Analysis and Modeling of Services Impacts on System Workload and Performance in Service-based Systems (SBS)

by

Billibaldo Martinez Aranda

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved January 2012 by the
Graduate Supervisory Committee:

Nong Ye, Chair
Tong Wu
Hessam Sarjoughian
Rong Pan

ARIZONA STATE UNIVERSITY

May 2012

ABSTRACT

In recent years, service oriented computing (SOC) has become a widely accepted paradigm for the development of distributed applications such as web services, grid computing and cloud computing systems. In service-based systems (SBS), multiple service requests with specific performance requirements make services compete for system resources. IT service providers need to allocate resources to services so the performance requirements of customers can be satisfied. Workload and performance models are required for efficient resource management and service performance assurance in SBS.

This dissertation develops two methods to understand and model the cause-effect relations of service-related activities with resources workload and service performance.

Part one presents an empirical method that requires the collection of system dynamics data and the application of statistical analyses. The results show that the method is capable to: 1) uncover the impacts of services on resource workload and service performance, 2) identify interaction effects of multiple services running concurrently, 3) gain insights about resource and performance tradeoffs of services, and 4) build service workload and performance models. In part two, the empirical method is used to investigate the impacts of services, security mechanisms and cyber attacks on resources workload and service performance. The information obtained is used to: 1) uncover interaction effects of services, security mechanisms and cyber attacks, 2) identify tradeoffs within

limits of system resources, and 3) develop general/specific strategies for system survivability.

Finally, part three presents a framework based on the usage profiles of services competing for resources and the resource-sharing schemes. The framework is used to: 1) uncover the impacts of service parameters (e.g. arrival distribution, execution time distribution, priority, workload intensity, scheduling algorithm) on workload and performance, and 2) build service workload and performance models at individual resources. The estimates obtained from service workload and performance models at individual resources can be aggregated to obtain overall estimates of services through multiple system resources.

The workload and performance models of services obtained through both methods can be used for the efficient resource management and service performance assurance in SBS.

ACKNOWLEDGMENTS

First of all I would like to thank my parents, siblings and family for their support and encouragement during my Ph.D. studies. Thanks to Rosa for her love, patience and support. My loved ones gave me the strength to achieve this goal.

I would like to thank my advisor, Dr. Nong Ye, for being a great teacher and guide during my Ph.D. studies. I have learned a lot from Dr. Ye not only in knowledge but also in attitude towards research, professional career and life. Also, I would like to thank Dr. Tong Wu, Dr. Rong Pan and Dr. Hessam Sarjoughian for being part of my dissertation committee and for their valuable insights and comments during this research.

I would like to express my gratitude to the Mexican Council of Science and Technology (CONACYT) and Arizona State University (ASU) for providing me with the financial support to accomplish this goal.

Finally, I would like to thank my friends and fellow Ph.D. students for making my time in the Ph.D. program more interesting and funnier. The experience of studying and doing research at ASU has been very valuable to me. I will remember it always.

TABLE OF CONTENTS

Page

LIST OF TABLES viii

LIST OF FIGURES xii

CHAPTER

INTRODUCTION ……..…… 1

2 IMPACTS OF SERVICES ON SYSTEM ACTIVITIES, RESOURCES WORKLOAD AND SERVICE PERFORMANCE……..... 6

2.1 Literature review 6

2.2 Shortcomings 9

2.3 Objectives 9

2.4 Methodology for data collection, analysis and modeling 10

2.4.1 Data Collection 10

2.4.2 Data Analysis 10

2.4.3 Data Modeling 13

2.5 Description of Experimental Scenarios 14

2.5.1 Service parameters in experimental scenarios 16

2.6 Results and Discussions 19

2.6.1 Impacts of VCS 19

2.6.2 Impacts of background network traffic 23

2.6.3 Impacts of security service of data encryption 25

CHAPTER Page

2.6.4 A summary of the impacts with VCS, security and traffic 27

2.6.5 Impacts of MDS 30

2.6.6 Cause-effect (ASQ) models of system dynamics 34

2.7 Conclusions 39

3 IMPACTS OF SERVICE, SECURITY AND CYBER ATTACKS AND THEIR IMPLICATIONS ON SYSTEM WORKLOAD, PERFORMANCE AND SURVIVABILITY 41

3.1 Literature review 41

3.2 Shortcomings 43

3.3 Objectives 44

3.4 Methodology of data collection and analysis 44

3.5 Description of experimental scenarios 44

3.5.1 VCS, data encryption and cyber attack scenarios 45

3.5.2 MDS, intrusion detection and cyber attack scenarios 47

3.5.3 System dynamics data collection 49

3.6 Results and discussions 51

3.6.1 System impacts of VCS, data encryption and cyber attacks 51

CHAPTER Page

3.6.2 System impacts of MDS, intrusion detection and cyber attacks....64

3.7 Conclusions....71

4 A FRAMEWORK TO ESTIMATE SERVICE WORKLOAD AND PERFORMANCE ……..……..…….. 74

4.1 Background....74

4.2 Previous Work....75

4.3 Shortcomings....77

4.4 Objectives....78

4.5 Framework Description....78

4.6 Models development....86

4.6.1 Processor (CPU) model....88

4.6.2 Processor model validation....89

4.6.3 Disk model....92

4.6.4 Disk model validation....95

4.7 Description of experiments for building workload and performance models....98

4.7.1 Processor experiments....98

4.7.2 Disk experiments....101

4.8 Results and Discussions....104

CHAPTER Page

4.8.1 Impacts of service parameters on processor workload and performance…….. 104

4.8.2 Workload and performance models for processor ..117

4.8.3 Impacts of service parameters on disk workload and performance.....120

4.8.4 Workload and performance models for disk 136

4.9 Conclusions.....138

5 CONCLUSIONS AND FUTURE WORK……. 143

REFERENCES 147

APPENDIX.....154

LIST OF TABLES

Table		Page
1.	Levels of the service parameters for VCS experiments……	17
2.	Levels of the service parameters for MDS experiments……...	18
3.	Consistent Impacts with VCS……...	21
4.	Consistent Impacts with background network traffic……...	23
5.	Consistent Impacts with security service of data encryption.…...	25
6.	Summary of system dynamics variables affected by more than one service parameter…….....……..	28
7.	Impacts of MDS parameters on system dynamics variables……	31
8.	Linear regression models of system dynamics for VCS-only……....37	
9.	Linear regression models of system dynamics for VCS & Traffic……	37
10.	Linear regression models of system dynamics for VCS & Security…….	38
11.	Linear regression models of system dynamics for VCS & Security & Traffic…..……	38
12.	Parameters levels for VCS, data encryption and cyber attacks……...	47
13.	Parameters levels for MDS, intrusion detection and cyber attacks……...	49

Table Page

14. System impacts of VCS, data encryption and cyber attacks……... 52

15. System impacts of MDS, intrusion detection and cyber attack……. 64

16. Description of active services during text editing condition……... 90

17. Services profiles containing arrival and execution time distributions…….. 90

18. P-values Mann-Whitney test for simulation runs under text editing……... 92

19. Default values for disk model parameters…… 94

20. Quantum Atlas 10K 9.1 GB disk basic characteristics……. 93

21. Service parameters (factors) for processor experiments……... 99

22. Service parameters (factors) for disk experiments……..… 102

23. Service parameters effects on CPU workload mean (μ) and standard deviation (σ)……. 109

24. Service parameters effects on Waiting Time mean (μ) and standard deviation(σ)…..…… 111

25. Service parameters effects on Operation Time mean (μ) and standard deviation (σ)……. 114

26. Service parameters effects on Completion Rate mean (μ) and standard deviation (σ)……. 116

Table Page

27. Regression models for service workload and performance at processor with RRP algorithm…… 119

28. Regression models for service workload and performance at processor with MLF algorithm……... 120

29. Service parameters effects on Disk Workload mean (μ) and standard deviation (σ)……. 126

30. Service parameters effects on Waiting Time mean (μ) and standard deviation (σ).…… 129

31. Service parameters effects on Operation Time mean (μ) and standard deviation (σ)……. 132

32. Service parameters effects on Completion Rate mean (μ) and standard deviation (σ)……. 135

33. Regression models for service workload and performance at disk with C-Look algorithm….……... 137

34. Regression models for service workload and performance at disk with SSTF algorithm……... 138

A1. Service profiles for processor experiments, 2-Services competition……. 155

A2. Service profiles for processor experiments, 5-Services competition……. 156

A3. Service profiles for processor experiments, 10-Services competition……. 157

Table Page

A4. Service profiles for disk experiments, 2-Services competition…… 161

A5. Service profiles for disk experiments, 5-Services competition…… 162

A6. Service profiles for disk experiments, 10-Services competition…… 163

LIST OF FIGURES

Figure Page

1. Different impacts of service parameters on system dynamics variables from the VCS with data encryption service and background network traffic……... 12
2. Computer and network set-up for VCS and MDS experiments... 15
3. Computer and network set-up up for services, security mechanism and cyber attacks experiments…….…….. 45
4. System impact characteristics of VCS…….. 62
5. System impact characteristics of data encryption……. 62
6. System impact characteristics of cyber attacks……. 63
7. System impact characteristics of motion detection……... 68
8. System impact characteristics of cyber attacks and intrusion detection…… 69
9. An abstract view of computer and network system components……... 79
10. Competition of services' instances at a single resource…… 81
11. Information related to resource idle and busy periods during the period of *NT*.…...…….. 82
12. Disk model structure (disk abstraction)…… 93
13. Disk workload comparison simulation model vs Lumb, et al. (2000) experiment……. 94

Figure Page

14. The effect of *T* on service workload and performance metrics at processor……. 105

15. Impacts of arrival distribution mean ($Arriv_{\mu}$) on CPU workload……. 107

16. Impacts of arrival distribution CV ($Arriv_{CV}$) on CPU workload……..... 108

17. Impacts of execution distribution mean (Ex_{μ}) on CPU workload…..... 108

18. Impacts of execution distribution CV (Ex_{CV}) on CPU workload……. 109

19. Impacts of workload by higher priority services (ρ_{HP}) on Waiting Time……... 110

20. Impacts of execution distribution mean (Ex_{μ}) on Waiting Time…… 110

21. Impacts of workload by higher priority services (ρ_{HP}) on Operation Time……...…… 112

22. Impacts of arrival distribution mean ($Arriv_{\mu}$) on Operation Time…… 112

23. Impacts of execution distribution mean (Ex_{μ}) on Operation Time…… 113

24. Impacts of execution distribution CV (Ex_{CV}) on Operation Time…… 113

Figure Page

25. Impacts of arrival distribution mean ($Arriv_{\mu}$) on Completion Rate……. 114

26. Impacts of arrival distribution CV ($Arriv_{CV}$) on Completion Rate……. 115

27. Impacts of execution distribution mean (Ex_{μ}) on Completion Rate……. 115

28. Deleted residuals vs fitted values CPU workload standard deviation (σ) model……. 118

29. The effect of *T* on service workload and performance metrics at disk…….. 121

30. Impacts of workload by other services (ρ_O) on service workload at disk…….. 123

31. Impacts of arrival distribution mean ($Arriv_{\mu}$) on Disk Workload……. 124

32. Impacts of arrival distribution CV ($Arriv_{CV}$) on Disk Workload……. 125

33. Impacts of block size (B) on Disk Workload…….. 125

34. Impacts of workload by other services (ρ_O) on Waiting Time… 127

35. Impacts of arrival distribution mean ($Arriv_{\mu}$) on Waiting Time…… 127

36. Impacts of arrival distribution CV ($Arriv_{CV}$) on Waiting Time…… 128

Figure Page

37. Impacts of block size (B) on Waiting Time…… 128

38. Impacts of workload by other services (ρ_O) on Operation Time…… 130

39. Impacts of arrival distribution mean ($Arriv_\mu$) on Operation Time…… 130

40. Impacts of arrival distribution CV ($Arriv_{CV}$) on Operation Time…… 131

41. Impacts of block size (B) on Operation Time……. 132

42. Impacts of arrival distribution mean ($Arriv_\mu$) on Completion Rate…..... 133

43. Impacts of arrival distribution CV ($Arriv_{CV}$) on Completion Rate……. 133

44. Impacts of block size (B) on Completion Rate……... 134

CHAPTER 1

INTRODUCTION

Service oriented computing (SOC) has emerged as a major research topic in recent years. Strong support from major computer and IT service providers companies such as IBM, Microsoft, Hewlett Packard, Oracle, SAP and Amazon has accelerated the acceptance and adoption of SOC. The loosely coupled nature of SOC allows companies to build new value-added services or upgrade existing services in a granular fashion to address new business needs. Service Oriented Architecture (SOA), the platform allowing the implementation of the SOC paradigm, has been adopted in various distributed systems such as web services, grid computing and cloud computing systems. Resource management and service performance are key aspects in SOA. Because system resources are shared among services, workloads placed on resources by services and their impact on service performance as a result of allocating resources to services must be considered in several stages of the IT services' life cycle, including modeling, composition, monitoring, optimization and management. In the modeling stage, service workload and performance must be estimated so that in the composition stage this knowledge can be used to allocate resources to services for satisfying service performance requirements. In the monitoring, optimization and management stages, service workload and performance need to be monitored so that resources and services can be adapted and optimized to accommodate dynamic system changes. The growing complexity and demand of services make individual IT

efforts for managing service-based systems (SBS) costly and inefficient. Service standardization is required to manage service-based systems efficiently. Existing efforts on service standardization (Curbera, et al. 2002), including UDDI (Universal Description Discovery and Integration), SOAP (Simple Object Access Protocol), WSDL (Web Service Description Language) and WSMO (Web Service Modeling Ontology), focus on the functional aspects of services and their specification for service discovery and interoperability. However, those standards do not provide support for non-functional aspects of services, such as service performance which is of particular concern for IT service providers as it directly affects client's satisfaction and loyalty (Subrata, Zomaya and Landfeldt 2008).

Service standardization should consider both functional and non-functional aspects of services. With this need in mind, ontologies and templates have been proposed for the specification of both functional and non-functional (e.g., workload, performance) aspects of services (Wang, et al. 2006; Lamparter, Ankolekar and Studer 2007; Hu, Cao and Gu 2008; Tran, Tsuji and Masuda 2009; Staikopoulos, et al. 2010). However, these studies fail to consider the dynamic nature of the execution environment as the availability of system resources and service demands change over time. For example, services demands can fluctuate, having higher demand peaks at rush hours but lower demands at other times. Similarly, resource availability can change due to communication overhead, hardware failure or cyber attacks. From the perspective of resource management, a service request adds activities and workload to system resources. If the machine hosting the service cannot provide enough resources to the service, the service is

likely to perform with a degraded performance (Li, et al. 2005). Different services consume different types and quantity of resources (e.g. CPU, memory, disk, network and so on.), and service performance or usually called “quality of service” (QoS) depends on the amount of resources assigned to the service (Wu and Woodside 2004; Stewart and Shen 2005; Zhang, Bivens and Rezek 2007). There exists a cause-effect relation of service activities (A) with resource workload/state (S) and service performance/quality (Q). However, models capturing these relations are not readily available from the design of system and application software which provides mostly logic-based operational models rather than workload and performance models. Previous studies on resource workload and service performance (Vazhkudai and Schopf 2002; Doyle, et al. 2003; Shivam, Babu and Chase 2006; Sun and Ifeachor 2006; Kan, Sun and Ifeachor 2010; Kang and Suh 2011; Zhang, Verma and Cheng 2011) address particular services or focus on specific resources, and limited system aspects. Workload and performance models are required at a more comprehensive, system-wide scale considering multiple resources, their interactions, and the impacts of service activities.

This research focuses on establishing systematic methods to understand and model the cause-effect relations of service-related activities (A) with resources workload/state (S) and service performance/quality (Q). The workload and performance models of services obtained through these methods support service standardization for modeling, composition, monitoring, optimization and management stages of service-based systems (SBS). Additionally by uncovering

these cause-effect (ASQ) relations, insights about resource and performance tradeoffs of services are obtained.

In chapter 2, an empirical method is proposed to analyze and model the impacts of services on system activities, resources workload and service performance. This method involves the collection of system-wide dynamics data and the application of statistical analyses to uncover and model resource workload and service performance. Various types of services and service scenarios are investigated, including a motion detection service (MDS) and four variations of the voice communication service (VCS): VCS, VCS with background network traffic, VCS with data encryption, and VCS with data encryption and background network traffic. The results uncover system-wide impacts of these services on resources workload and service performance, and identify interaction effects of multiple services running concurrently.

In chapter 3, the empirical method (in chapter 2) is used to investigate the impacts of services, security mechanisms and cyber attacks on resources workload and service performance, and to explore the implications of these impacts in developing strategies for system survivability. System dynamics data is collected under the conditions of two services of voice communication and motion detection, two security mechanisms of data encryption and intrusion detection and five cyber attacks (ARP poison, ping flood, vulnerability scan, fork bomb and remote dictionary). The results uncover the impacts of services, security and cyber attacks on resource workload and service performance, and

reveal important tradeoff effects that can be used in developing strategies for system survivability.

The empirical method captures the cause-effect (ASQ) relations of services, resources workload and service performance. However, the empirical method is limited by the time and effort required for experimental set-up, data collection and analysis. To overcome this limitation, in chapter 4 a framework is proposed to estimate the impacts of services on resource workload and service performance based on the assumption that system dynamics are mainly driven by: 1) the resource-sharing scheme of the system resources, including: admission control, allocation method, scheduling policy, and 2) the resource requirements (profile) of services competing for the resource. The framework is used to build service workload and performance models at processor and disk resources. These models can be used as quantitative basis for efficient management of resource workload and service performance in service-based systems.

CHAPTER 2

IMPACTS OF SERVICES ON SYSTEM ACTIVITIES, RESOURCES WORKLOAD AND SERVICE PERFORMANCE

2.1 Literature review

As people increasingly rely on online services deployed on computer and network systems to support operations in banking, telecommunications, transportation and many other conventional domains, service performance (quality) has become a major concern for IT service providers as it directly affects users' satisfaction and loyalty (Subrata, Zomaya and Landfeldt 2008). Service performance is usually measured through different metrics such as throughput, delay, jitter, accuracy, security, and so on. According to service functionality, some of these performance metrics may be more critical for a specific service than others. A list of performance metrics for various common services can be found in (Chen, Farley and Ye 2004). For example in a voice communication service (VCS), critical metrics for service performance are throughput and delay of voice data transmission. IT service providers are required to satisfy both functional and non-functional (performance) service aspects. The increasing diversity and demand of services specifically tailored to user requirements have increased the complexity of IT systems to a point where standardization is required to handle important system aspects such as resource management and service performance efficiently. When competing service requests with specific performance requirements are received, the IT service provider must determine if

it has enough resources to satisfy the service requests, including performance metrics, and the service and resource configurations required. Existing service standards such as UDDI (Universal Description Discovery and Integration), SOAP (Simple Object Access Protocol), WSDL (Web Service Description Language) and WSMO (Web Service Modeling Ontology) have been developed to provide support for functional aspects of services and their specification for services discovery and interoperability. However those standards do not provide support for non-functional aspects of services, such as service performance.

