2010 International Conference on P2P, Parallel, Grid, Cloud and Internet Computing

A Method to Increase Services Availability on Web-based Inter-Enterprise Service Delivery Platforms

Natalia Kryvinska, Christine Strauss
Department of eBusiness, School of Business, Economics and Statistics, University of Vienna
Bruenner Strasse 72, A-1210 Vienna, Austria
natalia.kryvinska@univie.ac.at, christine.strauss@univie.ac.at

Peter Zinterhof
Department of Computer Sciences
University of Salzburg
Jakob-Haringer-Str. 2, 5020 Salzburg, Austria
peter.zinterhof@sbg.ac.at

***Abstract*— With wide deployment of distributed inter-enterprise Service Delivery Platforms (SDPs) by the majority of enterprises, there is an urgent need to understand and solve service traffic issues of this evolving architectures. Thus, this paper discusses a mathematical model for the performance analysis of services availability at the inter-enterprise SDPs. The SDP is presented as an analytical model with the total number of customers fixed, thus forming a closed queuing network. The SDP' distributed architecture is modeled as a finite source queuing model. The expected response time for that model is analyzed and computed. The numerical results and the corresponding curves are provided. And, related to open questions, future work is summarized.**

Keywords- Service Delivery Platform (SDP), Services Availability, Enterprise Communication and Computing Architecture, Operations Support System (OSS), Business Support System (BSS), Service-oriented Architecture (SOA).

I. INTRODUCTION

The emergence of the virtual enterprises and inter-enterprise virtual platforms is the opportunity to enable productivity gains as well as flexibility and responsiveness to customer and market dynamics that enterprises need to be competitive in today's environment. But, to take advantage of this opportunity and to succeed in this new environment, enterprises need to create service delivery and communication strategies that establish tighter connections among their employees as well as with partners and suppliers. Central to this focus is the service architecture that powers the enterprise interactions with customers, between enterprises on this platform, as well as the processes for delivering value to customers and shareholders.

To enable enterprises to implement business strategies that are truly driven by a customer focus, it is required:

- create an instant and seamless connection across enterprises: linking people, processes, systems and networks so the customers are better served;
- provide seamless access to critical communications and business information to facilitate better, faster decisions and enable a more competitive enterprise;
- deliver personalized services needed to build long-term customer relationships.

Today's business environment can be seen as a logical effect of eBusiness and advanced integrated networks, including the Internet, that have transformed business processes. The new "Net" and its applications deployment models have revolutionized the way businesses interact, collaborate and transact business with customers, suppliers, partners, employees and shareholders.

Thus, the purpose of our work is to development a model enabling enterprises to attach their customers more tightly by an effective service delivery enabling business processes at every step of the way. This communications enablement not only enables greater synergy and velocity among vendors, suppliers and partners within the value creation and delivery chain of the business, but also, creates responsive and seamless customer service [1 - 7].

II. SDP FUNCTIONS, COMPONENT TECHNOLOGIES, AND PERFORMANCE FROM THE PERSPECTIVE OF INTER-ENTERPRISES SERVICES SUPPLYING

The SDP (Service Delivery Platform) bridges distributed inter- and intra-enterprise IT environments over communications networks to streamline new-services deployment and delivery. By combining technologies that deliver services to end users, SDPs facilitate communication between OSS/BSS (Operations Support System/Business Support System), applications spanning heterogeneous computing platforms as well as interfaces with physical network elements.

Thus, an effective SDP must handle high-volume traffic loads with carrier-grade reliability, and support a dynamic mix of service offerings to a growing subscribers' base over constantly changing network configurations. SDPs need to include powerful quality assurance and performance monitoring tools for quick rollout and high quality of new services, fulfillment of partner SLAs (Service Level Agreements), problems preemption and rapid decision for both IT and Business Operations departments [8 - 12].

978-0-7695-4237-9/10 $26.00 © 2010 IEEE
DOI 10.1109/3PGCIC.2010.30

A. Contemporary Technologies to Support an Effective Services Creation and Delivery

The Web 2.0 and Service-oriented Architecture (SOA) are on the top of discussed now issues among IT architects and business executives. Both technologies are on the edge of an exponential growth over the next few years, due to their flexibility, cost effectiveness, and ease of integration. Each technology creates highly distributed composite applications (e.g., other words - mash-ups) that connect components or subsystems to form higher-level functional systems or target applications, and meet the following requirements:

- robust reliability with minimized latency and high availability;
- multiple layers of security to protect against general and XML-specific attacks;
- off-load of resource-intensive functions;
- XML acceleration for faster and more efficient performance;
- consistent high quality services;
- highly productive, innovative composite applications that combine business applications with communication and collaboration services.

