discussion started by highlighting the complexity involved in analysing performance of distributed IT infrastructures, which results from the occurrence and interplay of multiple behaviours associated with the functional components of dispersed SOA frameworks.

We then proposed a modelling approach based on the classifications of important characteristics of the service and physical resource landscapes so that SOA performance is accurately characterised. Based on this taxonomy-driven approach, we presented a simple example model, which described service performance on a single machine. We use our taxonomy to inform the modelling of service performance by capturing those aspects of the service and physical resource domains that have direct bearing on SOA performance delivery. We believe that the modelling approach we have presented can provide accurate characterisation of performance patterns over SOA frameworks since it is based on a comprehensive set of service classifications that are relevant to performance.

In order to enhance the modelled features presented, further work will consider using the taxonomy for more complex scenarios on SOA performances modelling by giving detailed attention to hardware characteristics. We will also investigate the use of our approach to support the SLA-based modelling of (a) Brokering and

Optimisation through orchestration strategies that will explore parallel strategies in conjunction with service relationships and execution strategies, and (b) Autonomic capabilities or real-time intelligence in responding to unexpected operational changes. We also plan to carry out validations of our example models with benchmarked results from SOA implementations on physical test bed infrastructures and analyse the effects of loading on the distributed infrastructure in terms of likelihood, duration, and levels of congestion. To determine the effectiveness of our approach of taxonomy-based treatment of SOA performance, we intend to compare our results with other outputs from related work on SOA modelling and simulation frameworks [9] [10] [13] as well as SLA enforcement [11] [12].

Acknowledgments. We would like to express appreciation for contributions by researchers at SAP Belfast Laboratory with whom we had insightful discussions on the various topics presented in this paper. This research was jointly supported by INVEST Northern Ireland and SAP.

References

1. Morgenthal, J.P.: Taking Web Services To the Next Level: Taxonomy Needed. http://www.ebizq.net/topics (2003)
2. Biske, T.: Outside the Box: SOA, BPM, and Other Strategic IT Initiatives - Service Taxonomy. http://www.biske.com (2006)
3. Roth, B.: Service Oriented Architecture Best Practices: Meeting Your Goals Takes Effort: http://www.sys-con.com (2005)
4. Hill, I.: Service Taxonomy and Ontologies Deliver Success to Enterprise SOA: http://www.sys-con.com (2006)
5. Cohen, S.: Ontology and Taxonomy of Services in SOA: In Microsoft Architect Journal: (2007)
6. Rud, D., Schmietendorf, A., Dumke, R.: Resource Metrics for Service-Oriented Infrastructures: In Proc. Workshop on Software Engineering Methods for Service Oriented Architecture (SEMSOA), Hanover, Germany (2007)
7. Rud, D., Kunz, M., Schmietendorf, A., Dumke, R.: Performance Analysis in WS-BPEL-based Infrastructures: In Proc. 23rd Annual UK Performance Engineering Workshop (UKPEW), Lancashire, UK (2007)
8. Forster, I., Kishimoto, H., Savva, A., Berry, D., Djaoui, A., Grimshaw, A., Horn, B., Maciel, F., Siebenlist, F., Subramaniam, R., Treadwell, J., Von Reich, J.: Open Grid Services Architecture (2006)
9. Tsai, W.T., Fan, C., Chen, Y., Paul, R.: (2006) DDSOS: A Dynamic Distributed Service Oriented Simulation Framework: In Proc. 39th Annual Simulation Symposium (ANSS'06)
10. Gehlot, V., Way, T., Beck, R., DePasquale, P.: Model Driven Development of Service Oriented Architecture (SOA) Using Colored Petri Nets: First Workshop on Quality in Modelling (2006)
11. Ameller, D., Franch, X.: Service Level Agreement Monitor (SALMonitor): In Proc. 7th International Conference on Composition-Based Software Systems (ICCBSS) (2008)
12. Nan, Z., Qiu, X., Meng, L.: A SLA-Based Service Process Management Approach for SOA: In Proc. Ist International Conference on Communication and Networking in China (ChinaComm) (2006)
13. Standard ML of New Jersey.: What is SML?: http://www.smlnj.org/sml.html (2005)