Multiple studies have focused on extending semantics to incorporate service performance metrics through the use of ontologies and templates. Ontologies capturing functional and non-functional (e.g. workload, performance) service aspects within standards-based specification language were developed in Wang, et al. (2006), Lamparter, Ankolekar and Studer (2007) and Tran, Tsuji and Masuda (2009). Customizable semantic templates were proposed to model functional and non-functional aspects of web services in Hu, Cao and Gu (2008) and Staikopoulos, et al. (2010). However these ontologies and templates assume user-defined functions for performance metrics and fail to consider the dynamic nature of the execution environment as the availability of system resources and service demands change over time. From the resource management perspective, competing service requests and their associated system activities add workload to system resources which in turn affects service performance (Wu and Woodside 2004; Stewart and Shen 2005; Zhang, Bivens and Rezek 2007). Models capturing these cause-effect relations of service-related activities (A) on resources

workload/state (S) and service performance/quality (Q) are required for service standardization. Existing studies on resource workload and service performance models address particular services or focus on specific resources, and limited system aspects. For example, Vazhudai and Schopf (2002) used regression models to characterize the impact of I/O load variations on file transfer times in data grids. Doyle, et al. (2003) build internal-component models to predict the utilization of memory and storage resources for services with static content. Shivam, et al. (2006) studied the impact of various assignments of computing, network and storage resources on the completion time for batch processing tasks. Sun and Ifeachor (2006) used nonlinear regression models to predict the performance in a voice over IP (VoIP) setting by codec types under different network loads. Kan, et al. (2010) provided a prediction model for video quality on wireless networks based on network state metrics. Zhang, et al. (2011) developed a competitive market model for resource allocation by considering network delay in multi-class networks. Kang and Suh (2011) modeled the tradeoff between two service quality metrics: delay and reliability on wireless network transmissions. Workload and performance models are required at a more comprehensive, system wide scale considering multiple resources, their interactions, and the impacts of service-related activities. A systematic approach is required to capture these cause-effect relations independently of services functional and non-functional requirements.

2.2 Shortcomings

Based on the above literature review, shortcomings from existing research can be summarized as follows:

1) Available service standards do not provide support for resource workload and service performance specifications.
2) Ontologies and templates extending service standards assume user-defined functions for workload and performance metrics and fail to consider system dynamics.
3) Workload and performance models available from existing studies address particular services or focus on specific resources, and limited system aspects.

2.3 Objectives

Address the above shortcomings by developing an empirical method to analyze and model the impacts of service-related activities (A) on resources workload/state (S) and service performance/quality (Q). The empirical method should be able to capture ASQ relations independently of services functional and non-functional requirements.

Use the empirical method to gain insights about resource and performance tradeoffs of services, so this information together with the ASQ models can be considered for service standardization.

2.4 Methodology for data collection, analysis and modeling

The proposed method involves the collection of system-wide dynamics data and the application of statistical analyses to uncover and model resources workload and service performance. This section describes the process of data collection, analysis and modeling.

2.4.1 Data Collection

A system monitor tool was developed to collect Windows performance variables (Microsoft Corporation 2003) during experimental service scenarios. The data is collected from the server computer and reflects the impacts of service-related activities (A) on resources workload/state (S), and service performance/quality (Q). System dynamics variables from Windows performance objects such as Process, Processor, Memory, System, IP, TCP, UDP, Paging file, Server, web services and other objects can be collected simultaneously.

2.4.2 Data Analysis

The data collected from the experimental service scenarios is analyzed to investigate the impacts of service parameters (which mainly drive service-related activities) on resources workload/state (S) and service performance/quality (Q). The data analysis takes two steps: 1) data screening, and 2) effects analysis. The data screening first removes the variables falling into the following categories:

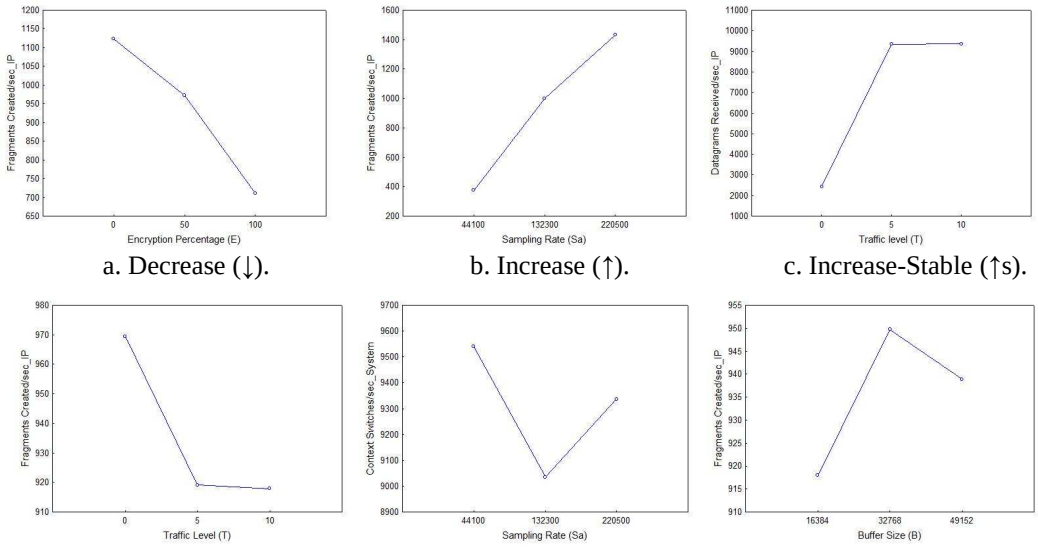
- A variable records the highest or peak value since the server computer is restarted. For example, the variable, Virtual Bytes Peak of the Process

object, records the highest virtual address space in bytes used by the VCS process since the server computer is restarted. Hence, values of the variable keep increasing over time.

- A variable collects the cumulative value over time since the server computer is restarted. For example, the variable, Datagrams Outbound Discarded of the IP object, counts the number of output IP datagrams with no errors that are discarded due to reasons such as lack of buffer space since the computer is restarted.
- A variable collects data that is not affected by the experimental conditions. For example, the variable, Priority Base of the Process object measures the base priority of a service process. Under the experimental scenarios the priority of the service processes is constant through all conditions.

For the variables of the Windows performance objects remaining after the data screening, ANOVA is performed using Statistica7 for each service scenario and each variable of Windows performance objects, with the variable as the dependent variable and the service parameters involved in the service scenario as the independent variables. ANOVA reveals the impacts of the service parameters individually and together on the system dynamics variable from Windows performance objects. If ANOVA results indicate a significant impact of one or more service parameters on a variable, the Tukey's honest significant difference (HSD) test in Statistica7 is performed to reveal how different levels of the service parameters affect the system dynamics variable. Figure 1 shows examples of the following six different impacts revealed through Tukey's test.

- Decrease (↓): the value of the S or Q variable decreases as the service parameter level increases. (see Figure 1.a)
- Increase (↑): the value of the S or Q variable increases as the service parameter level increases. (see Figure 1.b)
- Increase-Stable (↑s): the value of the S or Q variable increases and then keeps stable as the service parameter level increases. (see Figure 1.c)
- Decrease-Stable (↓s): the value of the S or Q variable decreases and then keeps stable as the service parameter level increases. (see Figure 1.d)
- V (v): the value of the S or Q variable decreases and then increases as the service parameter level increases. (see Figure 1.e)
- Inverse-V (Λ): the value of the S or Q variable increases and then decreases as the service parameter level increases. (see Figure .f)



a. Decrease (↓). b. Increase (↑). c. Increase-Stable (↑s).

d. Decrease-Stable (↓s) e. V (v). f. Inverse-V (Λ).

Figure 1. Different impacts of service parameters on system dynamics variables from the VCS with data encryption service and background network traffic.

If a system dynamics variable is significantly affected by more than one service parameter, the partial eta-squared index (Olejnik and Algina 2003) in Statistica7 is obtained to determine the impacts size so the impacts of the service parameters can be ordered by their sizes. The service parameter with the largest impact size on a system dynamics variable affects the system dynamic variable most. The system dynamics variables are then grouped into categories according to their impacts with service parameters and the size of those impacts.

2.4.3 Data Modeling

Among the system dynamics variables that appear in the impacts categories and considering the cause-effect chains of service parameters (A) on resources workload/state (S), and service performance/quality (Q), linear regression models are built to capture: 1) quantitative relations of service parameters (A) with resource workload/state (S) variables and 2) quantitative relations of resource workload/state (S) variables with the service performance/quality (Q) metrics. Only a representative subset of resource workload/state (S) variables is selected to build these workload and performance models. The selection of this subset of variables should be based on expert domain-knowledge of the services under consideration. Statistica7 is used to build the linear regression models.

2.5 Description of Experimental Scenarios

Two sets of experimental scenarios are implemented. One set of experiments involves four different variations of the voice communication service (VCS): VCS-only, VCS with background network traffic (VCS & Traffic), VCS with data encryption for security (VCS & Security) and VCS with data encryption and background network traffic (VCS & Security & Traffic). The voice communication service is a communication-intensive service. The data encryption service is used to encrypt voice data in the voice communication service for security purposes. Background network traffic is added to represent additional network activities that may occur during the voice communication service. Another set of experiments involves a motion detection service which is a computation-intensive service. Figure 2 shows the computer and network set-up for the two sets of experiments. The computer and network set-up consists of seven computers: one computer as a server, five computers running clients (one client on one computer), and one computer to generate network traffic. Each computer has 1 GB memory and Intel Pentium 4 processor of 2.2 GHz. All computers have Windows XP operating system with service pack 2 (SP2). The computer and network set-up stands alone without any other network connections to avoid interferences.

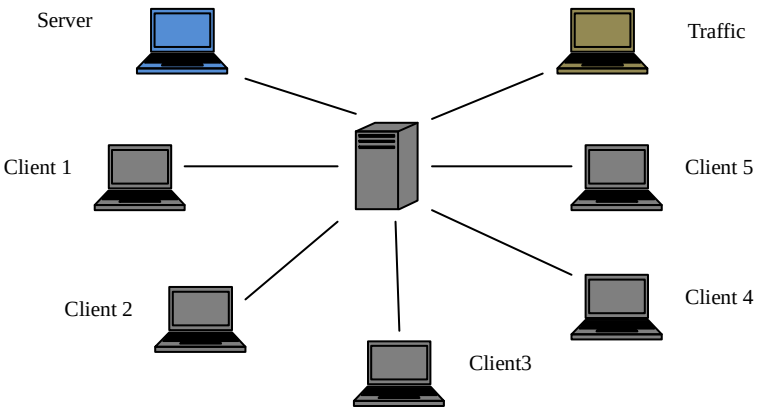


Figure 2. Computer and network set-up for VCS and MDS experiments.

In the voice communication service (VCS), a client sends a service request to the server via the network, and the server sends the requested voice data to the client. VCS is implemented by converting an open-source video conference software package (Abdel-qader 2007) into a web service using C# in .NET. The data encryption service provides data encryption using the Advanced Encryption Standard (AES) algorithm developed by Daemen and Rijmen (2001). The data encryption algorithm is implemented within VCS. If requested, voice data is encrypted on the VCS server before it is transmitted over the network to a client. The computer generating the background network traffic uses a web service to generate traffic by continuously sending data packets over the network to the server computer.

In the motion detection service, clients send service requests to the server to analyze video streams to detect motion. To focus on the computation-intensive aspect of the motion detection service, pre-recorded video files (each file with a different video resolution) stored on the server computer are used rather than

having video data transmitted over the network. When a client requests the motion detection service, a video file with a specified video resolution is opened and processed frame by frame at the rate of 20 frames per second to simulate real-time video streaming from peripheral devices such as a webcam. Video data is analyzed using a motion detection algorithm which is implemented by converting an open-source motion detection algorithm package (Kirillov 2007) into a web service using C# in .NET. The detection algorithm first extracts a reference frame from the initial frames of a video stream, and then calculates differences between the subsequent frames and the reference frame. Multiple clients can simultaneously request the server to process a video stream with a specified video resolution to detect whether there is any motion. A process thread is created for each client.

2.5.1 Service parameters in experimental scenarios

For VCS, three service parameters are used to produce various levels of VCS activities: the sampling rate (*Sa*), the number of clients (*C*), and the size of the buffer (*B*) for holding voice data before transmitting the data to the clients over the network. The sampling rate is the frequency of sampling voice data from the sound card. A higher sampling rate gives a better quality of voice data and yields more voice data. More VCS clients produce more workload on resources and more voice data. Table 1 lists the levels of the service parameters of the VCS used in the experiments.

For the security service of data encryption, two service parameters are used to produce various levels of data encryption activities: the encryption percentage (*E*) and the key length (*K*). The encryption percentage represents the percentage of VCS voice data being encrypted. The key length is the size of the key used for data encryption in the AES algorithm (Daemen and Rijmen 2001). A larger key length produces a stronger security. The levels of the encryption percentage and the key length used in the experiments are also listed in Table 1. For generating background network traffic, the number of threads created to generate and send packets to the server varies in the experiments as shown in Table 1. Each thread continuously generates and sends 32Kbytes packets to the server over the network.

Table 1. Levels of the service parameters for VCS experiments.

Service parameters	**Level 1**	**Level 2**	**Level 3**
Traffic (*T*)	0	5	10
Percent Encryption (*E*)	0%	50%	100%
Key Length (*K*)	128 bits	192 bits	256 bits
Sampling rate (*Sa*)	44,100 Hz	132,300 Hz	220,500 Hz
Number of clients (*C*)	1	3	5
Buffer size (*B*)	16 Kbytes	32 Kbytes	48 Kbytes

The motion detection service (MDS) has two service parameters: the video resolution (*R*) and the number of clients (*C*). A higher video resolution places a higher workload of analyzing more video data for motion detection and a longer delay of processing each video frame, but allows a better accuracy of motion detection since smaller moving objects can be captured. Each client runs one thread of motion detection. The levels of these two service parameters for MDS

are listed in Table 2. These service parameters produce various levels of MDS activities.

Table 2. Levels of the service parameters for MDS experiments.

Service parameters	**Level 1**	**Level 2**	**Level 3**
Video Resolution (*R*)	22x18px	44x36px	88x72px
Number of clients (*C*)	1	3	5

For the set of experiments involving VCS: VCS-only, VCS & Security, VCS & Traffic, VCS & Security & Traffic, and for the set of experiments involving MDS six Windows performance objects, namely, process, processor, memory, system, IP and web service are used to collect totally 186 variables of system dynamics data. Each combination of service parameters and their levels is run as an experimental condition. For example, the VCS-only scenario has nine experimental conditions for nine combinations which result from three service parameters of VCS and three levels of each parameter, respectively. For the set of experiments involving VCS, all the experimental conditions for all the four service scenarios are arranged in a random order and run continuously from one experimental condition to the next experimental condition. Then, this random order is reversed, and the reversed order is used to run all the experimental conditions for all the four service scenarios again after the computer network is cleaned up and restarted.

For the set of experiments involving MDS, all the nine experimental conditions of MDS are first run in a random order and after the computer network is cleaned up and restarted, the reversed order is run. The data collected from the

two opposite orders is used in the data analysis so that a specific order of running the experimental conditions does not affect the data analysis results. Each experimental condition is run to collect 30 data observations with a rate of one observation per second.

2.6 Results and Discussions

In this section, the impacts of VCS, data encryption and background network traffic uncovered by analyzing the data from the VCS experiments are presented. Also the impacts of MDS uncovered by analyzing the data from the MDS experiments are presented. Models capturing the impacts of service parameters (A) on resources workload/state (S) and service performance/quality (Q) for the experiments are provided.

2.6.1 Impacts of VCS

The data screening and effects analysis of the data collected from the set of the experiments involving VCS reveal the following:

- In the VCS-only scenario, 37 variables of Windows performance objects are significantly affected by at least one of the three service parameters (*Sa*, *C* and *B*) of VCS,
- In the VCS & Traffic scenario, 45 variables of Windows performance objects show a significant impact by at least one of the four service parameters of VCS and background network traffic (*Sa*, *C*, *B* and *T*),

- In the VCS & Security scenario, 43 variables of Windows performance objects show a significant impact by at least one of the five service parameters of VCS and the data encryption service (*Sa*, *C*, *B*, *E* and *K*), and
- In the VCS & Security & Traffic scenario, 46 variables of Windows performance objects show a significant impact by at least one of the six service parameters of VCS, data encryption and traffic (*Sa*, *C*, *B*, *E*, *K* and *T*).

To determine the consistent impacts of the VCS parameters, *Sa*, *C* and *B*, the results from the VCS-only, VCS & Security, VCS & Traffic, and VCS & Security & Traffic scenarios are compared to identify the impacts of *Sa*, *C* and *B* that remain constant across these scenarios. Table 3 shows the three groups of system dynamics variables according to their impacts with the service parameters of VCS. In Table 3, service parameters in each group are ordered according to their effect size. The service parameter with the largest impact is listed first.

Table 3. Consistent Impacts with VCS.

Impacts with VCS parameters	**Object**	**Variable**
1. *Sa*↑*C*↑*B*↓ (11 variables)	IP (2 variables)	*Performance (Q) variables:* Fragments Created/sec, Fragmented Datagrams/sec.
	Process (7 variables)	*State variables:* % Processor Time, % User Time, % Privileged Time. *Activity variables:* IO Other Operations/sec, IO Other Bytes/sec, Thread Count, Handle Count.
	Processor (2 variables)	*Activity variables:* % User Time, % Privileged Time.
2. *Sa*↑*C*↑ (3 variables)	IP (1 variable)	*Activity variable:* Datagrams Sent/sec.
	Memory (1 variable)	*Activity variable:* Cache Faults/sec.
	System (1 variable)	*Activity variable:* File Control Bytes/sec.
3. *C*↑ (2 variables)	Web Service (2 variables)	*Activity variables:* Current Anonymous Users, Current Connections.

Group 1, *Sa*↑*C*↑*B*↓: this group contains

- System activity variables measuring counts of threads/handles and associated IO other operations and bytes for scheduling and synchronizing threads;
- Resource workload/state variables measuring the CPU utilization of VCS in the user mode and the privileged mode;
- Performance variables of VCS measuring IP fragments created by the server to send to the clients and the resulting fragmented datagrams.