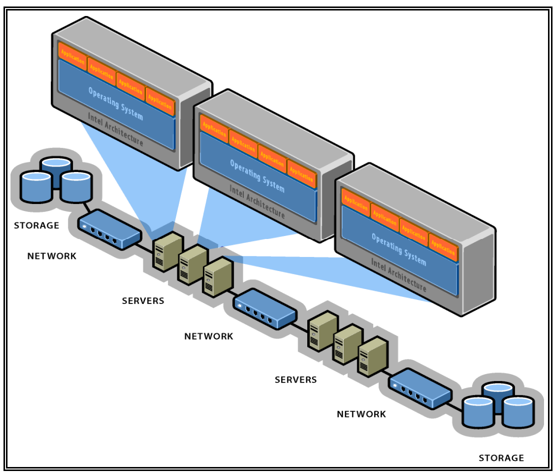

Figure 1. Complex, networked IT Architecture [13].

The growing challenge for enterprise IT architects is that because the composite applications are highly distributed, interactions between components may require several traversals across various areas of the network, each increasing the possibility of inconsistent performance or security problems. And, in turn, it decreases the availability of business services or processes built upon these composite applications. This issue becomes even more critical when Web 2.0 technologies are applied in order to leverage resources outside of the enterprise domains, using external networks and the Internet. To assure that all elements of broaden composite applications operate quickly, efficiently, and securely, a pervasive, reliable networked architecture is required. Besides, it is also important to understand the additional role the underlying infrastructure plays in Web 2.0 and SOA applications [14 - 18, 20].

However, the reason why enterprises have not yet applied high-availability distributed SDPs to their inter- and intra-infrastructures - is due to three primary challenges (Fig. 1):

1. costs to implement additional hardware can be quite excessive, also including expenses associated with additional tools and training;
2. complexity of developing and managing SDP solutions may be overwhelming for some organizations, including lock-step hardware and software upgrades;
3. reliability can be questionable due to limited testing possibilities of the complex solutions deployed [13, 15 - 18].

B. Role of SOA, Web 2.0, and Composite Applications in an Efficiency of the Inter-Enterprise SDPs

The Web 2.0 mashups, e.g., web applications that combine data from more than one source into a single integrated tool, and the SOA are the most illustrative models of what today are generically known as composite applications. Composite applications (e.g., building blocks or milestones of inter-enterprise SDPs) are application systems that are fundamentally enabled by network connectivity. They are composed of loosely coupled subsystems to form a higher-level functional system or target application. These subsystems can be data sources or services that perform a particular function, accepting input from and providing output to the target application. Composite applications can provide tremendous flexibility and, properly designed, offer high levels of business agility and productivity due to their ability to be reconfigured relatively quickly. The underlying model of composite design is similar in both Web 2.0 and SOA applications (Fig. 2).

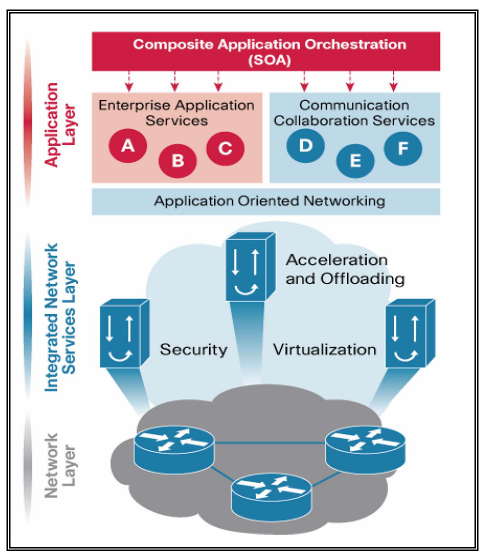

Figure 2. SDP model built upon SOA, Web 2.0 and other modern technologies [14].

Besides, the Web 2.0 is an evolutionary phenomenon that can be viewed from different perspectives. The user perspective encompasses a powerful trend toward user empowerment: Web 2.0 environments are greatly enriched by the simple premise that users should also be allowed to be content providers. The most influential examples of the synergies sets in this approach include wikis (Wikipedia), popular blogs (Engadget), photo-sharing sites (Flickr), video-sharing sites (YouTube), and social networks (MySpace, Facebook). The two main elements of this prosperous delivery environment are the concept of software as a service (SaaS) and mashup applications. SaaS allows the Web browser to challenge traditional desktop software when it comes to application delivery. Mashup applications unite data from different sources using open, intuitive protocols such as Extensible Markup Language (XML) and Representational State Transfer (REST) to create a contextually relevant presentation. By presenting data in innovative ways, mashups can significantly boost productivity, breaking down artificial barriers in data interpretation. A well known example of a mashup is to combine data from a Web application such as Google Maps that uses a REST interface, an architectural model for designing the easy flow of information via the Web.