The increase in values of these variables with the increasing level of VCS activities through *Sa* and *C* indicates that CPU utilization in the user mode and outgoing IP fragments are the main characteristics of VCS's workload and performance. The CPU utilization in the privileged mode also increases as the

level of VCS activities is raised due to the increase in scheduling and synchronization activities in privileged mode caused by the increase in VCS threads. The system dynamics variables in this group decrease their values as the buffer size increases because a larger buffer size results in a smaller frequency of sending out larger amounts of voice data each time. This reveals an important way of reducing the resource workload of VCS by increasing the buffer size. However, the throughput of VCS also decreases due to a larger buffer size.

Group 2, $Sa\uparrow C\uparrow$: this group includes

- System activity variables measuring cache faults and file control bytes associated with IO other operations and bytes in group 1;
- Performance variable measuring the IP datagrams sent out by the VCS server to the VCS clients.

The variables in this group, which measure the VCS performance and system activities for scheduling and synchronizing VCS threads, are similar to the system dynamics variables in group 1 and increase their values with the increasing level of VCS activities through *Sa* and *C*. However, the impacts of *B* on the system dynamics variables in group 2 are either weak or inconsistent across different VCS scenarios.

Group 3, $C\uparrow$: this group contains the system activity variables measuring the number of current network connections through IIS/web service which increase their value only with the increasing number of VCS clients.

In overall, the dominant impact of VCS manifest in outgoing network data, cache usage, processor usage in the user mode, and network connections for VCS clients. The increase in threads/handles, and CPU utilization in the privileged mode results from the increase in system activities.

2.6.2 Impacts of background network traffic

To determine the consistent impacts of the background network traffic parameter, *T*, the analysis results from the VCS & Traffic and VCS & Security & Traffic scenarios are compared to identify the impacts of *T* that remain constant across these scenarios. Table 4 shows the four groups of system dynamics variables according to their impacts with *T*.

Table 4. Consistent Impacts with background network traffic.

Impacts with Traffic	**Object**	**Variable**
1. *T*↓s (6 variables)	Memory (2 variables)	*Activity variables:* Demand Zero Faults/sec, Page Faults/sec.
	Process (3 variables)	*Activity variables:* IO Other Operations/sec, IO Other Bytes/sec, Thread Count.
	Processor (1 variable)	*State variable:* % User Time.
2. *T*↓ (8 variables)	System (8 variables)	*Activity variables:* System Calls/sec, Context Switches/sec, File Read Bytes/sec, File Write Bytes/sec, File Read Operations/sec, File Write Operations/sec, File Data Operations/sec, File Control Operations/sec.
3. *T*↑s (8 variables)	IP (3 variables)	*Activity variables:* Datagrams Received Delivered/sec, Datagrams Received/sec, Datagrams/sec.
	Processor (4 variables)	*Activity variables:* DPC Rate, DPCs Queued/sec, Interrupts/sec. *State variable:* % Privileged Time.
	System (1 variable)	*Activity variable:* File Control Bytes/sec.
4. *T*↑ (1 variable)	Memory (1 variable)	*Activity variable:* Committed Bytes.

Group 1, *T*↓s: this group includes several variables in group 1 of Table 3 that increase with more VCS activities. These variables decrease their value as *T* increases from level 1 to level 2 because VCS and traffic compete for CPU and network bandwidth. An increase in background network traffic reduces the usage of CPU in the user mode and network bandwidth by VCS and thus decreases system activities of VCS. However, VCS activities stop decreasing and remain at the similar level as T increases from level 2 to level 3 due to the saturation of the network bandwidth by the incoming background traffic and the consequent drop of the additional incoming network traffic.

Group 2, *T*↓: this group contains the variables related to file data (read and write) operations and bytes, file control operations, system calls and context switches. These variables decrease with the increasing level of *T* because more CPU time is spent on processor interrupts from the network interface card due to the incoming background traffic, leaving less CPU time for context switches and system calls to operating system service routines for CPU scheduling.

Group 3, *T*↑s: this group includes incoming network data which is measured by IP object variable and increases with *T*. The incoming network data is the main characteristic of background network traffic in the experiments. The rest of the variables in group 3 measure processor interrupts from the network interface card to handle incoming network traffic and CPU utilization in the privileged mode for those processor interrupts, and increase their values with *T*. File control bytes in the system increase with both *T* in Table 4 and *Sa* and *C* in

Table 3. Hence, file control bytes in the system seem to increase with more activities in the system, including VCS and incoming network traffic. The values of the variables in group 3 no longer increase as *T* increases from level 2 to level 3. This leveling off effect may be caused by the saturation of the network bandwidth by the highest level of *T* in the experiments and the consequent drop of additional incoming network traffic.

Group 4, *T*↑: Committed Bytes of the Memory object in Group 4 reflect the memory usage by the incoming network traffic and increase with *T*.

2.6.3 Impacts of security service of data encryption

To determine the consistent impacts of the activity parameters, *E* and *K*, the analysis results from the VCS & Security and VCS & Security & Traffic scenarios are compared to identify the impacts of *E* and *K* that remain the same across these scenarios. Table 5 shows the two groups of system dynamics variables according to their impacts with *E* and *K*.

Table 5. Consistent Impacts with security service of data encryption.

Impacts with Security parameters	**Object**	**Variable**
1. *E*↓ (8 variables)	IP (2 variables)	*Performance (Q) variables:* Fragments Created/sec, Fragmented Datagrams/sec.
	Processor (1 variable)	*State variable:* % Privileged Time.
	System (5 variables)	*Activity variables:* File Control Operations/sec, File Data Operations/sec, File Write Operations/sec, File Read Operations/sec, System Calls/sec.
2. *E*↑ (32 variables)	IP (4 variables)	*Activity variables:* Datagrams Received Delivered/sec, Datagrams Received/sec, Datagrams/sec, Datagrams Sent/sec.

	Memory (5 variables)	*Activity variables:* Cache Faults/sec, Demand Zero Faults/sec, Page Faults/sec, Page Reads/sec, Page Inputs/sec.
	Process (13 variables)	*Activity variables:* Page Faults/sec, IO Other Operations/sec, IO Other Bytes/sec, IO Data Bytes/sec, IO Write Bytes/sec, IO Write Operations/sec, IO Read Bytes/sec, IO Data Operations/sec, IO Read Operations/sec, Thread Count. *State variables:* % Processor Time, % User Time, % Privileged Time.
	Processor (4 variable)	*State variable:* % User Time. *Activity variables:* DPC Rate, DPC Queued/sec, Interrupts/sec.
	System (3 variables)	*Activity variables:* File Control Bytes/sec, File Read Bytes/sec, File Write Bytes/sec.
	Web Service (3 variable)	*Activity variable:* Current Anonymous Users, Current Connections, Post Requests/sec.

Group 1, $E\downarrow$: this group includes the IP variables which measure the throughput performance of VCS and decrease their values by increasing encryption percentage due to the time required to encrypt voice data packets before transmission and the resource competition between the data encryption part and the data transmission part of VCS. The data encryption slows down the rate of sending out encrypted voice data for VCS. The % Privileged Time of the processor object for scheduling and synchronization of activities in the system also decreases with E due to more computation time for data encryption in the user mode. More data encryption decreases file data (read and write) and control operations but increases file data and control bytes in the system (group 2).

Group 2, $E\uparrow$: this group contains IO reading and writing data bytes in IO data operations and associated system activities (including threads, page faults in cache and memory, CPU utilization in the user mode, and processor interrupts from data channels and disk drivers) which increase with E due to more data encryption work. CPU utilization in privileged mode also increases with E for

scheduling and synchronizing more activities in the system. Current network connections via the web service increase with *E* because each connection for sending out encrypted voice data lasts longer. The total amount of IP datagrams (received and sent) increases with *E* because the data encryption slows down the use of network bandwidth by VCS, leaving more network bandwidth available for incoming network traffic.

Hence, the security service of data encryption is characterized by more reading and writing data bytes in IO operations on data channels in the system and an associated increase in threads, cache and memory usage, CPU utilization, and overall activities in the system. The competition between the data encryption and the network data transmission for CPU time exists, causing a decrease in the throughput of VCS network data transmission and a longer network connection session as the encryption percentage increases. The decrease in the throughput of VCS in turn leaves more network bandwidth for incoming background traffic. The encryption percentage has much larger impacts than the key length which shows only weak or inconsistent impacts on the system dynamics variables affected by *E*.

2.6.4 A summary of the impacts with VCS, security and traffic

Table 6 summarizes major groups of system activity, workload/state and performance variables with similar impacts from more than one parameter of VCS, security and traffic.

Table 6. Summary of system dynamics variables affected by more than one service parameter.

Group	Variable	Object	Impacts with VCS	Impacts with Security	Impacts with Traffic
1	% User Time	Processor	*Sa*↑*C*↑	*E*↑	
	% Privileged Time	Process	*Sa*↑*C*↑	*E*↑	
	% Processor Time				
	% User Time				
	IO Other Operations/sec				
	IO Other Bytes/sec				
	Thread Count				
	Datagrams Sent/sec	IP	*Sa*↑*C*↑	*E*↑	
	Cache Faults/sec	Memory	*Sa*↑*C*↑	*E*↑	
2	Fragmented Datagrams/sec	IP	*Sa*↑*C*↑*B*↓	*E*↓	*T*↓s
	Fragments Created/sec				
3	File Control Bytes/sec	System	*Sa*↑*C*↑	*E*↑	*T*↑s
4	Datagrams/sec	IP		*E*↑	*T*↑s
	DPC Rate	Processor		*E*↑	*T*↑s
	DPCs Queued/sec				
	Interrupts/sec				
5	File Control Operations/sec	System		*E*↓	*T*↓
	File Data Operations/sec				
	File Write Operations/sec				
	File Read Operations/sec				
	System Calls/sec				
6	Page Faults/sec	Memory		*E*↑	*T*↓s
	Demand Zero Faults/sec				
	File Read Bytes/sec	System		*E*↑	*T*↓
	File Write Bytes/sec				
7	Current Anonymous Users	Web Service	*C*↑	*E*↑	
	Current Connections				

In summary, VCS produces an increase in the following:

- Outgoing network data (see group 2 in Table 6),
- CPU utilization in the user mode (see group 1 in Table 6), and
- Network connections for VCS clients (see group 7 in Table 6).

The security service of data encryption produces an increase in the following:

- Reading and writing data bytes in IO operations on data channels in the system,
- CPU utilization in the user mode (see group 1 in Table 6),
- Cache and memory usage (see group 1 and group 6 in Table 6), and
- Processor interrupts from data channels and disk devices (see group 4 in Table 6).

Background network traffic in the experiments produces an increase in the following:

- Incoming network data (group 4 in table 6), and
- Processor interrupts from the network interface card (see group 4 in Table 6).

VCS, the security service and background traffic all increase activities in the system, which consistently manifest in the increase in

- File control bytes/sec in the system (see group 3 in Table 6).

Both VCS and the security service of data encryption create threads and require CPU privileged time and IO other operations and bytes for scheduling and synchronizing threads (see group 1 in Table 6).

VCS, the security service of data encryption and background network traffic compete for system resources in the following ways:

- Competition between the data encryption and the network data transmission for CPU time: more data encryption causes a decrease in the throughput of VCS network data transmission and a longer network connection session which in turn leaves more network bandwidth for incoming background traffic (see group 2 and group 7 in Table 6).
- Competition between VCS and incoming background traffic for CPU time and network bandwidth. More incoming background traffic reduces VCS activities (see group 2 in Table 6).

In group 5 of Table 6, file data (read and write) and control operations in the system are reduced by more data encryption and background traffic because the repetitive use of the same data files by the data encryption service and background traffic. Also in group 5 of Table 6, system calls/sec in the system are reduced by more data encryption and background traffic due to more CPU time on handling more interrupts from data channels and the network interface card and thus less CPU time for CPU scheduling and synchronization through system calls.

2.6.5 Impacts of MDS

There are 46 variables of Windows performance objects that show significant impacts with the MDS parameters, *C* and *R*, as shown in Table 7. The MDS parameter with the largest impact size is listed first in Table 7.

Table 7. Impacts of MDS parameters on system dynamics variables

Impacts with MDS parameters	**Object**	**Variables**
1. *C*↑*R*↑ (4 variables)	Process (2 variables)	*State variables:* % Processor Time, % User Time.
	Processor (1 variable)	*State variables:* % User Time.
	System (1 variable)	*Activity variable:* Exception Dispatches/sec.
2. *C*↑*R*↑s (12 variables)	Process (9 variables)	*Activity variables:* IO Other Operations/sec, IO Read Bytes/sec, IO Data Bytes/sec, Handle Count. *State variables:* Working Set, Page File Bytes, Private Bytes, Virtual Bytes, Pool Nonpaged Bytes.
	System (2 variables)	*Activity variables:* File Read Bytes/sec, Processes.
	Memory (1 variable)	*State variable:* Committed Bytes.
3. *C*↑ (10 variables)	Process (1 variable)	*State variable:* % Privileged Time.
	System (1 variable)	*State variable:* Processor Queue Length.
	Web Service (8 variables)	*Activity variables:* Current Anonymous Users, Current Connections, Current ISAPI Extension Requests, Bytes Sent/sec (*C*s↑), Bytes Received/sec (*C*s↑), Bytes Total/sec (*C*s↑), Files Sent/sec, Files/sec (*C*s↑).
4. *R*↓*C*↑ (7 variables)	Process (4 variables)	*Activity variables:* Thread Count, IO Read Operations/sec, IO Data Operations/sec, IO Other Bytes/sec.
	System (3 variable)	*Activity variables:* Threads, File Read Operations/sec, File Data Operations/sec.
5. *C*↓*R*↓ (7 variables)	System (6 variables)	*Activity variables:* File Write Operations/sec, File Control Operations/sec, File Write Bytes/sec, Context Switches/sec, File Control Bytes/sec, System Calls/sec.
	Processor (1 variable)	*State variable:* % Privileged Time.
6. *C*↓*R*↑ (4 variables)	Memory (3 variables)	*Activity variables:* Demand Zero Faults/sec, Page Faults/sec, Cache Faults/sec.
	Process (1 variable)	*Activity variable:* Page Faults/sec.

Group 1 (*C*↑*R*↑) and group 2 (*C*↑*R*↑s): the variables in these two groups measure memory bytes, file and IO read bytes, CPU time in the user mode which are used by MDS processes. MDS processes involve the use of memory in bytes, file and IO read bytes and CPU time in the user mode which increases with more MDS clients and higher video resolutions. File and IO read bytes, associated

memory usage in bytes and MDS processes created for these activities keep stable as *R* increases from level 2 to level 3 possibly due to the constraint on the file read speed.

Group 3 ($C\uparrow$) and group 4 ($R\downarrow C\uparrow$): the variables in this group measure the communication with the web service (i.e., user connections and data communication in files and bytes) which increases with more MDS clients since MDS is implemented using the web service software. More MDS clients produce more MDS threads which in turn cause an increase in % Privileged Time of the processor for the web service and Processor Queue Length due to the scheduling and synchronization of more threads. File and IO read operations also increase with more MDS clients. MDS threads and file and IO read operations of these threads decrease by increasing video resolution because a higher video resolution requires more computation time and produces a longer delay to determine the motion level in a video frame.

Group 5 ($C\downarrow R\downarrow$) and group 6 ($C\downarrow R\uparrow$): the variables in these two groups measure file and IO write operations and bytes, page faults in cache and memory, context switches, system calls to operating system service routines for scheduling and synchronization, and % Privileged Time associated with context switches and systems calls. All MDS clients use the same video files for motion detection and produce the same outcome of motion detection. Hence, more MDS clients using the same data reduce page faults in cache and memory and file and IO write operations to record the motion detection results. This characteristic is associated

with the special feature of the MDS scenario in the experiments, and may not hold for MDS which processes different video files. A higher video resolution causes more page faults in cache and memory because a video frame of a higher resolution has more differences in the data content. Context switches, system calls and associated % Privileged Time on the processor for the overall system decrease with MDS clients due to more computation time and longer processing delay in the user mode. This also causes the longer processor queue as seen in group 3 of Table 7.

Hence, MDS is characterized by file and IO read operations and bytes, memory usage, and CPU utilization in the user mode which increase with MDS clients. Increasing the video resolution has the following impacts:

- Increases page faults in cache and memory, and
- Decreases MDS threads, file and IO read operations from fewer MDS threads, and CPU utilization in the privileged mode for thread scheduling and synchronization, due to a longer processing delay of computing the motion level in a video frame of a higher resolution. However, file and IO read bytes do not decrease with the increasing video resolution because more data is processed for a video frame of a higher resolution.

In summary, MDS produces an increase in File and IO read operations and bytes, and associated cache and memory usage, and CPU utilization in the user mode. Using video data of a higher resolution for motion detection produces more

memory-related system overhead in terms of more page faults and a longer processing delay of computing the motion level for a MDS thread which in turn reduces CPU availability for other processes/threads.

2.6.6 Cause-effect (ASQ) models of system dynamics

ANOVA and Tukey's test uncovered the significant, qualitative relations of service parameters with resources workload/state and service performance of VCS, data encryption service, background network traffic and MDS. Cause-effect (ASQ) models capturing system dynamics for each service can be further built for each performance metric (Q). The performance metrics of each service mainly depend on the workload/state of system resources which are driven by the service parameters. First, models are built to capture the workload/state of system resources (S) with service parameters (A), and then models are built to capture the effect on performance metrics (Q) due to the workload/state of system resources (S). These quantitative models can be used directly to determine resources workload/state and consequently service performance given certain levels of service parameters and support service standardization for services modeling, composition, monitoring, optimization and management stages of service-based systems (SBS).

For the voice communication service (VCS) a major performance metric (Q) is the network throughput of voice data from the VCS server to the VCS clients. Fragments Created/sec of the IP object shown in group 1 of VCS impacts in Table 3, can be used to measure the throughput of VCS. Other performance

metrics for VCS (e.g., processing delay) are not collected in this study. Hence, the VCS throughput is used as an example of how to build the system activity- state-performance dynamics models. For each service scenario involving VCS, system dynamics models are built using the service parameters of all the services involved in the service scenarios. For example in the VCS & Security scenario, the five service parameters of VCS (*Sa*, *C* and *B*) and the security service of data encryption (*E* and *K*) are used to build the system dynamics models. Fragments Created/sec of the IP object, as performance metric for VCS, can be found in Group 2 of Table 6 that summarize major groups of VCS, security and background traffic impacts. The following seven system dynamics variables are selected from the groups in Table 6 with at least one system dynamics variable selected from each group:

- %Processor Time_Process (from Group 1 in Table 6)
- Thread Count_Process (from Group 1 in Table 6)
- Interrupts/sec_Processor (from Group 4 in Table 6)
- File Control Bytes/sec_System (from group 3 in Table 6)
- File Read Bytes/sec_System (from group 6 in Table 6)
- System Calls/sec_System (from group 5 in Table 6)
- Current Connections_Web Service (from group 7 in Table 6).

By using the data collected from the VCS experiments for the above system dynamics variables to perform the linear regression, the system dynamics models for each service scenario involving VCS were built. These models are

shown in Tables 8-11. In these tables, an R^2 value in the range of [0, 1] indicates the goodness-of-fit of a given model to the data. The higher the R^2 value, the better fit of the model to the data.

The model of the performance metric (Q) for VCS in Tables 8-11 has the R^2 value greater than 0.9. This indicates that the performance metric (Q) for VCS can be well predicted from the variables representing resources workload/state which can be predicted from the service parameters in each service scenario. Note that not all the service parameters are needed to predict each variable representing resources workload/state since different system dynamics variables may be affected by different service parameters. Moreover, not all seven variables representing resources workload/state are required to predict the performance metric (Q) for VCS. Three, four, five and seven system dynamics variables are needed in the performance model for the VCS-only, VCS & Traffic, VCS & Security, and VCS & Security & Traffic scenarios respectively. Hence, as the complexity of the service scenarios increases more variables representing resources workload/state are required to predict the performance metric (Q). The R^2 values for the models of some resource workload/state variables (File Read Bytes/sec_System, System Calls/sec_ System, and Current Connections_Web Service in Table 8 and Table 9) with the service parameters are small ($R^2 \leq 0.7$) possibly because these system dynamics variables may have nonlinear relations with the service parameters under these service scenarios. However, under these

service scenarios these resource workload/state variables (S), with small R^2, are not needed in the models for the performance metric (Q).

Table 8. Linear regression models of system dynamics for VCS-only.

Voice Communication Service (VCS)		
S or Q Variable	**Regression Model**	R^2
% Processor Time_Process (S_{PT})	S_{PT} = -1.79 + 0.00003(*Sa*) + 1.17(*C*) - 0.000055(*B*)	0.831
Thread Count_Process (S_{TC})	S_{TC} = 11.2 + 0.000018(*Sa*) + 4.18(*C*)	0.974
Interrupts/sec_Processor (S_I)	S_I = 241 + 0.000385(*Sa*) + 15.3(*C*) - 0.000294(*B*)	0.874
File Control Bytes/sec_System (S_{FCB})	S_{FCB} = - 1344903 + 12(*Sa*) + 517389(*C*)	0.868
File Read Bytes/sec_System (S_{FRB})	S_{FRB} = 93719 - 0.38(*Sa*) - 2994(*C*)	0.397
System Calls/sec_System (S_{SC})	S_{SC} = 84208 - 0.0284(*Sa*) - 2462(*C*)	0.381
Current Connections_Web Service (S_{CC})	S_{CC} = 6.15 + 0.000003(*Sa*) + 1.05(*C*)	0.563
Fragments Created/sec_IP (Q)	Q = - 515 + 0.917(S_{TC}) + 1.27(S_I) + 0.000669(S_{FCB}) Or Q = - 162 + 28.8(S_{PT}) + 1.37(S_{TC}) + 0.000636(S_{FCB})	0.999 0.999

Table 9. Linear regression models of system dynamics for VCS & Traffic.

Voice Communication Service & Traffic (VCS & T)		
S or Q Variable	**Regression Model**	R^2
% Processor Time_Process (S_{PT})	S_{PT} = - 2.01 + 0.00003(*Sa*) + 1.26(*C*) - 0.00006(*B*)	0.826
Thread Count_Process (S_{TC})	S_{TC} = 11.4 + 0.000017(*Sa*) + 4.19(*C*) - 0.000004(*B*)	0.972
Interrupts/sec_Processor (S_I)	S_I = 685 + 115(*T*) - 0.000615(*Sa*) - 28.2(*C*)	0.736
File Control Bytes/sec_System (S_{FCB})	S_{FCB} = 2251534 + 990325(*T*) + 4.54(*Sa*) + 196432(*C*)	0.736
File Read Bytes/sec_System (S_{FRB})	S_{FRB} = 88136 - 1682(*T*) - 0.0274(*Sa*) - 2319(*C*)	0.576
System Calls/sec_System (S_{SC})	S_{SC} = 79480 - 1445(*T*) - 0.196(*Sa*) - 1892(*C*)	0.574
Current Connections_Web Service (S_{CC})	S_{CC} = 6.83 + 0.979(*C*) - 0.000009(*B*)	0.532
Fragments Created/sec_IP (Q)	Q = 1375 + 58.4(S_{PT}) + 6.13(S_{TC}) - 5.38(S_I) + 0.000624(S_{FCB})	0.989

Table 10. Linear regression models of system dynamics for VCS & Security.

Voice Communication Service & Security (VCS & S)		
S or Q Variable	**Regression Model**	**R^2**
% Processor Time_Process (S_{PT})	S_{PT} = - 31 + 0.00015(*Sa*) + 6.57(*C*) - 0.000086(*B*) + 0.396(*E*)	0.787
Thread Count_Process (S_{TC})	S_{TC} = 11.5 + 0.000018(*Sa*) + 4.21(*C*) - 0.000001(*B*)	0.974
Interrupts/sec_Processor (S_I)	S_I = 263 + 0.0003(*Sa*) + 12.9(*C*) - 0.00038(*B*) - 0.141(*E*)	0.775
File Control Bytes/sec_System (S_{FCB})	S_{FCB} = - 2630375 + 17.9(*Sa*) + 759289(*C*) + 26040(*E*)	0.847
File Read Bytes/sec_System (S_{FRB})	S_{FRB} = -2128216 + 9.16(*Sa*) + 385766(*C*) + 29047(*E*)	0.765
System Calls/sec_System (S_{SC})	S_{SC} = 107604 - 0.142(*Sa*) - 6773(*C*) + 0.0758(*B*) - 360(*E*)	0.764
Current Connections_Web Service (S_{CC})	S_{CC} = 5.09 + 0.000002(*Sa*) + 1.51(*C*) + 0.0143(*E*)	0.746
Fragments Created/sec_IP (Q)	Q = - 2526 - 36.6(S_{PT}) + 18.1(S_{TC}) + 7.99(S_I) + 0.000496(S_{FCB}) - 24.6(S_{CC})	0.963

Table 11. Linear regression models of system dynamics for VCS & Security & Traffic.

Voice Communication Service & Security & Traffic (VCS & S & T)		
S or Q Variable	**Regression Model**	**R^2**
% Processor Time_Process (S_{PT})	S_{PT} = -27.5 - 0.121(T) + 0.00014(*Sa*) + 6.09(C) - 0.00007(*B*) + 0.377(*E*)	0.792
Thread Count_Process (S_{TC})	S_{TC} = 11.3 + 0.000018(*Sa*) + 4.21(*C*) + 0.00547(*E*)	0.973
Interrupts/sec_Processor (S_I)	S_I = 613 + 118(*T*) - 0.00042(*Sa*) - 18(*C*) + 0.315(*E*)	0.74
File Control Bytes/sec_System (S_{FCB})	S_{FCB} = 792072 + 990021(*T*) + 10.9(*Sa*) + 470505(*C*) + 26355(*E*)	0.748
File Read Bytes/sec_System (S_{FRB})	S_{FRB} = -1953250 - 15728(*T*) + 8.05(*Sa*) + 341722(*C*) + 5.35(*B*) + 27052(*E*)	0.766
System Calls/sec_System (S_{SC})	S_{SC} = 98673 - 1119(*T*) - 0.117(*Sa*) - 5729(*C*) + 0.0423(*B*) -322(*E*)	0.752
Current Connections_Web Service (S_{CC})	S_{CC} = 5.55 + 1.47(*C*) - 0.000007(*B*) + 0.0163(*E*)	0.741
Fragments Created/sec_IP (Q)	Q = 1833 - 6.2(S_{PT}) + 24.6(S_{TC}) - 6.34(S_I) + 0.000732(S_{FCB}) - 0.00433(S_{SC}) - 32.3(S_{CC}) - 0.000691(S_{FRB})	0.925

2.7 Conclusions

Through conducting the experiments of running different services (voice data communication, data encryption for security, motion detection, and background network traffic) and collecting, analyzing and modeling the experimental data under various services, system-wide impacts of these services on system activities, resources workload/state and service performance were uncovered (Tables 3-7). Specifically, the voice communication service (VCS) produces an increase in outgoing network data, CPU utilization in the user mode, and network connections. The security service of data encryption produces an increase in reading and writing data bytes in IO operations on data channels in the system, CPU utilization in the user mode, cache and memory usage, and processor interrupts from data channels and disk devices. Background network traffic in the experiments produces an increase in incoming network data and processor interrupts from the network interface card. The computation-intensive motion detection service (MDS) produces an increase in file and IO read operations and bytes, and associated cache and memory usage, and CPU utilization in the user mode.

The VCS, security service and background network traffic all increase activities in the system, which consistently manifest in the increase in file control bytes/sec in the system. Both VCS and security service create threads and require CPU privileged time and IO other operations and bytes for scheduling and synchronizing threads. The VCS, security service and background network traffic

compete for system resources, especially CPU time and network bandwidth, resulting in tradeoffs among these services in their resource workload and performance. Although the experimental set-up for the VCS and MDS scenarios is small (1 server, 5 clients) in comparison to typical IT service-based scenarios that can have up to dozens of servers and hundreds or even thousands of clients, the system-wide impacts uncovered for these specific services on system activities, resources workload/state and service performance are still valid for larger set-ups, since, independent of the computer and network set-up, services still generate the same type of system activities and require the same type of system resources to provide the functionality required by clients, although in different quantity depending in the number of client's requests and their performance (QoS) requirements. The regression models that were built for these scenarios to capture the quantitative relations of service parameters with resources workload/state (S) and service performance (Q) have to be used with caution when used to estimate resource workload and performance in larger service scenarios due to the uncertainty generated by model extrapolation.

The method presented in this study for collecting system dynamics data, analysis and modeling can be used to uncover system-wide impacts and identify interaction effects of services, independent of their functional and non-functional requirements. The information uncovered by this method can be used to provide support for service modeling, composition, monitoring, optimization, and management stages of service-based systems (SBS).

CHAPTER 3

IMPACTS OF SERVICE, SECURITY AND CYBER ATTACKS AND THEIR IMPLICATIONS ON SYSTEM WORKLOAD, PERFORMANCE AND SURVIVABILITY

3.1 Literature review

In general, survivability is often defined as the capability of a system to fulfill its mission in a timely manner even in the presence of attacks, failures or accidents (Lipson and Fisher, 1999; Atighetchi et al., 2004; Yi and Zhang, 2005; Zhang et al., 2007; Xiao et al., 2007; Zuo and Panda, 2009). For service-based systems, survivability is linked to service performance. Survivability without some definition of the minimum service performance required to be survivable is meaningless (Li, Shu and Feng 2009). Although service performance metrics may vary according to service functionality, they usually measure performance aspects such as: timeliness, precision and accuracy (Chen, Farley and Ye, 2004). A system unable to adapt in the presence of cyber attacks, failures or accidents is very limited since the presence of these conditions most likely will degrade the service performance to a point below minimum requirements if adaptation decisions are not taken.

A general approach to system survivability involves calling in the reserves for additional system resources. However, reserving additional system resources for unforeseen events can be costly and impractical. There is always a limitation

in how much system reserves can be held. There is always a possibility that the damage caused by cyber attacks lead to a severe shortage of system resources.

For this part of the research different ways of making tradeoffs within the limits of system resources are explored based on the impacts of services, security mechanisms and attacks on system activities, resources workload/state, and service performance/quality. The execution of a service request adds workload to system resources. Security activities may be added to protect the system from cyber attacks. These activities also require system resources to fulfill its mission. Cyber attacks themselves can be represented as additional system activities launched by malicious users with the purpose to compromise system resources, services and security. Services, security mechanisms and attacks drive system activities which change the workload/state of system resources. Changes in the workload/state of system resources affect the performance of services (Ye, 2002; Ye, Newman and Farley, 2005; Ye, 2008). System impacts of services, security mechanism and attacks in the form of activity-state-performance chains are not well understood at system scale, especially under services, security mechanisms and attacks simultaneously. Such cause-effect chains are not readily available from the design of system and application software which provides mostly algorithm-based operational models.

Previous studies on resources workload and service performance impacts (Vazhkudai and Schopf 2002; Doyle, et al. 2003; Shivam, Babu and Chase 2006; Sun and Ifeachor 2006; Kan, Sun and Ifeachor 2010; Kang and Suh 2011; Zhang,

Verma and Cheng 2011) address particular services or specific resources, covering limited system aspects. Hence, it is necessary to investigate the cause-effect chain of system activities, resources workload/state and service performance/quality (Ye, Yau, et al. 2010) driven by services, security mechanisms, cyber attacks, and their parameters. Based on the analytical results on system impacts of services, security mechanisms and cyber attacks, implications of those impacts in developing strategies for system survivability can be explored.

3.2 Shortcomings

Based on the above literature review, shortcomings from existing research can be summarized as follows:

1) The general approach of reserving additional system resources for survivability in case of unforeseen events is costly and impractical.
2) System adaptation decisions require understanding the impacts of services, security mechanisms and cyber attacks on resources workload/state (S), and service performance/quality (Q), but these activities (A) - workload/state (S) - performance/quality (Q) chains are not well understood at the system scale, especially under services, security mechanisms and cyber attacks simultaneously.

3.3 Objectives

Address the above shortcomings by using the empirical method proposed in chapter 2 to investigate the impacts of services, security mechanisms and cyber attacks on resources workload/state (S) and service performance/quality (Q).

Use the results on the impacts of services, security mechanisms and cyber attacks to identify tradeoffs within the limits of system resources and develop general/specific strategies for system survivability.

3.4 Methodology of data collection and analysis

The method used involves the collection of system-wide dynamics data and the application of statistical analyses to uncover resources workload and service performance. The method is fully described in section 2.4.

3.5 Description of experimental scenarios

The experimental scenarios involve two specific services (voice communication and motion detection), two security mechanisms (data encryption and intrusion detection), and five cyber attacks (ARP Poison, ping flood, vulnerability scan, fork bomb, and remote dictionary). Two sets of experiments are run. One set of experiments involves the voice communication service, data encryption as security mechanism to protect voice data transmitted over the network, and five cyber attacks. Another set of experiments involves the motion detection service, intrusion detection as security mechanism to protect the system

running the motion detection service, and five cyber attacks. Figure 3 shows the computer and network set-up for the two sets of experiments consisting of one server, five clients and the attacker. Each computer has an Intel processor Pentium 4 2.2 GHz, 1 GB memory and Windows XP operating system with service pack 2 (SP2). The computer and network set-up stands alone without any other network connections to avoid interferences.

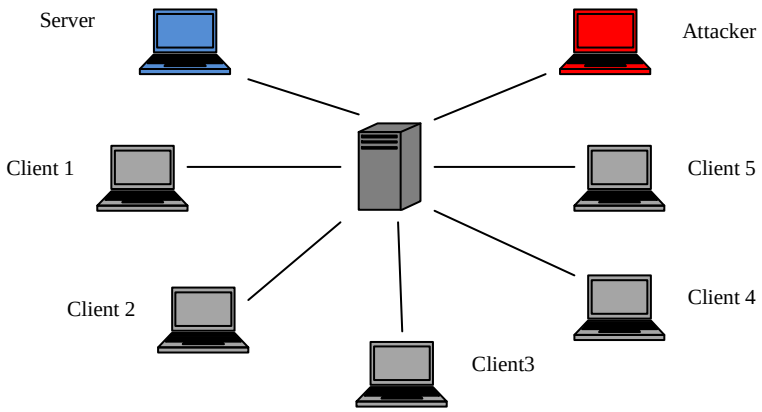


Figure 3. Computer and network set-up for services, security mechanism and cyber attacks experiments.

3.5.1 VCS, data encryption and cyber attack scenarios

Voice communication is a communication-intensive service. In voice communication, a client sends a service request to the server via the network, and the server sends the requested voice data to the client. There are one server and up to five clients, each running on its own computer. Voice communication is implemented by converting an open-source video conference software package (Abdel-qader 2007) into a web service using C# in .NET and Internet Information Service (IIS) version 6. The data encryption service uses the Advanced

Encryption Standard (AES) algorithm developed by Daemen and Rijmen (2001). The data encryption is paired with voice communication to protect the confidentiality of voice data transmitted over the network. Data encryption is implemented within voice communication software. If requested, voice data is encrypted on the server before the voice data is transmitted over the network to a client. Each attack (*A*) in the experiments is launched against the server computer. ARP poison is a man-in-the-middle attack that corrupts the content of the ARP table on the server computer. Cain and Abel® v4.9.30 is used to perform the ARP Poison attack. Ping flood is a denial of service (DOS) attack that quickly fills up network resources for holding network connections. Ping ® v2.0 is used to perform the attack. Vulnerability scan is an attack that searches for system vulnerabilities by scanning the network open ports. Nmap ® v4.76 is used to perform the attack. Fork Bomb is also a DOS attack that keeps creating processes/threads and thus fills up system resources for holding processes/threads. Remote dictionary is a brute-force attack that keeps trying different user names and passwords to gain access to the administrator account on a computer via the Windows desktop connection utility. Tscrack® v2.1 is used to perform the remote dictionary attack.

Voice communication has three service parameters: 1) the sampling rate (*Sa*) which determines the quality of the sampled voice data, 2) the number of clients (*C*) requesting the service, and 3) the size of the buffer (*B*) holding the sampled voice data at the server before transmission. The parameters for data

encryption are: 1) the encryption percentage (*E*) which is the percentage of packets encrypted, and 2) the key length (*K*) which is the size of the key used for encryption. Table 12 defines the levels of each of the parameters used in the experiment. 486 experimental conditions for 3*3*3*3*3*2 combinations of levels for *Sa*, *C*, *B*, *E*, *K* and *A* are run. For each attack (*A*), 486 experimental conditions are run in a random order, and then run again in a reverse order after cleaning up and restarting the server. System dynamics data from both orders of experimental runs is used for data analyses so a particular order of running the experimental conditions does not affect the analysis results.

Table 12. Parameters levels for VCS, data encryption and cyber attacks.