On the other hand, SOA is a design and architecture paradigm centered on the creation of component "services" that can be combined to create business application systems. SOA is an architectural philosophy that does not specifically require or align itself with any particular technology set. It is focused on providing a tighter affiliation between business process and IT architecture in a modular fashion, with the goals of providing business agility, flexibility, and cost-effectiveness in long-term use. The SOA service components exhibit some typical core characteristics that deliver on the promise of flexibility, ease of integration, and cost benefits. They are:

- loosely coupled using defined interfaces;
- internal functions, structure, and states are completely internalized and irrelevant to other components in the system;
- can be combined and recombined as needed;
- discoverable by other existing or new components or systems within the architecture;
- amenable to service agreements, e.g., capable of providing and adhering to publishable service definitions that outline functional capabilities, interfaces, inputs, and outputs.

The general idea of service orientation is to decompose functional processes into modular services or sub-processes that can be served by IT systems to optimally support higher-level business processes. The web services protocols such as XML and Simple Object Access Protocol (SOAP) currently serve as the standard technology set for SOA [14 - 17, 19, 20].

In the next sections we examine and model how services availability on enterprise SDPs can be increased when applying mathematical methods in combination with different available technologies.

C. *Underlying Virtualization, Grid, and Consolidating Technologies to Enable High Services Availabilities on SDP*

The enterprises with large-scale IT infrastructures are facing a double-edged challenge. The financial pressures exerted on IT budgets by the never-ending increase in demand for storage and compliance requirements, along with the ever-present need to provide resilient business continuity solutions.

The advent of SOA has led to an unpredictability of demand. The assumption from end users, more so now than ever, and the applications demanded to be available 24x7, 365 days a year. This need for trust in systems is an essential part of expectations from customers, partners and employees.

The benefits of virtualization in being able to reduce costs for large-scale organizations are undeniable. However, while server virtualization has brought major benefits, it can also introduce potential vulnerabilities. In a physical server environment, loss of a single server has significantly less impact than in the virtual world where, workload dependant, the consolidation ratio of virtual machines running on a single physical server could be in the 10-15x range. A physical server failure can affect all of the virtual machines and applications running on that piece of hardware. Similarly failure of the virtualization layer itself impacts all running virtual environments. The complexity of this scenario grows as organizations standardize on server virtualization and deploy tier one applications in a virtual server environment. In short virtualization, while hugely effective in what it does, is not enough on its own to provide safeguards against unplanned downtime. Furthermore, while server virtualization can address consolidation at the server level, it can be found desirable at the level of storage, data and applications.

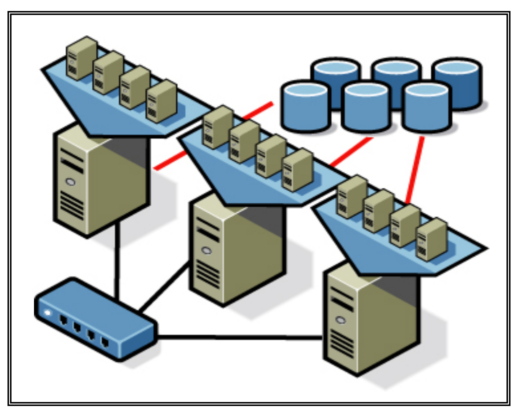

Figure 3. An example of a combination of server virtualization with other technologies shaping SDP [13].

Some enterprises may not be in a position to deploy a grid infrastructure. The reasons for this may be one of enterprise size, footprint size, IT policy, outsourcing, lack of budget. In these circumstances it is generally recognized as good practice for applications with non-intensive workloads to use server virtualization in order to maximize consolidation.

However, where maximizing consolidation, availability and agility are of paramount importance, a combination of server virtualization and grid-based solutions are the best way to maximize the benefits of consolidation, availability and agility. Working in tandem, they can ensure enhanced server virtualization, the ability to dynamically scale within and across nodes, and the dynamic resizing of virtual nodes.

Compared to other models of computing, IT systems designed and implemented in the grid style deliver a higher quality of service, at a lower cost, with greater flexibility. Higher quality of service results from having no single point of failure, a powerful security infrastructure, and centralized, policy-driven management.

Lower costs, meanwhile, derive from increasing the utilization of resources and dramatically reducing management and maintenance costs. Rather than dedicating a stack of software and hardware to a specific task, all resources are pooled and allocated on demand, which eliminates underutilized capacity and redundant capabilities. Grid computing also enables the use of smaller individual hardware components, which reduces the cost of each individual component and provides more flexibility to devote resources in accordance with changing needs.