Service Parameters	**Level 1**	**Level 2**	**Level 3**
Sampling rate (*Sa*)	44,100Hz	132,300Hz	220,500Hz
Number of Clients (*C*)	1	3	5
Buffer size (*B*)	16Kbytes	32Kbytes	48Kbytes
Encryption Percentage (*E*)	0%	50%	100%
Key Length (*K*)	128 bits	192 bits	256 bits
Cyber Attack (*A*)	no attack	attack	

3.5.2 MDS, intrusion detection and cyber attack scenarios

Motion detection is a computation-intensive service. In motion detection service (MDS), clients send service requests to the server to analyze video streams to detect motion. To focus on the computation-intensive aspect of motion detection, pre-recorded video files (each file with a different video resolution) stored on the server computer are used instead of having video data transmitted over the network. When a client requests the motion detection service, a video file

with a specified video resolution is opened and processed frame by frame at the rate of 20 frames per second to simulate real-time video streaming from peripheral devices such as a webcam. Video data is analyzed using a motion detection algorithm which is implemented by converting an open-source motion detection algorithm package (Kirillov 2007) into a web service using C# in .NET and Internet Information Service (IIS) version 6. The detection algorithm first extracts a reference frame from the initial frames of a video stream, and then calculates differences between the subsequent frames and the reference frame. Multiple clients can simultaneously request the server to process a video stream with a specified video resolution to detect whether there is any motion. A process thread is created for each client. Snort® is used as the network intrusion detection software in the experiments. The intrusion detection is run independently from the motion detection service. The same five cyber attacks (ARP Poison, ping flood, vulnerability scan, fork bomb, and remote dictionary) are also run in this set of the experiments. Motion detection has two parameters: the video resolution (*R*) and the number of clients (*C*). Table 13 defines the levels of each parameter. Totally 36 experimental conditions (3*3*2*2 combinations of levels for *R*, *C*, intrusion detection, and attack) are run. For each attack, the 36 experimental conditions are run in a random order, and then run again in reverse order after cleaning up and restarting the server. System dynamics data from both orders of experimental runs is used for data analyses so a particular order of running the experimental conditions does not affect the analysis results.

Table 13. Parameters levels for MDS, intrusion detection and cyber attacks.

Service parameters	**Level 1**	**Level 2**	**Level 3**
Video resolution (R)	22×18	44×36	88×72
Number of Clients (C)	1	3	5
Intrusion Detection (I)	no I	I	
Cyber Attack (A)	no A	A	

3.5.3 System dynamics data collection

For the two sets of experiments, Windows performance objects (Microsoft 2009) are used to collect system dynamics data from the server computer. The data collected reflects system activities, resources workload/state and service performance/quality. Fifteen Windows performance objects, including Process, Processor, Memory, Paging File, Physical Disk, IP, UDP, TCP, Redirector, Network, Server, Web Services, System, Objects, and Terminal Service Session (TSS) are collected. Each object has a number of variables that provide information of activities, state and performance of system resources. The activities, state and performance monitored by each object are described below.

- Process: monitor running application programs and system processes.
- Processor: monitor various aspects of the processor activities, state and performance.
- Memory: monitors behavior of physical memory (RAM) and virtual memory (including space in physical memory and on disk), especially movement of pages between disk and physical memory.

- Paging File: monitors paging to retrieve data from disk devices to memory.
- Physical Disk: monitors read and write activities, state and performance of hard disk drives.
- IP: monitors received and sent datagrams at the IP layer and various IP errors.
- UDP: monitors received and sent datagrams through UDP (User Data Protocol) and UDP errors.
- TCP: monitors received and sent data segments through TCP (Transmission Control Protocol) and TCP errors.
- Network: monitors received and sent data at the network layer.
- Redirector: monitors the handling of application requests for network connections originating at the computer by the redirector which redirects application data between network layers.
- Server: monitors communication between the computer and the network.
- Web Services: monitors file transfer rates, bandwidth usage, connections and errors through the Internet Information Services (IIS).
- System: monitor the overall activities of system components including processes, threads, system calls, context switches for the processor, memory, file operations, etc.
- Objects: monitors logical objects in the system, including processes, threads, events, etc.

- Terminal Service Session: provides per-session statistics of system activities, resource and performance.

There are 384 system dynamics variables from these 15 Windows performance objects. Thirty data observations of these variables are collected for each experimental condition at a rate of one observation collected per second. Additionally, two service performance metrics for the MDS are collected: processing delay and motion level. The motion level is computed as follows: Motion level = Number of detected changed pixels / Total number of pixels. The processing delay is computed as the delay of processing each frame from a video stream. The code for computing the motion level and the processing delay was added to the motion detection software.

3.6 Results and discussions

In this section the system impacts characteristics of voice communication, data encryption, intrusion detection, motion detection, and cyber attacks are presented.

3.6.1 System impacts of VCS, data encryption and cyber attacks

Table 14 presents system impacts of voice communication, data encryption and cyber attacks on system activities, resources workload/state and service performance/quality in major groups with each group of system dynamics variables showing similar impacts with voice communication, data encryption and attack parameters. For each group, the system impact characteristics, competition

of voice communication, data encryption and attacks for system resources, and selection of Windows performance objects for monitoring system impacts are discussed. The implications of these impacts for system survivability and attack detection strategies are also discussed.

Table 14. System impacts of VCS, data encryption and cyber attacks.

Group of System Impacts	**Object**	**Variables**
1. $A\downarrow E\uparrow Sa\uparrow C\uparrow$ (17 variables)	Process (9 variables)	*Activity variables:* Page Faults/sec ($K\downarrow$), IO Read Operations/sec ($K\downarrow$), IO Write Operations/sec ($K\downarrow$), IO Data Operations/sec ($K\downarrow$), IO Other Operations/sec (K↓), IO Read Bytes/sec ($K\downarrow$), IO Write Bytes/sec ($K\downarrow$), IO Data Bytes/sec ($K\downarrow$). *State variable:* %Privileged Time ($K\downarrow$).
	System (3 variables)	*Activity variables:* File Read Bytes/sec ($K\downarrow$), File Write Bytes/sec (K↓), File Control Bytes/sec ($K\downarrow$).
	Physical Disk (2 variables)	*Activity variables:* Avg. Disk Bytes/Transfer, Avg. Disk Bytes/Write.
	Web Service (2 variables)	*Activity variables:* Post Requests/sec ($K\downarrow$), ISAPI Extension Requests/sec ($K\downarrow$).
	TSS (1 variable)	*Activity variable:* Page Faults/sec.
2. $A\downarrow E\downarrow K\downarrow Sa\uparrow C\uparrow$ (1 variable)	IP (1 variable)	*Performance (Q) variable:* Fragments Created/sec.
3. $A\downarrow E v K v$ (16 variables)	TSS (4 variables)	*State variables:* Working Set ($Sa\Lambda C\uparrow$), Page File Bytes ($Sa\Lambda C\uparrow$), Private Bytes ($Sa\Lambda C\uparrow$), Virtual Bytes ($Sa\Lambda C\uparrow$).
	Memory (2 variables)	*State variables:* Committed Bytes ($Sa\Lambda C\uparrow$), Pool Paged Bytes.
	Objects (5 variables)	*Activity variables:* Processes, Threads ($Sa\Lambda C\uparrow$), Events ($Sa\Lambda C\uparrow$), Mutexes ($Sa\Lambda C\uparrow$), Semaphores ($Sa\Lambda C\uparrow$).
	Process (4 variables)	*State variables:* Virtual Bytes, Working Set, Page File Bytes, Private Bytes.
	System (1 variable)	*Activity variable:* Processes ($Sa\Lambda C\uparrow$).
4. $A\downarrow$ (1 variable)	Memory (1 variable)	*State variable:* System Code Resident Bytes.
5. $A\uparrow E\downarrow K\downarrow Sa\downarrow C\downarrow$ (5 variables)	System (3 variables)	*Activity variables:* File Read Operations/sec, File Write Operations/sec, File Data Operations/sec.
	Network (2 variables)	*Activity variables:* Bytes Total/sec, Packets/sec.
6. $A\uparrow E\uparrow Sa\uparrow C\uparrow$ (2 variables)	Web Service (1 variable)	*Activity variable:* Current ISAPI Extension Requests.
	Memory (1 variable)	*Activity variable:* Cache Faults/sec.

7. *A*↑ (except Fork Bomb) (3 variables)	Network (1 variable)	*Activity variable:* Packets Received/sec.
	Server (1 variable)	*Activity variable:* Pool Nonpaged Bytes.
	System (1 variable)	*State variable:* %Registry Quota In Use.
8. *E*↓ (15 variables)	Physical Disk (6 variables)	*Activity variables:* Disk Transfers/sec, Disk Writes/sec, Disk Write Bytes/sec, Transition Faults/sec, Write Copies/sec. *State variables:* Avg. Disk Write Queue Length.
	Web Service (5 variables)	*Activity variables:* Bytes Sent/sec, Files Sent/sec, Files/sec, Bytes Received/sec, Bytes Total/sec.
	Process (1 variable)	*Activity variable:* Handle Count.
	IP (1 variable)	*Activity variable:* Fragmented Datagrams/sec.
	UDP (1 variable)	*Activity variable:* Datagram/sec.
	Processor (1 variable)	*State variable:* %DPC Time.
9. *E*↑ (9 variables)	Memory (3 variables)	*Activity variables:* Pages/sec, Pages Input/sec, Page Reads/sec.
	Physical Disk (3 variables)	*Activity variables:* Avg. Disk sec/Transfer, Avg. Disk sec/Write. *State variable:* Current Disk Queue Length.
	TSS (2 variables)	*State variables:* %Processor Time, Pool Nonpaged Bytes.
	System (1 variable)	*State variable:* Processor Queue Length.

Group 1, *A*↓*E*↑*Sa*↑*C*↑(*K*↓): Most variables in this group reflect IO activities (including file, network and device IOs), page faults generated by VCS and data encryption, and bytes for read, write and control operations on files representing disks, serial and parallel devices in the system. These variables increase their values with *Sa*, *C* and *E*. Hence, VCS and data encryption increase IO activities, page faults and bytes for file operations. These variables decrease their values with *K* because the use of a larger key length increases the computation time of data encryption, leaving less CPU time for file operations of VCS and data encryption. % Privileged Time for VCS and data encryption services increase with more VCS and data encryption activities due to more CPU

time spent in the privileged mode for scheduling and synchronizing system activities. File Control Bytes/sec of the System object also increase with more system activities scheduling and synchronization. All the system dynamics variables in this group decrease with *A* because cyber attacks consume CPU time, leaving less CPU time for VCS and data encryption activities.

System impact characteristics of voice communication and data encryption: VCS and data encryption activities increase with a higher sampling rate, more clients and more percentage of data encryption, thus causing an increase in IO activities, page faults, and bytes for file operations in the system. However, a larger key length used for data encryption increases the computation and reduces the IO aspect of VCS and data encryption due to limited CPU time.

*Competition for system resource*s: VCS, data encryption and cyber attacks compete for limited CPU time. The presence of cyber attacks reduces CPU time available for VCS and data encryption.

Selection of Windows performance objects for monitoring system impacts: Physical Disk and Terminal Service Session objects provide similar information about bytes for file operations and page faults which are covered by the Process and System objects. Hence, with the Process and System objects, the use of the Physical Disk and Terminal Service Session objects are not necessary.

Implications for system survivability: the competition for limited CPU time among cyber attacks, VCS and data encryption can be used to suppress the

level and system impacts of cyber attacks and sustain CPU time for VCS and data encryption by increasing the system activities of VCS and data encryption. The system activities of VCS and data encryption can be increased, for example, by increasing the sampling rate and the encryption percentage. The increased system activities of VCS and data encryption demands and takes more CPU time, thus leaving less CPU time to be taken by cyber attacks.

Group 2, $A{\downarrow}E{\downarrow}K{\downarrow}Sa{\uparrow}C{\uparrow}$: The variable Fragments Created/sec of the IP object measures the network throughput of VCS. The VCS throughput increases with *Sa* and *C*. However, the network throughput of VCS decreases with the increasing of *E*, *K* and *A*, as data encryption and cyber attack activities competes with VCS for CPU time.

System impact characteristics of voice communication: the increasing level of VCS due to a higher sampling rate and more clients results in more network throughput. The network throughput is a major performance metric for VCS.

Competition for system resources: same as those for group 1. The throughput performance of VCS is degraded by adding data encryption.

Selection of Windows performance objects for monitoring system impacts: IP datagrams need to be created for sending out data over the network. Hence, Fragments Created/sec at the IP layer can be used to measure the network throughput of VCS.

Implications for system survivability: since data encryption decreases the network throughput of voice communication, reducing the level of data encryption through less encryption percentage and a smaller key length may be necessary when a cyber attack is present and the network throughput of VCS needs to be maintained at a certain level.

Group 3, $A{\downarrow}E\nu K\nu(Sa\Lambda C{\uparrow})$: The workload/state variables in this group indicate memory usage, and the activity variables indicate processes/threads. Memory usage and the number of active processes/threads increase with *C* and the increase of *Sa* from level 1 to level 2. Because both memory and processes/threads running in the system have a limit in the system, the memory usage and processes/threads can reach their limit as *Sa* further increases to level 3, causing the memory usage and processes/threads to stop increasing and start decreasing. Hence, *Sa* has a major impact on memory usage and processes/threads than *C*. For data encryption, as *E* and *K* increase from level 1 to level 2, processes/threads decrease because more computation for data encryption takes more CPU time. Fewer processes/threads for data encryption lead to less memory usage. As computation for data encryption further increases with *E* and *K* going from level 2 to level 3, more processes/threads are created to handle the computation demand, thus increasing memory usage. Due to competition for CPU time among cyber attacks, VCS and data encryption, the presence of cyber attacks reduces memory usage and the number of processes/threads of VCS and data encryption.

System impact characteristics of voice communication and data encryption: VCS increases memory usage and processes/threads. Increasing *Sa* places higher workloads on memory and processes/threads than *C*. Data encryption uses memory and creates processes/threads. Setting an appropriate level of *E* and *K* optimize memory and processes/threads workloads due to data encryption.

Competition for system resources: same as those for group 1.

Selection of Windows performance objects for monitoring system impacts: the System and Process objects, which cover information in group 1, also cover the information in group 3 about processes/threads and memory usage.

Implications for system survivability: same as those for group 1.

Group 4, *A*↓: System code resident bytes of the Memory object in this group shows the size of operating system code currently in physical memory that can be written to disk when not in use. Such code is for managing application processes, and is reduced by cyber attacks since cyber attacks reduce activities of VCS and data encryption. Operation system code is there for managing application processes but does not change with the increasing activity level of VCS and data encryption.

System impact characteristics of voice communication, data encryption and cyber attacks: applications such as voice communication and data encryption

need operating system code in memory for managing applications. Cyber attacks decrease operating system code in memory for managing application.

Competition for system resources: same as those for group 1.

Selection of Windows performance objects for monitoring system impacts: the Memory object is necessary to monitor operating system code in memory.

Implications for cyber attack detection: a significant decrease in operating system code in memory can be used to detect the presence and increase of outside activities coming to the computer such as cyber attacks.

Group 5, $A\uparrow E\downarrow K\downarrow Sa\downarrow C\downarrow$: The variables in this group reflect file operations and total packets sent and received at the network layer, and increase their values with cyber attacks. Cyber attacks, except Fork bomb, occur through the network, and thus increase network activities as reflected by file operations on the network device and network packets. Fork bomb keeps creating processes/threads, and thus increase file operations to store information of new processes/threads. Although cyber attacks increase file operations due to increased network activities, cyber attacks do not increase bytes for those network-related file operations. Hence, data involved in cyber attacks are not as significant as those involved in VCS and data encryption since VCS and data encryption increase bytes for file operations as shown in group 1. Due to competition for CPU time among cyber attacks, VCS and data encryption, system dynamics variables in this

group decrease their values as the system activities of VCS and data encryption increases through the increase of *Sa, C, E* and *K*.

System impact characteristics of cyber attacks: cyber attacks increase network packets and file operations in the system but not bytes for file operations.

Competition for system resources: same as those for group 1.

Selection of Windows performance objects for monitoring system impacts: the System and Network objects are necessary to capture system impact characteristics of cyber attacks, VCS and data encryption.

Implications for system survivability: same as those for group 1.

Group 6, *A*↑*E*↑*Sa*↑*C*↑: All the system activities due to VCS, data encryption and cyber attacks increase cache faults. Current ISAPI Extension Requests of web service are also increased by system activities due to VCS, data encryption and cyber attacks because all these activities use Internet Information Services platform.

System impact characteristics of voice communication, data encryption and attacks: cache faults are increased by all the activities in the system.

Selection of Windows performance objects for monitoring system impacts: the Memory object is necessary to capture cache faults.

Implications for system survivability and attack detection: cache faults can be used to measure system activities and workload. Although some activities in the system such as stealthy attacks or malicious insider activities may not seem noticeable, they are still expected to produce system impacts in terms of increasing cache faults. Hence, they can still be caught by monitoring their system impacts on cache faults.

Group 7, *A*↑: The variables in this group measure received network packets, pool nonpaged bytes, and % registry quota in use, which are affected by cyber attacks only.

System impact characteristics of attacks: all cyber attacks, except Fork Bomb, are network-based attacks and involve network packets received by the server computer and the use of registry quota.

Selection of Windows performance objects for monitoring system impacts: the System and Network objects are necessary to capture system impacts in this group.

Implications for cyber attack detection: amounts of received network packets and use of registry quota can be used to detect network-based attacks.

Group 8, *E*↓: The variables in this group reflect network data sent by VCS and network data received due to cyber attacks. These variables decrease their values with *E* because more data encryption takes more CPU time and leaves less CPU time for voice communication and attacks.

System impact characteristics of voice communication, data encryption and attacks: similar to those covered by groups 2 and 7.

Competition for system resources: same as those for group 1.

Implications for system survivability: same as those for groups 1 and 2.

Group 9, $E\uparrow$: The variables reflect page reads from disk to memory and bytes written to disk, the number of threads in the processor queue, the queue length for disk, and processor usage. These variables increase their values with *E*. Cyber attacks (*A*) have no effects on these variables. Voice communication affects these variables but not in a consistent manner with VCS parameters. System impacts in this group are similar to those in group 1. Thus, information in group 1 can be used to cover information in this group.

System impact characteristics of data encryption: data encryption increases page faults as shown in group 1 and thus page reads from hard disk to memory. Data encryption also needs to write to disk, and uses CPU time.

Selection of Windows performance objects for monitoring system impacts: the Memory, System and Physical Disk objects cover the information in this group.

Figures 4-6 summarize the major system impacts of VCS, data encryption and cyber attacks, respectively, by illustrating the cause-effect chains of system activities, resources workload/state and services performance/quality. The system

impacts of VCS and data encryption are similar except for the impacts on memory and processes/threads workloads. Increasing the sampling rate of voice communication can be limited by memory and processes/threads constraints in the system, whereas selecting an encryption percentage and a key length for data encryption in the middle range can help reduce workloads on memory and processes/threads. Increasing both the sampling rate and the number of clients increases voice communication activities and network throughput, whereas using a large key length slows down data encryption activities. Cyber attacks in the experiments are characterized by their system impacts on increasing received network packets and file operations.

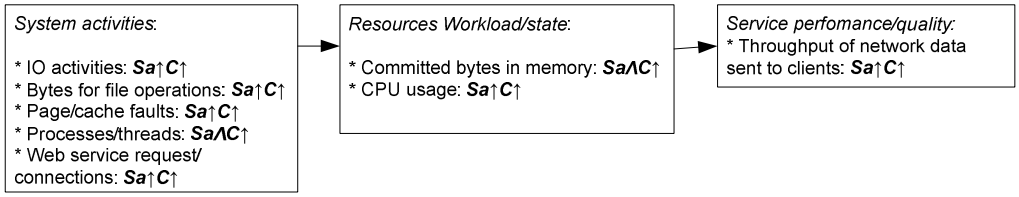


Figure 4. System impact characteristics of VCS.

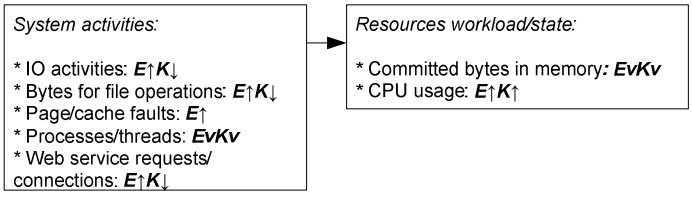


Figure 5. System impact characteristics of data encryption.

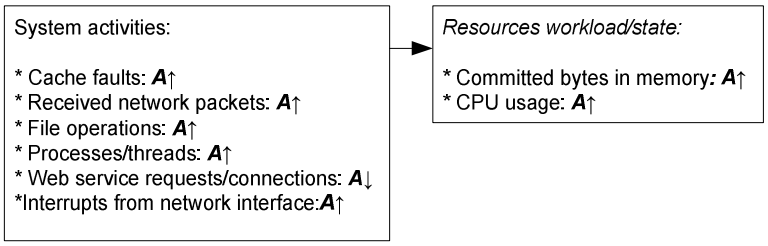


Figure 6. System impact characteristics of cyber attacks.

The competition for system resources among VCS, data encryption and cyber attacks manifests mainly in their competition for CPU time. An increase in one of the three activities decreases the two other activities. The competition for limited CPU time among cyber attacks, voice communication and data encryption can be used to suppress the level and system impacts of cyber attacks and sustain CPU time for voice communication and data encryption by increasing the system activities of VCS and data encryption. When an attack occurs and is detected, the system activities of VCS and data encryption can be increased, by increasing *Sa* to take away more CPU time from the cyber attack and sustain the performance level of VCS and data encryption. When a cyber attack is present and the network throughput of the voice communication service need to be maintained at a certain level, the data encryption service may need to be sacrificed to a certain degree. Cache faults are increased by all three activities, and can be used to indicate the overall system workloads by all activities on the system and detect stealthy, insider activities. Operating system code in memory is a useful indicator of competition between activities originating inside and outside the system. A significant decrease in operating system code in memory indicates a significant increase in outside activities coming to the system, and can be used to detect

cyber attacks. Among the fifteen Windows performance objects monitored, the System, Process, Memory, IP and Network objects cover most of system impact characteristics. These objects can be used for monitoring major system impacts of activities going on in the system.

3.6.2 System impacts of MDS, intrusion detection and cyber attacks

Table 15 presents system impacts of motion detection, intrusion detection (*I*) and cyber attacks (*A*) on system activities, resources workload/state and service performance in major groups with each group of system dynamics variables showing similar impacts with motion detection, intrusion detection and cyber attack parameters. The implications of these impacts for system survivability and attack detection strategies are discussed.

Table 15. System impacts of MDS, intrusion detection and cyber attacks

Group of System Impacts	**Object**	**Variables**
1.*C*↓*R*↓ (*A*↑) (8 variables)	Processor (1 variable)	*State variable*: % Privileged Time.
	System (6 variables)	*Activity variables*: Context Switches/sec (*A*↑), File Control Bytes/sec, File Control Operations/sec, File Write Bytes/sec, File Write Operations/sec, System Calls/sec.
	TSS (1 variable)	*State variable*: % Privileged Time (*A*↑).
2.*C*↓*R*↑ (*A*↑*I*↑) (10 variables)	Memory (8 variables)	*Activity variables*: Demand Zero Faults/sec (*I*↑), Page Faults/sec (*A*↑), Page Output/sec (*I*↑), Page Writes/sec (*I*↑), Pages/sec (*A*↑*I*↑), Page Reads/sec (*I*↑), Pages Input/sec (*I*↑), Cache Faults/sec (*A*↑*I*↑).
	Processor (1 variable)	*Activity variable*: Interrupts/sec (*A*↑).
	TSS (1 variable)	*Activity variable*: Page Faults/sec.
3.*R*↓*C*↑(*A*↓) (6 variables)	Process (4 variables)	*State variable*: % Privileged Time. *Activity variables*: IO Data Operations/sec (*A*↓), IO Other Bytes/sec (*A*↓), IO Read Operations/sec (*A*↓).

	System (2 variables)	*Activity variables*: File Read Operations/sec (A↓), File Data Operations/sec.
4. *C*↑*R*↑ (*A*↑*I*↑) (13 variables)	Performance (1 variable)	*Performance (Q) variable*: Processing Delay.
	Process (4 variables)	*Activity variables*: IO Read Bytes/sec, IO Data Bytes/sec. *State variables*: % Processor Time, % User Time.
	Processor (1 variable)	*State variable*: % User Time (*A*↑*I*↑).
	System (5 variables)	*Activity variables*: Threads (*A*↑), Processes (*A*↑*I*↑), File Read Bytes/sec, Exception Dispatches/sec (*A*↑*I*↑). *State variable*: Processor Queue Length.
	TSS (2 variables)	*State variable*: % User Time. *Activity variable*: Thread Count (*A*↑).
5. *C*↑*I*↑(*A*↑) (13 variables)	Memory (1 variable)	*State variable*: Committed Bytes (*A*↑).
	Objects (4 variables)	*Activity variables*: Processes (*A*↑), Mutexes, Semaphores, Events.
	Process (4 variables)	*State variables*: Page File Bytes (*A*↑), Private Bytes (*A*↑), Virtual Bytes (*A*↑), Working Set (*A*↑).
	TSS (4 variables)	*State variables*: Page File Bytes (*A*↑), Private Bytes (*A*↑), Virtual Bytes, Working Set (*A*↑).
6. *A*↓*C*↑(*I*↑ with *A*) (13 variables)	Memory (6 variables)	*State variables*: Cache Bytes, Pool Paged Bytes, Pool Paged Resident Bytes, System Cache Resident Bytes, Pool Nonpaged Bytes, Pool Nonpaged Bytes.
	Objects (1 variable)	*State variable*: Sections.
	Paging File (1 variable)	*State variable*: % Usage.
	Process (2 variables)	*Activity variables*: Thread Count, Handle Count.
	TSS (3 variables)	*State variables:* Pool Nonpaged Bytes, Pool Paged Bytes. *Activity variable*: Handle Count.
7. *C*↑ (6 variables)	Process (1 variable)	*Activity variable*: IO Other Operations/sec.
	Web Service (5 variables)	*Activity variables*: Current Anonymous Users, Current Connections, Current ISAPI Extension Requests, Files Sent/sec, Files/sec.
8. *R*↑ (1 variable)	Performance (1 variable)	*Performance variable*: Motion level.
9. *A*↑ (except Fork Bomb) (7 variables)	IP (4 variables)	*Activity variables*: Datagrams Received Delivered/sec, Datagrams Received/sec, Datagrams Sent/sec, Datagrams/sec.
	Processor (3 variables)	*State variable*: % Interrupt Time. *Activity variables*: DPC Rate, DPCs Queued/sec.

Note: the impact in parentheses occurs to some but not all variables in the group.

The number of clients/threads for motion detection and the video resolution produce most system impacts as follows.

More MDS clients increase and a higher video resolution increases:

- Processing delay (group 4)
- IO read and data bytes (group 4)
- CPU usage in user mode and in overall (group 4)
- Processes/threads and processor queue length (group 4).

More MDS clients increase and a higher video resolution decreases:

- IO read and data operations (group 3)
- File read and data operations (group 3).

More motion detection clients increase:

- Committed bytes in memory (groups 5 and 6)
- Connections and current ISAPI extension requests of web service (Group 7)
- IO other operations (group 7).

More MDS clients decrease and a higher video resolution increases:

- Page faults, reads and writes (group 2)
- Cache faults (group 2)
- Interrupts of processor (group 2)

More MDS clients decrease and a higher video resolution decreases:

- System calls and context switches (group 1)

- CPU usage in privileged mode (group 1)
- File write and control operations and bytes (group 1).

A higher video resolution increases:

- Motion level (group 8).

System impact characteristics of motion detection: In summary, more MDS clients competing for CPU time reduce available CPU time for each client, and thus increase the processing delay for each client. A higher video resolution improves the motion level of motion detection. Motion detection produces processes/threads, takes CPU time in user mode and memory space, and involves file and IO read operations and bytes. A higher video resolution requires processing more video data for each file and IO read operation, thus reducing the number of file and IO read operations processed per second. An increase in the number of clients and/or video resolution for motion detection increase the use of CPU time in user mode by motion detection, and leave less CPU time in privileged mode to handle system calls, context switches, and associated file write and control operations and bytes. More MDS clients decrease page faults, reads and writes and cache faults because clients use the same video stream files in the experiments. Page faults, reads and writes and cache faults are increased by a higher video resolution because a video stream file for a higher video resolution has more data contents and causes more page and cache faults to read and write such data contents. Web service connections are increased by motion detection because the motion detection software is web-based software. This may not hold

if motion detection software does not use the Internet Information Service (IIS) application. Since all the clients use the same video files, more MDS clients reduce the need for getting new data from memory to cache and from files on disk to memory, and thus reduce page faults and reads, and cache faults. However, this characteristic will not hold if MDS clients use different video files, which is likely the real case for motion detection. Figure 7 highlights the major system impact characteristics of motion detection.

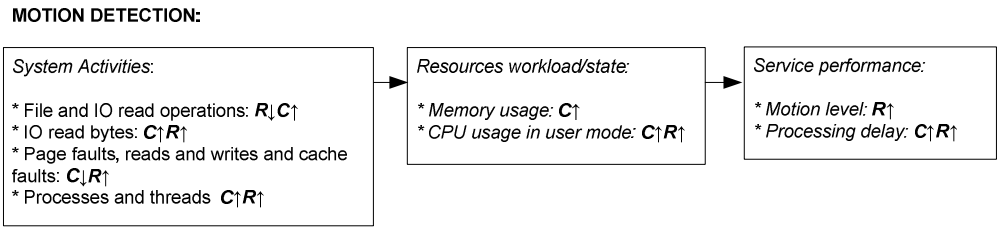


Figure 7. System impact characteristics of motion detection.

Cyber attacks have the following system impacts. Cyber attacks increase:

- Context switches (group 1)
- CPU usage in privileged mode (group 1) and in user mode (group 4)
- Interrupts from the network interface card and handles (groups 2, 6 and 9)
- Page faults (group 2)
- Cache faults (group 2)
- Memory usage (group 5)
- Processes/threads/exception dispatches (groups 4 and 5)
- Received and sent network data (group 9, excluding Fork Bomb)

Cyber attacks decrease:

- IO read, data and other operations (group 3)
- File read and data operations (group 3)
- Cache bytes (group 6).

System impact characteristics of attacks: In summary, cyber attacks (mostly network-based attacks) in the experiments increase processes/threads and thus context switches among processes/threads, interrupts from the network interface card and handles for processing those interrupts, network traffic, memory usage, page and cache faults. CPU usage increases in user mode for executing attack processes/threads and in privileged mode for handling interrupts from the network interface card. Figure 8 highlights the major system impact characteristics of cyber attacks and intrusion detection. The system impacts of cyber attacks shown in Figure 6 for VCS and in Figure 8 for MDS are consistent. Small differences in the system impacts of cyber attacks for the two set of experiments are attributed to the functional differences between the MDS which is a computation-intensive service and VCS which is a communication-intensive service.

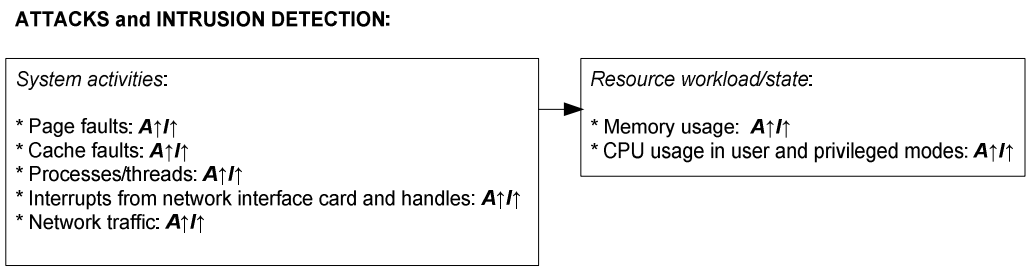


Figure 8. System impact characteristics of cyber attacks and intrusion detection.

System impact characteristics of intrusion detection: As shown in groups 2, 4, 5, 6 of Table 15 and Figure 8, the system impacts of intrusion detection are similar to those of cyber attacks because intrusion detection activities increase with the presence of cyber attacks. Note that cache faults are increased by all the activity parameters in both sets of experiments except the number of clients for motion detection due to the use of the same video files by all the motion detection clients. Hence, cache faults can be used as an indicator of all activities going on the system and thus the overall system loads.

Competition for system resources among motion detection and attacks: Cyber attacks decrease IO and file read operations and cache bytes of motion detection due to competition for CPU time between attacks and MDS.

Implications for system survivability: Similar to VCS experiments, there is also competition for system resources among MDS, cyber attacks and intrusion detection, especially CPU time. This competition can be used to suppress the level and system impacts of cyber attacks by increasing the activity level of MDS. Cache faults are increased by all the activities, therefore can be used as an indicator of the overall system workload and to detect stealthy insider activities.

Selection of Windows performance objects for monitoring system impacts: The System, Process, Memory and IP objects cover most of system impact characteristics for MDS, intrusion detection and cyber attacks.

3.7 Conclusions

Experiments were conducted to collect system dynamics data under the services of voice communication, motion detection, data encryption and intrusion detection along with cyber attacks. The analysis on the data collected from those experiments uncovers the system-wide impacts of these services and cyber attacks on system activities, resources workload/state and service performance/quality. The system impacts of voice communication and data encryption on IO activities, bytes for file operations, page faults, and processes/threads are similar except that increasing the sampling rate of voice communication can be limited by memory and processes/threads constraints in the system whereas selecting an encryption percentage and a key length for data encryption in the middle range can help reduce workloads on memory and processes/threads. Increasing both the sampling rate and the number of clients increases voice communication activities and network throughput, whereas using a large key length slows down data encryption activities. As expected, voice communication is associated with large amounts of network data sent from the server to the clients. Like voice communication and data encryption, motion detection affects file and IO operations but more on file and IO read operations. For the motion detection service the use of CPU time in user mode is more apparent than that by voice communication and data encryption services. Cyber attacks (mostly network-based attacks, except Fork Bomb) increase processes/threads and thus context switches among processes/threads, interrupts from the network interface card and handles for

processing those interrupts, network traffic, memory usage, and page and cache faults. CPU usage in user mode increases for executing attack processes/threads and CPU usage in privileged mode increases for handling the processor interrupts generated by the additional network traffic. As expected, network-based attacks are associated with an increase in network traffic to and from the server. By looking for an unexpected increase in the amount of network incoming traffic network-based attacks can be identified. Intrusion detection activities increase with the presence of cyber attacks. As a result, system impacts of intrusion detection are similar to those of the cyber attacks. Cache faults are increased by all the activities, and can be used as an indicator of the overall system workload by everything occurring in the system. The variable Cache faults/sec of the Memory object can be used to detect stealthy, insider attack activities by looking for suddenly unexpected changes in the value of the variable. The results show five Windows performance objects: System, Process, Memory, IP and Network mainly capture most of system impact characteristics. The variables in these Windows performance counters can be used to monitor the system impacts of services, thus reducing the need of collecting information from additional Windows performance objects. Although the computer and network set-up for these experiments is relatively small (1 server, 5 clients) in comparison to typical IT service-based scenarios that can have up to dozens of servers and hundreds or even thousands of clients, the system impact characteristics uncovered for these specific service scenarios as well as the selected group of variables identified to monitor system impacts of services and cyber attacks are still valid and can be

used for resource and performance management and cyber attack detection, since independent of the computer and network set-up these services, security mechanisms and cyber attacks still produce the same type of system activities and require the same type of system resources.

The competition for system resources by all the activities in the system, including voice communication, data encryption, motion detection, intrusion detection, and cyber attacks manifests dominantly in their competition for limited CPU time. This competition for limited CPU time among services and cyber attacks gives rise to a promising system survivability strategy for suppressing the level and system impacts of cyber attacks by increasing the intensity levels of services. For example, in the voice communication scenarios the intensity level of voice communication and data encryption can be increased by increasing the sampling rate and/or the encryption percentage. The increased intensity level of voice communication and data encryption demands will take more CPU time and thus leaving less CPU time to be used for cyber attacks. Moreover, when an attack is present and the performance of a service needs to be maintained at a certain level another promising strategy for system survivability involves using the uncovered tradeoffs between service performance metrics and/or services to sustain the performance of services above the required level. For example, in order to maintain the network throughput of the voice communication at a certain level, the data encryption may need to be sacrificed to a certain degree, by reducing the encryption percentage, in order to achieve survivability.

CHAPTER 4

A FRAMEWORK TO ESTIMATE SERVICE WORKLOAD AND PERFORMANCE

4.1 Background

As more organizations move their services and operations towards service oriented computing (SOC) there is an urgent need to develop service oriented architectures (SOA) and solutions for service computing to enable services provisioning by service providers to service consumers (clients) in order to satisfy their business needs (Zhang, Zhang and Cai 2007). In SOA, software applications are viewed as independent atomic services that can be dynamically selected and composed at runtime to increase system's flexibility, scalability and service's reusability. As these SOA environments grow in size and complexity, efficient management of service performance and system's resources becomes increasingly difficult (Zhang, Bivens and Rezek 2007). Previous studies have identified the value of modeling system dynamics to guide resource allocation in achieving the required service performance (Wu and Woodside 2004; Stewart and Shen 2005; Zhang, Bivens and Rezek 2007). Services compete with each other for the system's resources required to perform their intended functionality. The amount of system's resources assigned to each service will impact its performance. Therefore, efficient resource and performance in service-based systems (SBS) require understanding the dynamic effects of services on the workload/state of system's resources and service performance.