The progressive enterprises have to implement a combination of server virtualization with grid computing to take advantage of database consolidation, running multiple, disparate workloads on the shared resources of the grid. The result is a more available, scalable, flexible and cost effective infrastructure resulting in better service levels to customers, users and partners (see an example in Fig. 3).

Latest grid-based solutions are now available that also offer all the benefits of server virtualization to single-instance databases on a physical hardware infrastructure. Many databases can be consolidated into a single cluster with minimal overhead while providing the high availability benefits of failover protection, online rolling patch application, as well as rolling upgrades for the operating system.

With these next-generation grid-based solutions, there are no limits to server scalability and if applications grow to require more resources than a single node can supply, they can be easily upgraded online. If the node becomes overloaded, users can migrate the instance to another node in the cluster using an online migration utility with no downtime for application users [21 - 25].

III. THE THEORETICAL MODEL OF SDP PERFORMANCE

The examination and improvements-modeling of system performance issues are essential tools in the development and engineering processes that can be used at all stages of the lifecycle of business services. Simple, approximate models have a high value in the early stages to uncover major performance problems, which affect the design of the architecture before the cost of rectification is too high. The design tools support rapid prototyping, allowing users to go through the three important stages: predict, design, and comparison. The questions of the development of new modeling methods for rapid analysis, and some others, like new performance standards, and closer connections between performance analysis and service design are the most interesting for the future wide deployment of business services (Fig. 4) [18, 27].

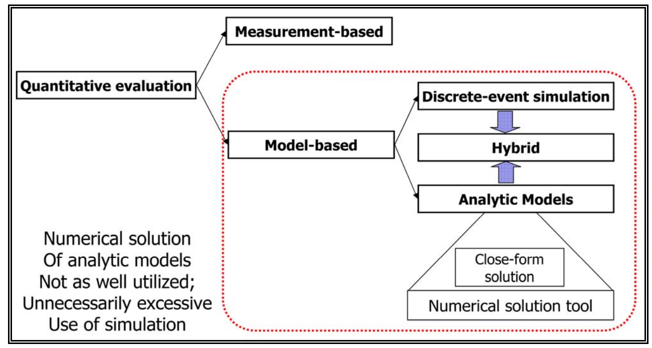

Figure 4. Theoretical Modeling Procedure [26].

Therefore, in order to model the SDP behavior, we consider it as a network of queues (M/M/2/K/K model) where the total number of customers (e.g., enterprise servers) is fixed and limited since no customers are allowed to arrive or depart. This network is called closed network, which can be analyzed using Markov chains. And, the steady-state occupancy distribution has a product form under assumptions similar to those used for open networks.

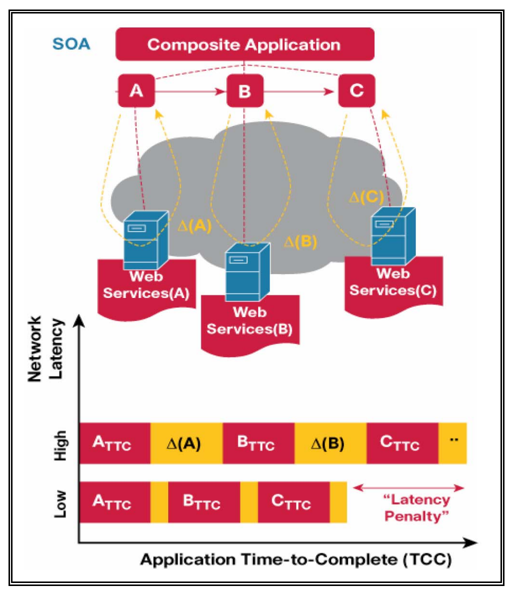

Figure 5. Large scale, distributed SDP [14].

Traditional services were provided by the service logic and data resident within the local machine. The capacity for these services is very much determined by the architecture and component capacities within the service node. The SDP has a distributed architecture in which service logic is executed cooperatively by different network elements that can be geographically dispersed (Fig. 5). An M/M/2/K/K, also known as the machine repair model or the cyclic queue model. In this context, there are K jobs cycling in a system consisting of K terminals or, in our case enterprise sub-servers, and two central Servers with a work queues. A job (e.g., request for a business service) is sent from a user/workstation/sub-server to the Server after an exponentially distributed "think time" and after being processed by the Server the job enters another think phase at a user side. The input and output messages of a transaction are treated as a single composite service. Also, the "think" time and the Server processing time are considered as an average "operating" time. In general, an M/M/m/K/K model is presented a model of a system with K users and m parallel servers. There are at least as many users as servers. If $K < m$, then m - K servers are never used and may be discarded. The user think times are distributed exponentially with parameter λ. Service times at all Servers are distributed exponentially with parameter μ. The system is in state j (j = 0, 1,…, K) if j users are waiting for their requests to be completed and K - j users are thinking. The instantaneous transition rate from state j to state j + 1 is equal to [27 - 33]:

$$\lambda_j = (K-j)\lambda, \qquad j = 0,1,...,K-1; \tag{1}$$

since each of the thinking users submits requests at rate λ. The rates from state j to state j-1 depend on whether the number of requests is less than the number of Servers, in a similar way to M/M/m/K/K model:

$$\mu_j = \begin{cases} j\mu & \text{for} \quad j = 1,2,...,m-1 \\ m\mu & \text{for} \quad j = m, m+1, K \end{cases} \tag{2}$$

The balance and normalizing equations yield

$$\begin{aligned} p_j &= \frac{K!}{(K-j)!\,j!}\rho^j p_0; \qquad j = 0,1,...,m-1 \\ p_j &= \frac{K!}{(K-j)!\,m!\,m^{j-n}}\rho^j p_0; \quad j = m, m+1,...,K \end{aligned} \tag{3}$$

with ρ – server utilization and p_0 given by:

$$p_0 = \left[\sum_{j=0}^{m-1} \frac{K!}{(K-j)!\,j!}\rho^j + \sum_{j=m}^{K} \frac{K!}{(K-j)!\,m!\,m^{j-m}}\rho^j\right]^{-1} \tag{4}$$

The throughput, T, can be obtained either as the average number of requests completions, or as the average number of requests submissions, per unit time. The former approach requires the average number of busy Servers, r:

$$r = \sum_{j=1}^{m-1} j p_j + m\sum_{j=m}^{K} p_j \tag{5}$$

The expression for the throughput is then $T = r\mu$. Alternatively, we could find the average number of requests in service or in the queue, L:

$$L = \sum_{j=1}^{K} j p_j \tag{6}$$

Then the average number of thinking users is K-L. Since each of them submits requests at rate λ, the throughput is equal to $T = (K\text{-}L)\lambda$.

In the two special cases when m = 1 and m = K, the expressions have a simpler form. If there is a single Server, the steady-state probabilities are:

$$p_j = \frac{\rho^j}{(K-j)!}\left[\sum_{i=0}^{K}\frac{\rho^i}{(K-i)!}\right]^{-1}; \quad j = 0,1,...,K, \tag{7}$$

and, the throughput is equal to:

$$T = (1-p_0)\mu. \tag{8}$$

When the number of Servers is equal to the number of users, no request has to queue and users do not interfere with each other in any way. The steady-state distribution of the number of requests in service is binomial [27 - 33].

The average number of busy Servers is:

$$r = \frac{K\rho}{1+\rho}. \tag{9}$$

The throughput is given by [32, 33]:

$$T = \frac{K\lambda}{1+\rho} \tag{10}$$

A. Numerical Results of the Modeling

In Table 1 and Fig. 6 are given results of the probabilities for finding k (K = 1-20, m = 2) in system. In comparison with m = 1 Server system, the probability of k customers in system, when m = 2 Servers is decreasing, because service rate is growing.

TABLE I. THE PROBABILITY OF *K* CUSTOMERS IN SYSTEM (*K* = 2; *M* = 2)

λ (μ=1)	*p* (*K*=1)	*p* (*K*=2)	*p* (*K*=5)	*p* (*K*=10)	*p* (*K*=20)
0.001	0.000999	0.001996	0.0049751	0.00990055	0.019604158
0.1	0.0909091	0.165289	0.309296	0.367955	0.172604712
0.2	0.166667	0.277778	0.392711	0.240372	0.003741596
0.3	0.230769	0.35503	0.380255	0.102153	$4.7193*10^{-5}$
0.4	0.285714	0.408163	0.332182	0.0371103	$1.0565*10^{-6}$
0.5	0.333333	0.444444	0.276339	0.0133042	$4.1387*10^{-8}$
0.6	0.375	0.46875	0.22443	0.00500218	$2.5232*10^{-9}$
0.7	0.411765	0.484429	0.180347	0.00200906	$2.172*10^{-10}$
0.8	0.444444	0.493827	0.1445	0.00086305	$2.4549*10^{-11}$
0.9	0.473684	0.498615	0.115964	0.00039469	$3.4576*10^{-12}$
0.999	0.49975	0.5	0.0936573	0.0001923	$5.933*10^{-13}$