4.2 Previous Work

Much work has been done on individual workload and performance modeling for computer and network systems. Statistics and data mining techniques have been extensively used to model resource workload and performance, for example, linear regression models were used in Vazhudai and Schopf (2002) to characterize the effect of I/O workload variations on file transfer times for data grids environment. Doyle, et al. (2003) built internal-component models to predict the utilization of memory and storage resources for services with static content. Abrahao and Zhang (2004) applied principal component analysis (PCA) to characterize CPU utilization of various services in a utility computing setting. Shivam, et al. (2006) built regression models to predict the completion time of various assignments of computing, network and storage resources for batch processing tasks. Sun and Ifeachor (2006) used nonlinear regression models to predict the performance in a voice over IP (VoIP) setting by codec types under different network loads. Kan, et al. (2010) used neural networks to model video quality on wireless networks based on network state metrics.

Control theory has also been used for resource and performance management. Feedback control was used in Harada, et al. (2007) to maximize the performance of individual tasks by adjusting resource allocation. Kjaer, et al. (2009) used online feedback control to minimize CPU allocation to services while satisfying performance requirements. Kang and Suh (2011) used and heuristic

feedback control algorithm to predict delay and reliability on wireless network transmissions by adjusting the size of the error control block at the MAC layer. The disadvantage of feedback control methods is their reactive nature since changes in the environment have to propagate through the entire system before being compensated. The workload and performance models presented in these above studies cannot be generalized since they were designed for individual services, covering specific system resources or performance metrics.

General approaches to manage system resources and performance, independently of service functional and non-functional requirements have been developed. Lee, et al. (1999) proposed a mixed integer programming formulation for the multiple resource-multiple QoS problem. In this formulation, the performance (QoS) requirements for each service must be satisfied based on available systems resources. A relation between resources and performance (QoS) metrics is identified, but no description or details about the functions capturing these relations were provided. Similarly, Bashandy, et al. (2005) proposed a dynamic programming approach to solve the multiple resource-multiple QoS problem where performance (QoS) metrics are characterized as functions of system resources but the form of the functions were not defined. Zhang, et al (2007) developed an automated approach to model performance in service-based systems based on Bayesian networks. This approach incorporates existing domain knowledge into the statistical learning framework, but requires considerable amount of time for building the network model.

Queuing theory and queuing networks have been used to model performance metrics for service-base systems in Liu, et al. (2006) and Liu, Gorton and Zhu (2007). Although these models are stable and mathematically sound, assumptions required for the framework to work (e.g. scheduling algorithm, arrival and service distributions) may not be reasonable for all services. For example, Poisson distributed arrivals may not be a reasonable assumption for services under periods with high user traffic (Yu, et al. 2006).

In chapter 2, an empirical method was proposed to analyze and model the impacts of services on system activities, resources workload and service performance. This method involves the collection of system-wide dynamics data and the application of statistical analyses to uncover and model resource workload and service performance. The results show the empirical method can be used to capture the cause-effect (ASQ) relations of service-related activities (A) on resources workload/state (S) and service performance/quality (Q). However, considering the large number of possible combinations of services that can occur on a computer and network system, and thus need to be investigated, the empirical method is limited by the time and effort required for experimental set-up, data collection and analysis.

4.3 Shortcomings

Based on the above literature review, shortcomings from existing research can be summarized as follows:

1) Workload and performance models are essential to manage system's resources and performance efficiently.
2) Limited applicability of available workload and performance models from previous studies.
3) Although the empirical approach presented in chapter 2 effectively captures the cause-effects (ASQ) relations of service-related activities on resources workload and service performance, it is limited by the time and effort required for experimental set-up, data collection and analysis.

4.4 Objectives

One objective is to develop a general framework to estimate the impacts of services on resource workload and service performance under a wide variety of service conditions and independently of service functional and non-functional requirements.

Another objective is to use the framework to build the models required for resource and performance management in service-based systems (SBS).

4.5 Description of the Framework

Services require system resources such as processor (CPU), memory, disk and network to perform their intended functionality. The amount and type of resources required by each service depends on its functional and non-functional (e.g. performance) requirements (Stewart and Shen 2005; Ye, et al. 2010). Service

activities such as the number of service requests, including performance requirements, are reflected in the amount of resources required by the service.

Figure 9 presents an abstract view of the major components in a computer and network system. This figure shows the access pattern to be followed by services through multiple system resources. The main system resources are processor (CPU), memory, disk and network, but other system resources (e.g. video/sound cards) can also be included. Inter-component interaction is not considered. Each system resource has its own queue. In this framework, a model for each of the system resources is required. Each resource model should capture hardware (e.g. speed, capacity) and software (e.g. access, allocation, scheduling) characteristics of each system resource. Section 4.6 describes the details of the model development process.

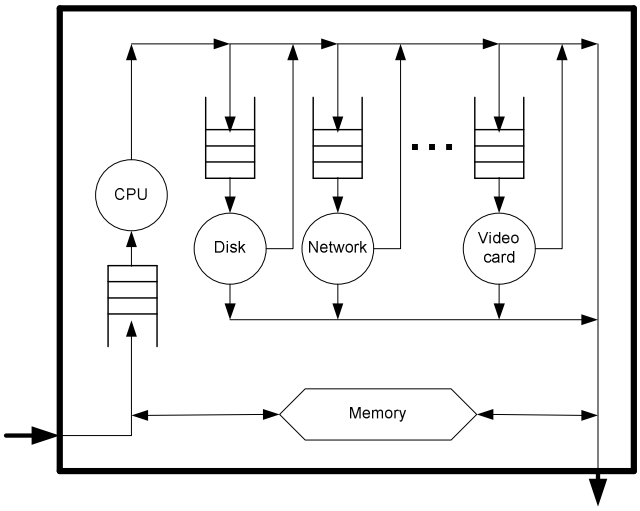


Figure 9. An abstract view of computer and network system components.

For services requiring system resources, memory is allocated first before accessing any other resource. Peak memory usage is allocated to services. Memory unlike the rest of system resources (CPU, disk and network) can be assigned to multiple services simultaneously. When memory capacity is exceeded, swapping is necessary to provide services with the memory required for execution. The effect of memory swapping on workload and performance is not intended to be modeled in this study, therefore, memory capacity is considered only as an upper bound on the number of services that can be admitted into the system. The Processor (CPU) controls the access of services to other system resources. Processor scheduling is managed by the operating system (OS). Once a service has completed execution through all resources, memory is deallocated and the service exits the system. Under this framework, system dynamics are mainly driven by: 1) the resource-sharing scheme of the system resources, including: admission control, allocation method, scheduling policy, and 2) the resource requirements (profile) of services competing for the resource.

The estimation of services workload and performance starts with the estimation of individual service workload and performance on individual resources, and proceeds to the aggregated workload and performance of these services through multiple system resources. Assuming access to system resources has been granted (admission control) and each resource can only be allocated to one service at a time (allocation method) the scheduling policy is the only aspect of the resource-sharing scheme that may affect resource workload and service

performance. Service profiles characterize per-resource needs as functions of service functional and non-functional (e.g. performance) requirements. For each resource required by the service, the arrival and the execution time distributions are specified in its service profile. The arrival distribution represents the frequency at which service instances arrive to the resource. The execution time distribution represents the amount of time the resource is required by each service instance. Arrival and execution time distributions are not limited to exponential distributions. Service profiles can be derived from application domain knowledge or obtained empirically by running experiments covering service conditions of interest to collect information regarding arrival and execution time at each resource, and then using the maximum likelihood estimation (MLE) (McLachlan and Peel 2000) method to find the best distributions representing arrival and execution time information.

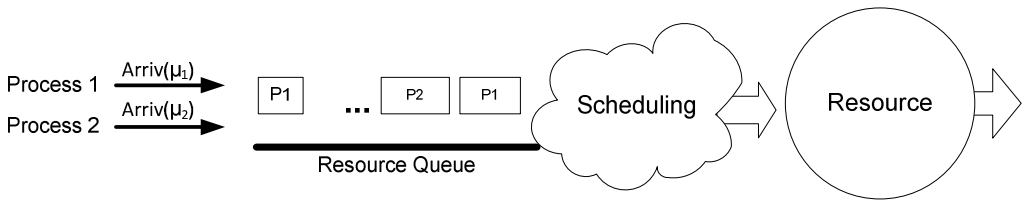


Figure 10. Competition of services' instances at a single resource.

The competition of services' instances at each single resource is shown in Figure 10. Service instances from multiple services may arrive to the resource queue according to the arrival distribution specified in their service profiles. Services' instances may have different priorities according to their type. The resource-scheduling algorithm determines how services' instances are ordered

within the queue and which service instance access the resource each time using scheduling rules such as first in-first out (FIFO), by priority, shortest job time first (SJF), earliest deadline first (EDF), round robin, etc (Silberschatz, Galvin and Gagne 2009). Rules can be preemptive, meaning an instance can be pushed out by another instance with a higher priority. Once a service instance has been assigned to the resource, the service instance seizes the resource for a time period or quantum (unless is preempted). This quantum can be equal to a fixed time (e.g. 10 milliseconds) or equal to the time required by the service instance.

If the competition of services' instances at a single resource (Figure 10) is observed over a period of time *NT*, information related to idle and busy periods of the resource such as those shown in Figure 11 can be collected.

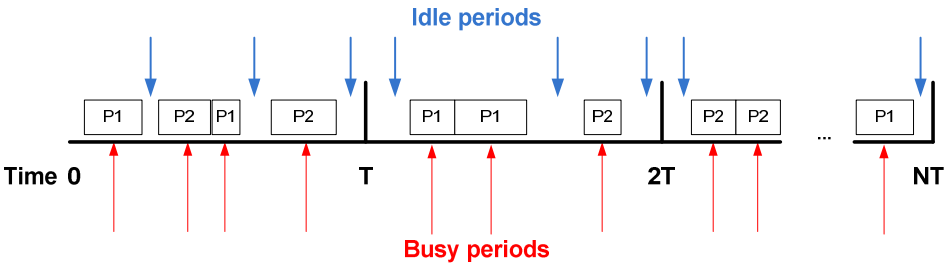


Figure 11. Information related to resource idle and busy periods during the period of *NT*.

Each busy period represents a length of time in which the resource was allocated to a particular service instance for execution. This information about Busy and idle periods together with information regarding the arrival time of service instances to the resource queue is used to estimate individual services' workload and performance at each resource. The entire observation period (*NT*),

is divided into *N* periods of length *T* and the following workload and performance metrics are estimated for each service:

1. Resource Workload (Utilization): Workload metric defined as the proportion of time *T* during period n in which the resource was busy due to instances of service type i. The resource workload at period n due to service instances of type i ($\overline{U}_i^n$) can be calculated using Eq. 1:

$$\overline{U}_i^n = {\sum_{J_i^n} Opt_{ij_i^n}^n}\Big/_{T} \qquad \text{(Eq. 1)}$$

where:

Indices:

i: Service type competing for the resource, i= *1,.., I*

n: Period number, n = *1,.., N*

j_i^n: Index of service instance of type i during period *n*, j_i^n= 1,.., J_i^n

Variables:

T: Length of time (in seconds) for each of the n periods (fixed).

$Opt_{ij_i^n}^n$: Operation time of instance j_i^n of service type i during period n.

2. Waiting Time: Defined as the average waiting time at resource queue for instances of service type i during period n. The average waiting time at period n for service instances of type i ($\overline{Wt}_i^n$) can be calculated using Eq. 2:

$$\overline{Wt}_i^n = {\sum_{J_i^n} Wt_{ij_i^n}^n}\Big/_{J_i^n} \qquad \text{(Eq. 2)}$$

where:

$Wt_{ij_i^n}^n$: Waiting time of instance j_i^n of service type i during period n.

3. Operation Time: Defined as the average operation time of instances of service type i during period n. The operation time is the sum of the execution time and the overhead time due to the management of the service instance by the resource-scheduling algorithm. The average operation time at period n for service instances of type i ($\overline{Opt}_i^n$) can be calculated using Eq. 3:

$$\overline{Opt}_i^n = {\sum_{J_i^n} Opt_{ij_i^n}^n}\Big/{J_i^n} \quad \text{(Eq. 3)}$$

4. Completion rate: Performance metric defined as the rate of instances of service type i per second that complete execution during period n. The completion rate at period n for service instances of type i ($\bar{cr}_i^n$) can be calculated using Eq. 4:

$$\bar{cr}_i^n = {\sum_{J_i^n} I(j_i^n)}\Big/{T} \quad \text{(Eq. 4)}$$

where:

$I(j_i^n)$: Binary indicator variable. It takes a value of 1 if j_i^n service instance of type i completes its execution time in the resource during period *n*, otherwise it takes a value of 0.

5. Response Time: Performance metric obtained by adding the waiting time (Eq. 2) and operation time (Eq. 3), and defined as the average time instances of service type i spend in the resource during period n.

$$\overline{rt}_i^n = \overline{Wt}_i^n + \overline{Opt}_i^n \quad \text{(Eq. 5)}$$

The metrics obtained from equations 1-5 represent workload and performance metrics for each service type at individual resources. These metrics are estimated based on the average of individual service instances observed during periods of fixed length *T*. The effect of different lengths for *T* on the workload and performance metrics is reported in section 4.8.

Once workload and performance metrics for each service type at individual resources are estimated, aggregated workload and performance metrics are obtained. The total workload on a specific resource is obtained by summing the resource workloads of all services competing for the resource (Eq. 6). The total resource workload can be compared to resource availability to identify bottleneck resources.

$$\overline{RU}^n = \sum_{i=1}^{I} \overline{U}_i^n \quad \text{(Eq. 6)}$$

The average total response time of a service is obtained by summing the average waiting time and average operation time of the service for all resources considered (Eq.7).

$$\overline{RT}_i^n = \overline{Wt}_{i(CPU)}^n + \overline{Opt}_{i(CPU)}^n + \overline{Wt}_{i(Disk)}^n + \overline{Opt}_{i(Disk)}^n + \cdots \overline{Opt}_{i(\ldots)}^n \quad \text{Or}$$

$$\overline{RT}_i^n = \bar{r}t_{i(CPU)}^n + \bar{r}t_{i(Disk)}^n + \cdots \bar{r}t_{i(\ldots)}^n \qquad \text{(Eq. 7)}$$

The total completion rate of a service is determined by the resource with the smallest completion rate (Eq.8).

$$\overline{CR}_i^n = min(cr_{i(CPU)}^n + \bar{cr}_{i(disk)}^n + \cdots \bar{cr}_{i(\ldots)}^n) \qquad \text{(Eq. 8)}$$

The total response time and total completion rate can be compared to service requests (QoS) requirements to identify if performance requirements of services are being satisfied.

4.6 Models development

The framework described in the previous section (4.5) requires the collection of information regarding: 1) idle and busy periods for each of the system resources, and 2) the detailed tracking of service instances along the system to estimate service workload and performance at each resource (Eq. 1-5) and then aggregate this information to obtain overall resources workload and service performance estimates (Eq. 6-8). In order to collect this information, models of system resources are required. Each resource model should capture hardware (e.g. speed, capacity) and software (e.g. access, allocation, scheduling) characteristics of each system resource. Since inter-component interaction is not considered, each model is viewed as an independent component. In Chapters 2 and 3, Windows performance objects (Microsoft 2009) were collected to capture

workload and performance information. Windows performance objects provide mostly aggregated values of workload and performance variables from multiple service instances rather than workload and performance values associated with individual service instances. For example, the variable % Processor Time of the Process object measures the percentage of time the processor spent executing threads (instances) of a particular service during a period of 1 second. The information provided by this variable is detailed enough for the analyses performed in Chapters 2 and 3, but it provides no information to estimate the waiting time or the operation time of individual service instances, and thus cannot be used to estimate individual service performance metrics at the resource such as the operation time and waiting time. Only aggregated workload for service instances during the period (1 second) can be estimated.

For this part of the research, two system resources are modeled: processor and disk to illustrate how the framework can be applied to estimate workload and performance of services. The details and assumptions for these models are given in sections 4.6.1-4.6.4. Models are implemented using discrete-event simulation (DEVS) formalism into ARENA v12 software. Hardware and resource-sharing scheme characteristics incorporated in the models can be configured to particular hardware and software specifications. These models can collect data regarding: 1) idle and busy periods for each of the system resources, and 2) the detailed tracking of service instances along the system. Memory resource is considered to have an upper bound on the number of services that can be admitted into the

system. Network, video card, sound card and other system resources along with interactions between components are out of the scope for this research, but they can be considered in future work.

4.6.1 Processor (CPU) model

The processor (CPU) model implements a round robin priority preemptive (RRP) scheduling algorithm. This algorithm intends to represent the processor scheduling algorithm used in Windows XP operating systems (Russinovich and Solomon 2005), although at a higher abstraction level. Important service parameters (factors) that affect processor workload and performance are: services' priority, arrival and execution time distributions, and the competition for processor with other services. Service instances from multiple services may arrive to the resource queue according to the arrival distribution specified in their service profiles. The processor time required by a service instance is based on the execution time distribution specified in the service profiles. Services' instances may have different priorities according to their type. When a service instance arrives at the resource queue, it is ordered according to its priority, if the service instance is selected to seize the processor, it seizes the processor until it is preempted by a higher-priority service instance arriving at the queue, until it terminates execution, or until its quantum ends. If quantum ends and the service instance still requires additional processor time, the service instance is sent back to the queue and the processor is assigned to the next service instance selected by the scheduling algorithm. The quantum is set to 10 ms (milliseconds)

(Silberschatz, Galvin and Gagne 2009). Overhead time (context switch) is set to 0.5 ms and represents the time required by the scheduling algorithm to un-seize the previous service instance and seize the next one. No form of priority boosting is considered. The model can be customized to represent other scheduling algorithms, parameters such as quantum and overhead can be modified, as well as the number of service types competing for the processor, their arrival and execution time distributions, and their priorities.