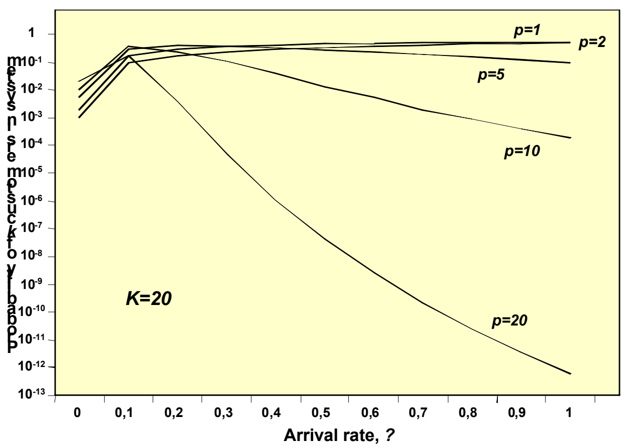

Figure 6. The probability of k customers in system (*K* = 20; *m* = 2)

The average number of customers in system (e.g., in the queue or service) *L* is presented in Table 2 (e.g., some numerical results) and plotted in Fig. 7.

TABLE II. AVERAGE NUMBER OF CUSTOMERS IN SYSTEM (M=2)

λ (μ=1)	*L* (*K*=1)	*L* (*K*=2)	*L* (*K*=5)	*L* (*K*=10)	*L* (*K*=20)
0.001	0.000999	0.004991	0.00797	0.0178706	0.520955
0.1	0.0909091	0.421487	0.565494	0.933449	9.09961
0.2	0.166667	0.722223	0.837156	1.07753	5.16433
0.3	0.230769	0.940829	0.966054	1.06821	3.86458
0.4	0.285714	1.10204	1.02606	1.06317	3.13407
0.5	0.333333	1.22222	1.05412	1.06742	2.73696
0.6	0.375	1.3125	1.06818	1.07318	2.48467
0.7	0.411765	1.38062	1.07654	1.07855	2.30245
0.8	0.444444	1.4321	1.08277	1.08363	2.16323
0.9	0.473684	1.47091	1.08826	1.08866	2.05468
0.999	0.49975	1.49975	1.09341	1.0936	1.96996

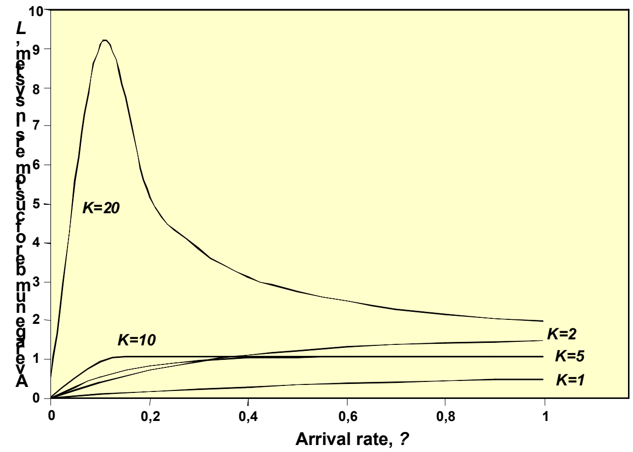

Figure 7. Average number of customers in system (*m* = 2)

In Table 3 and Fig. 8 we show the average time in system *T* (e.g., throughput), when *m* = 2 servers, and distribution of the service time is exponential.

TABLE III. THE AVERAGE TIME IN SYSTEM (E.G., THROUGHPUT) *T* (*M* = 2)

λ (μ=1)	*T* (*K*=1)	*T* (*K*=2)	*T* (*K*=5)	*T* (*K*=10)	*T* (*K*=20)
0.001	0.000999	0.001995	0.004992	0.00998213	0.019479
0.1	0.0909091	0.157851	0.443451	0.906655	1.09004
0.2	0.166667	0.255555	0.832569	1.78449	2.96713
0.3	0.230769	0.317751	1.21018	2.67954	4.84063
0.4	0.285714	0.359184	1.58958	3.57473	6.74637
0.5	0.333334	0.38889	1.97294	4.46629	8.63152
0.6	0.375	0.4125	2.35909	5.35609	10.5092
0.7	0.411765	0.433566	2.74642	6.24501	12.3883
0.8	0.444445	0.45432	3.13378	7.1331	14.2694
0.9	0.473684	0.476181	3.52056	8.02021	16.1508
0.999	0.49975	0.49975	3.90269	8.89749	18.012

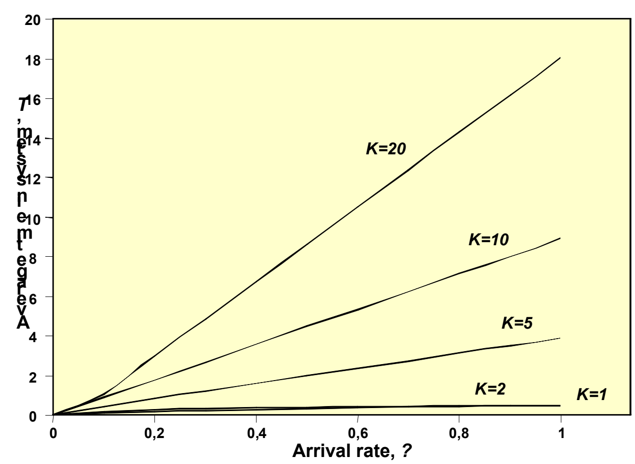

Figure 8. The average time in system (e.g., throughput) *T* (*m* = 2)

IV. CONCLUSIONS

We have examined in this paper how Web 2.0, Service-oriented Architecture along with composite applications and integrated network services, as well as underlying technologies (like servers virtualization in combination with grid computing and consolidating techniques) can help to create an optimal SDP foundation enabling to fulfill the potential as the next “disruptive force” of innovation in the enterprise. We have also discussed and modeled here how SDP can provide key categories of critically important business services, e.g., basic enablement, performance optimization with application enrichment, to ensure services availability, reliability, scalability, security, and predictable performance across diverse network/IT environments, and to support optimal alignment of composite applications with the business process they support [14 - 17, 19, 20].

Besides, the analysis and modeling of system performance issues are essential tools in the development and engineering processes that can be used at all stages of the lifecycle of business services. Simple, approximate models have a high value in the early stages to uncover major performance problems, which affect the design of the architecture before the cost of rectification is too high. The design tools support rapid prototyping, allowing users to go through the three important stages: predict, design, and comparison. The questions of the development of new modeling methods for rapid analysis, and some others, like new performance standards, and closer connections between performance analysis and service design are the most interesting for the future wide deployment of business services [18, 27].

REFERENCES

[1] White paper, “Avaya Communication Architecture”, LB1842, Avaya, March 2006. www.avaya.com/

[2] W. H. Davidow, M. S. Malone, The virtual corporation. HarperCollins Publishers, New York, 1992.

[3] H. P. M. Jägers, W. Jansen, G. C. A. Steenbakkers, “Characteristics of Virtual Organizations”, Proceedings of the VoNet workshop, Bern, April 27-28, 1998.

[4] N. Lippis, “Web Services & SOA Redefine IP Telephony Landscape”, The Lippis Report, Issue 51, February 2006.

[5] N. Venkatraman, J. C. Henderson, “Real strategies for virtual organizing”, Sloan Management Review, Fall 1998.

[6] E. Johnson, “Elusive Integration: Linking Sales and Operations Planning”, ASCET, Vol. 7, 2005.

[7] J. W. Palmer, C. Speier, “A typology of virtual organizations: an empirical study”, Proceedings of the Association for information systems 1997 Americas conference, Indianapolis, 15-17 August 1997.

[8] Solution Brief, “BEA Service Delivery Platform with Wily Performance Monitoring Enabler for Telecommunications”, PSB1433E0906-1A, BEA Systems, a subsidiary of Oracle Corporation, September 2006.

[9] S. Weerawarana, F. Curbera, F. Leymann, T. Storey, D. F. Ferguson, Web Services Platform Architecture: SOAP, WSDL, WS-Policy, WS-Addressing, WS-BPEL, WS-Reliable Messaging, and More. Prentice Hall, 2005.

[10] J. H. Cho, B.-F. Yu and J.-O. Lee, “Deploying Application Services Using Service Delivery Platform (SDP)”, LNCS 4773/2007, Proceedings of 10th Asia-Pacific Network Operations and Management Symposium (APNOMS 2007), Sapporo, Japan, October 10-12, 2007.

[11] R. Christian and H. Hanrahan, “Defining Generic Architectural Requirements for the Service Delivery Platform”, Proceedings of the Southern African Telecommunications and Networks Conference, SATNAC 2007, Mauritius, September 2007.

[12] I. Demydov, N. Kryvinska, C. Strauss, M. Klymash, I. Ivanochko, “Enterprise Distributed Service Platforms - an Approach to the Architecture and Topology”, Emerging Research and Projects Applications Symposium (ERPAS 2009), in conjunction with MoMM2009, 14-16 December 2009, Kuala Lumpur, Malaysia.

[13] On-line Presentation, VMware, 2004. available at: http://www.synegi.com/docs/vmware-synegi-presentation.ppt

[14] White Paper, “Cisco Service-Oriented Network Architecture: Support and Optimize SOA and Web 2.0 Applications”, Cisco Systems, C11-450967-00 1/08, 2008.

[15] N. Kryvinska, L. Auer, C. Strauss, “Managing an Increased Service Heterogeneity in Converged Enterprise Infrastructure with SOA”, Inderscience Publishers, International Journal of Web and Grid Services (IJWGS), vol. 4, no. 4, 2008.

[16] R. Christian and H.E. Hanrahan, “Defining a Service Delivery Platform Architecture using Generic IMS and SOA Concepts”, 11th International Conference on Intelligence in Service Delivery Networks (ICIN 2007), Bordeaux, October 2007.

[17] V. Radhakrishnan et al, “PIAF: An Application Framework for Unlocking IMS Engendered Network Capabilities,” in Programming Models for the IMS, 10th International Conference on Intelligence in Service Delivery Networks (ICIN), Bordeaux, France, May 2006.

[18] N. Kryvinska, Converged Network Service Architecture: A Platform for Integrated Services Delivery and Interworking. Electronic Business series, vol. 2, International Academic Publishers, Peter Lang Publishing Group, 2010.

[19] N. Kryvinska, L. Auer, C. Strauss, “The Place and Value of SOA in Building 2.0-Generation Enterprise Unified vs. Ubiquitous Communication and Collaboration Platform”, The Third International Conference on Mobile Ubiquitous Computing, Systems, Services, and Technologies (UBICOMM 2009), NexTech 2009, 11-16 October, Sliema, Malta, 2009.

[20] I. Demydov, N. Kryvinska, M. Klymash, “An Approach to the Flexible Information/Service Workflow Managing in Distributed Networked Architectures”, The International Workshop on Design, Optimization and Management of Heterogeneous Networked Systems (DOM-HetNetS’09), in conjunction with ICPP-2009, September 22-25, Vienna, Austria, 2009.

[21] White paper, “Making the Business Case for Data Centre Consolidation”, Oracle and IDG Global Solutions, 2009.
http://www.oracle.com/technology/tech/grid/index.html

[22] N. Kryvinska, L. Auer, C. Strauss, “SOI Framework for the Efficient Management of Complex Resource-Intensive Applications on Constrained Devices”, The International Workshop on Design, Optimization and Management of Heterogeneous Networked Systems (DOM-HetNetS’09), in conjunction with ICPP-2009, September 22nd -25th, Vienna, Austria, 2009.

[23] T. Thein, J. S. Park, “Availability analysis of application servers using software rejuvenation and virtualization”, Journal of Computer Science and Technology, v.24, n.2, March 2009.

[24] J. Alonso, L. Silva, A. Andrzejak, P. Silva, J. Torres, “High-available grid services through the use of virtualized clustering”, 8th IEEE/ACM International Conference on Grid Computing, 19-21 September, Austin, Texas, 2007.

[25] G. Goth, “Virtualization: Old Technology Offers Huge New Potential”, IEEE Distributed Systems Online, vol. 8, no. 2, pp. 3, Feb. 2007.

[26] K. Trivedi, “Dependability, Security and Survivability Models”, Keynote, 7th International Workshop on the Design of Reliable

Communication Networks (DRCN 2009), October 25-28, 2009 - Washington, D.C., USA.

[27] N. Kryvinska, H. M. Nguyen, “Large Intelligent Network Modeling Using M/M/2/K/K System”, IEEE 9th Asia Pacific Conference on Communications (APCC2003), 21-24 September, Penang, Malaysia, 2003.

[28] V. Cardellini, M. Colajanni, and P. S. Yu, “Dynamic Load Balancing on Web-server Systems”, IEEE Internet Computing, vol. 3, no.3, May/June 1999.

[29] P. J. Emstad, P.-O. Osland, “Dynamic load balancing in a two-server system”, in Proceedings of the 17th International Teletraffic Congress (ITC-17), Salvador da Bahia, Brazil, September 2001.

[30] R. Ackeley, A. Elvidge, T. Ingham, and J. Shepherdson, “Network Intelligence – Performance by Design”, IEICE Transactions on Communication, vol. E80-B, No.2, February 1997.

[31] E. Masuda, T. Mishima, N. Takaya, K. Nakai, and M. Hirano, “A Large-Capacity Service Control Node Architecture Using Multicasting Access to Decentralized Databases in the Advanced Intelligent Network”, IEICE Transactions on Communication, vol. E84-B, No.10, October 2001.

[32] L. Kleinrock, and R. Gail, Solution Manual for Queueing Systems. Volume II: Computer Applications. Technology Transfer Institute, 1986.

[33] I. Mitrani, Probabilistic modeling. Cambridge University Press, 1998.