4.6.2 Processor model validation

In order to validate the processor model, a study was performed to compare the workload information generated with the processor model to the processor workload information collected under a real computer and network setting in Lakshminarasimhan (2005). Lakshminarasimhan (2005) used windows performance objects to collect resource workload and performance information on a server during two normal activities: text editing and web browsing. Each normal condition was run with the absence/presence of cyber attacks. The server had Windows XP® Operating system with Pentium 4 3.0 GHz processor, 3.0 GB of RAM and 120GB hard disk. The text editing condition (under no-attacks) was selected for comparison. Text editing condition was run in Lakshminarasimhan (2005) for a period of 10 minutes, for a total of 600 observations (one per second) collected for each variable of the Windows performance objects. Six services were identified to access the processor during text editing condition. The description of each of the active services is given in Table 16. At system level,

services are associated to processes. The profiles for the active services, in Table 17, were estimated by using the observations collected for the variable % Processor Time_Process() for the process associated to each service. This variable measures the percentage of time the processor was busy executing instances (threads) of a particular service during one-second intervals. The arrival distribution contained in the profile of each service represents the interval frequency at which services arrive to the processor during text editing conditions. The execution time distribution also contained in the profile of each service represents the processor time per interval required by the service instances.

Table 16. Description of active services during text editing condition.

Services	**Description**
csrss	Client/server run-time subsystem responsible for the windows console, creating and/or deleting threads, and some parts of the 16-bit virtual MS-DOS environment.
explorer	Responsible of user shell and desktop.
mmc	Microsoft Management Console application used to display management plug-ins accessed from the Control Panel, such as the Device Manager.
system	Checks the correct performance of the entire system, including drivers, ports, memory, disk and all other components.
smlogsvc	Monitor machine's performance, periodically scheduled checks on your system and create logs, notify of problems.
word	Responsible of text editing activity.

Table 17. Services profiles containing arrival and execution time distributions.

Index(*i*)	**Services**	**Arrival Time Dist. (sec)**	**Execution Time Dist. (sec)**
1	word	Norm(1.7,1.23)	Norm(2.61,1.51)
2	system	Expo(105.8)	1.5625
3	csrss	Expo(48.75)	1.5625
4	explorer	Rand*200	1.5625/6.25
5	mmc	Rand*300	1.5625
6	smlogsvc	Expo(9.5)	1.5625

The variable selected for comparison is the % Idle Time of the Processor object which measures the percentage of time the Processor was idle waiting for services to be executed. The service profiles in Table 17 were used as input parameters to the processor model. Ten simulations were run using the processor model. Each simulation was run for 100 seconds. The framework described previously was used to estimate the workload (utilization) of each service on the processor using Eq. 1. The length of the periods was set to 1 second (T=1), similar to the length of the collection interval for windows performance objects. The % Idle Time for each period n was calculated using Eq. 9.

$$\%IdleTime_n = (1 - \sum_{i=1}^{6} \bar{U}_i^n) * 100 \qquad \text{(Eq. 9)}$$

The Mann-Whitney test (Mann and Whitney 1947) was used to compare the % Idle Time estimates for each simulation run with the % Idle Time observations collected during the text editing condition in Lakshminarasimhan (2005). Table 18 shows the p-values obtained for the Mann-Whitney test using Minitab v14. For nine out of ten simulation runs, a p-value higher than 0.1 indicates the workload information obtained using the data collected during simulation runs is not significantly different to the workload information observed during the text editing condition. These results probe the processor model can produce accurate processor workload information based on the service profiles. Services profiles can be used to capture multiple services conditions independent of services functional and non-functional requirements.

Table 18. P-values Mann-Whitney test for simulation runs under text editing.

Run	Mann-Whitney test (p-values)
1	0.3863
2	0.2627
3	0.029*
4	0.7138
5	0.3368
6	0.3349
7	0.7695
8	0.6474
9	0.5905
10	0.1054

4.6.3 Disk model

Disk is considered the slowest resource in a computer and network system (Riska and Riedel 2006). Effective disk management is required to prevent disk becoming a bottleneck in system performance. Figure 12 shows the disk model structure which represents a high level abstraction of a real disk structure. Similar to a real disk, the operation time for a service instance in the disk model is the sum of the access time and the transfer time. Transfer time measures the time needed to read/write the data required and it mainly depends on the transfer rate and the data size. Access time measures the time it takes the disk head to reach the disk block (sector) required for the read/write operation. Access time has two major components: Seek time and rotational latency. Seek time measures the time it takes the read-write head to reach the track containing the required block. Rotational latency measures the time it takes to rotate the platter to reach the

specific block along the track. Access time is the major contributor to the operation time.

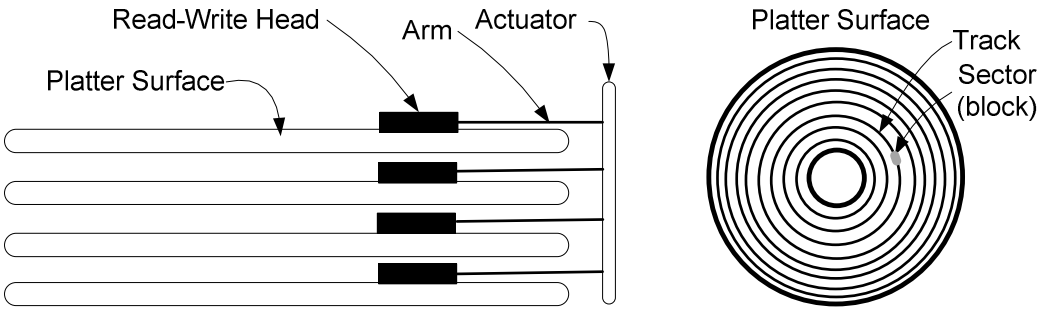


Figure 12. Disk model structure (disk abstraction)

Table 19 lists the default values for the parameters considered in the disk model. These parameters can be customized to particular disk hardware and software characteristics. The access pattern refers to the track location of the service instances (disk requests) along the platters' surface. This is an important factor that affects disk workload and performance. In general, it is accepted that servers and desktop environments operate under mostly random access patterns for the purpose of workload and performance analysis (Thomasian and Liu 2002; Riska and Riedel 2006). Other service parameters (factors) that affect disk workload and performance are: block size which directly affects transfer time, the arrival distribution of the services instances, and the competition for disk with other services.

Table 19. Default values for disk model parameters

Disk Parameters	Default values
No. of tracks per surface	10,000
Transfer rate (read/write)	300 Mb/s
Avg. rotational latency	4.16 ms
Avg. seek time	9 ms
Constant seek time	0.5 ms
Block size	4 Kb
Access pattern	random
Scheduling algorithm	C-Look

Most disk scheduling algorithms focus on minimizing access time, since access time is the main contributor to disk operation time. The C-Look (circular elevator) algorithm focus on minimizing the seek time part of the access time. C-Look sorts arriving service instances (disk requests) in its queue according to the track where the block to be read/write is located, then it starts executing service instances from the innermost track request to the outermost track request. When the outermost track request is reached, it moves back to the innermost track request and starts executing service instances moving outward again. Seek time is estimated using Eq. 10. Track distance is measure as the number of tracks the read-write head has to move to reach the desired track from the current track position. Track time is the time it takes the read-write head to move one track and is estimated based on the average track distance and the average seek time. Track time is assumed to be constant. When random access pattern to the disk is assumed, the average track distance is roughly one third of the number of tracks per surface (Jacob, Ng and Wang 2008), and track time is estimated by dividing the average seek time by the average track distance. By minimizing track distance

seek time is reduced which in turn reduces access time. Rotational latency which is the second component of access times is considered constant for each request (average rotational latency).

$$Seek\ time = Constant\ seek\ time + track\ distance * track\ time \quad \text{(Eq. 10)}$$

4.6.4 Disk model validation

For the validation of the disk model, a study was performed to compare the disk workload information generated using the disk model to the disk workload information collected during one of the experiments in Lumb, et al. (2000). In the experiment, the impact of various scheduling algorithms on disk workload (utilization) is investigated. Lumb, et al. (2000) used the DiskSim simulator (Parallel Data Lab 2011) for the experiment. This simulation environment has been previously validated against various disks from different manufacturers, including the Quantum Atlas 10K 9.1 GB disk which results are used for validation of the disk model. The input traces used for the experiment in Lumb, et al. (2000) are used as inputs for the disk model. Input traces for this and other disk experiments are available through the DiskSim website (http://www.pdl.cmu.edu/DiskSim/diskspecs.shtml). For Quantum Atlas 10K 9.1 GB disk, input traces consist of 10,000 random access requests at an approximate 2:1 ratio of reads to writes, with most requests requiring 4 Kb block size. There is 0 (zero) time between requests. Table 20 contains the Quantum Atlas 10K 9.1 GB disk basic characteristics. These characteristics were abstracted from hardware characteristics of the disk and the input trace information. Disk model parameters

(Table 19) are adjusted to incorporate these characterisitcs. The maximum queue size is fixed to 20 at all times, similar to the experiment in Lumb, et al. (2000).

Table 20. Quantum Atlas 10K 9.1 GB disk basic characteristics.

Disk Characteristics	Values
No. of data surfaces	6
No. of tracks per surface	10,042
Avg. No. of sectors per track	298
Transfer rate (read/write)	160 Mb/s
Delay before transfer (read/write)	0.1 ms
Avg. seek time	5 ms
Min-Max seek times	1.2-10.8 ms
Avg. rotational latency	3 ms
Min rotational latency	0.5 ms
Access pattern	random
Scheduling algorithm	C-Look, SSTF

Two different scheduling algorithms are used, the C-Look algorithm described in section 4.6.3, and the shortest seek time first (SSTF) algorithm. Similar to C-Look, SSTF focus on minimizing seek time. SSTF assigns the disk to the service instance (request) with the minimum seek distance regardless of direction, it scans the queue for the service instance with requested track closest to the current track where the read-write head is located. Different from the DiskSim simulator, the disk model doesn't distinguish between read/write requests and no additional overheads on disk are considered. Figure 13 shows the comparison of the disk workload information generated by the disk model using SSTF and C-Look scheduling algorithms with the disk workload information observed during the experiment in Lumb, et al. (2000).

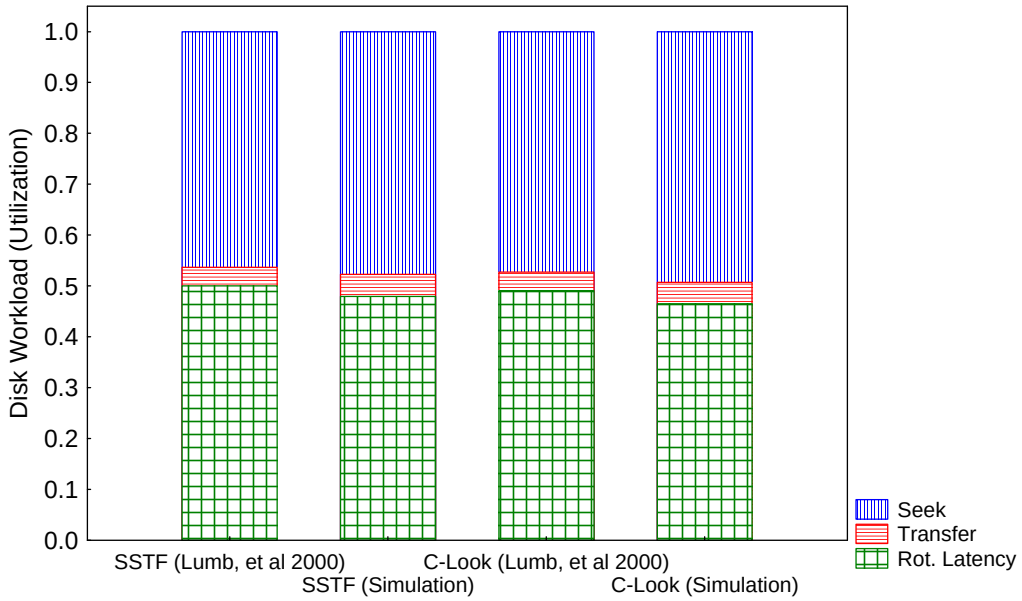


Figure 13. Disk workload comparison simulation model vs. Lumb, et al. (2000) experiment.

Disk workload information in each column is brokendown into workload due to rotational latency, workload due to data transfer, and the workload due to seek time. The first column in Figure 13 represents the disk workload information observed during the experiment in Lumb, et al. (2000) when SSTF algorithm is used. The first column can be compared to the second column containing the disk workload information obtained with the disk model when SSTF algorithm is used. Similarly, the third column contains the disk workload information observed during the experiment in Lumb, et al. (2000) when C-Look algorithm is used. The third column can be compared to the fourth column containing the disk workload information obtained with the disk model when C-Look algorithm is used. The small differences in percentage between the disk workload information generated using the disk model and the workload information observed in Lumb, et al.

(2000) for the SSTF (2.1%) and C-Look (2.6%) scheduling algorithms probe the disk model can produce representative disk workload information if the proper values for model parameters and service profiles are used. These small workload differences are mainly due to the higher abstraction level of the disk model in comparison with level of details used in DiskSim simulator.

4.7 Description of experiments for building workload and performance models

The framework described in section 4.5 together with the processor (4.6.1) and disk (4.6.3) models are used to analyze the impacts of various service parameters (e.g. arrival distribution, execution time distribution, priority, workload intensity, scheduling algorithm) on workload and performance.

Experiments were designed to build workload and performance models for each resource. The experiments cover a wide variety of service conditions. Details of the experiments are provided in the following sections 4.7.1 – 4.7.2.

4.7.1 Processor experiments

The processor experiments cover the service parameters (factors) and their levels which are shown in Table 21 and explained in the text below.

Table 21. Service parameters (factors) for processor experiments.

Service parameters	Levels
Arrival time distribution type (2)	exponential, normal
Execution time distribution type (2)	exponential, normal
No. of services in competition (3)	2, 5, 10
Scheduling algorithm (2)	RRP, MLF
Workload intensity (3)	0.5, 0.7, 0.9
Relation of execution time – service priority (2)	Small execution time – High priority, Large execution time – High priority

The distributions assumed for the arrival and execution times of services are either exponential or normal. Two, five or ten services can compete for processor during the experimental conditions (cases). The priorities of services change according to the number of services competing for the processor. If two services compete for processor, service one has higher priority than service two. If five services compete for processor, service one has the highest priority, services two and three have the second highest priority and services four and five have the lowest priority. If ten services compete for processor, services one and two have the highest priority, services three and four have the second highest priority, services five, six and seven have the third highest priority and services eight, nine and ten have the lowest priority. Two different scheduling algorithms are used, the round robin priority preemptive (RRP) algorithm with default values for the parameters described in section 4.6.1, and the multi-level feedback priority preemptive (MLF) algorithm. MLF has three different queues. The goal of this algorithm is to avoid service instances with large execution times affecting the performance of service instances with smaller execution times. Arriving services' instances are ordered according to its priority in the first queue. Service instances

in the second and third queues are also ordered according to their priority. Service instances in the first queue are executed first. If the first queue is empty, service instances in the second queue are executed. Service instances in the third queue are executed only if the first and second queues are empty. A service instance in the first queue preempts any service instance from the second or third queues. Service instances from first queue can be preempted only by arriving service instances with higher priority. The amount of time a service instance seizes the processor depend on the queue they come from. For service instances in the first queue, quantum is set to 10 ms (milliseconds) such as that in the RRP algorithm. For service instances in the second and third queues quantum is set to 20 ms and 30 ms, respectively. Once the processor has been assigned to a particular service instance, the service instance seizes the processor until the instance is preempted, until it terminates execution, or until its quantum ends. If the service instance is preempted, it goes back to the queue it was before execution, this ensures service instances will proceed to the next queue only if the processor has been allocated for at least a full quantum. If the service instance quantum ends and the service instance still requires the processor, it is sent to the next queue. Overhead time (context switch) is set to 0.5 ms and represents the time required by the scheduling algorithm to un-seize the previous service instance and seize the next one. No form of priority boosting is considered.

Workload intensity (WI_{CPU}) factor is an estimate of the processor workload due to all services competing for the processor and can be obtained using Eq. 11, a result from queueing theory (Gross and Harris 1998). Three levels

of workload intensity levels are investigated: low (0.5), medium (0.7) and high (0.9). Workload intensity tends to under-estimate processor workload since it does not consider the overhead workload due to the scheduling algorithm. The relation of execution time – service priority explores the effect of assigning the highest priorities to services with small execution times versus the effect of assigning the highest priorities to services with large execution times.

$$WI_{CPU} = \sum_{i=1}^{I} {}^{\lambda_i}/_{\mu_i} \qquad \text{(Eq. 11)}$$

where:

i = Service type, i = 1,…I λ_i = Arrival rate service type i.

μ_i = Execution rate service type i.

Based on the levels of each factor in Table 21, totally 72 x 2 = 144 experimental conditions (2*2*3*2*3*2) are run. Each experimental condition (case) is replicated 10 times. The length of each simulation run is 100 seconds. The services profiles used in each case can be found in the appendix section.

4.7.2 Disk experiments

The disk experiments were designed to cover the service parameters (factors) and their levels which are shown in Table 22 and explained in the text below.

Table 22. Service parameters (factors) for disk experiments.

Service parameters	Levels
Arrival time distribution type (2)	exponential, normal
Block Size in MB (5)	0.04, 0.016, 0.032, 0.064, 0.128
No. of services in competition (3)	2, 5, 10
Scheduling algorithm (2)	C-Look, SSTF
Workload intensity (4)	0.6, 0.8, 1, 1.2

The arrival distribution for service instances (requests) can be either exponential or normal. Service instances may require distinct block sizes to be read/write from the disk. Block sizes depend on services type. Two, five or ten services can compete for disk during experimental conditions. Two different scheduling algorithms are used, the C-Look algorithm and the shortest seek time first (SSTF) algorithm. Both algorithms focus on minimizing seek time. Default values for disk model parameters are assumed (Table 19). C-Look and SSTF algorithms do not consider services priorities. Workload intensity (WI_{CPU}) factor is an estimate of disk workload due to all services competing for the disk. Disk workload intensity is calculated using Eq. 12. The workload intensity formula (Eq. 12) considers the workload on disk due to data transfer and the workload on disk due to disk access. Disk access time is the major component of disk operation time and it is affected by the arrival rate of service instances (requests). The arrival rate of service instances is positively correlated with the number of service instances in queue, that is, if the arrival rate is increased the number of service instances in queue increases. Increasing the number of service instances in queue reduces the seek distance the read-write head has to travel between requests. Workload intensity (Eq. 12) tends to over-estimate disk workload

because it doesn't capture the reduction effect in seek distance between requests when the number of service instances in queue increases. Four levels of workload intensity levels are investigated: low (0.6), medium (0.8), medium-high (1) and high (1.2). .

$$WI_{Disk} = \sum_{i=1}^{I} \frac{\left(\frac{B_i}{Transfer\ rate} + ConstantSeek + Avg.SeekTime + Avg.RotationalLatency\right)}{\theta_i} \quad \text{(Eq. 12)}$$

where:

i = Service type, i = 1,…I θ_i = Arrival mean service type i.

B_i = Block size for service type i.

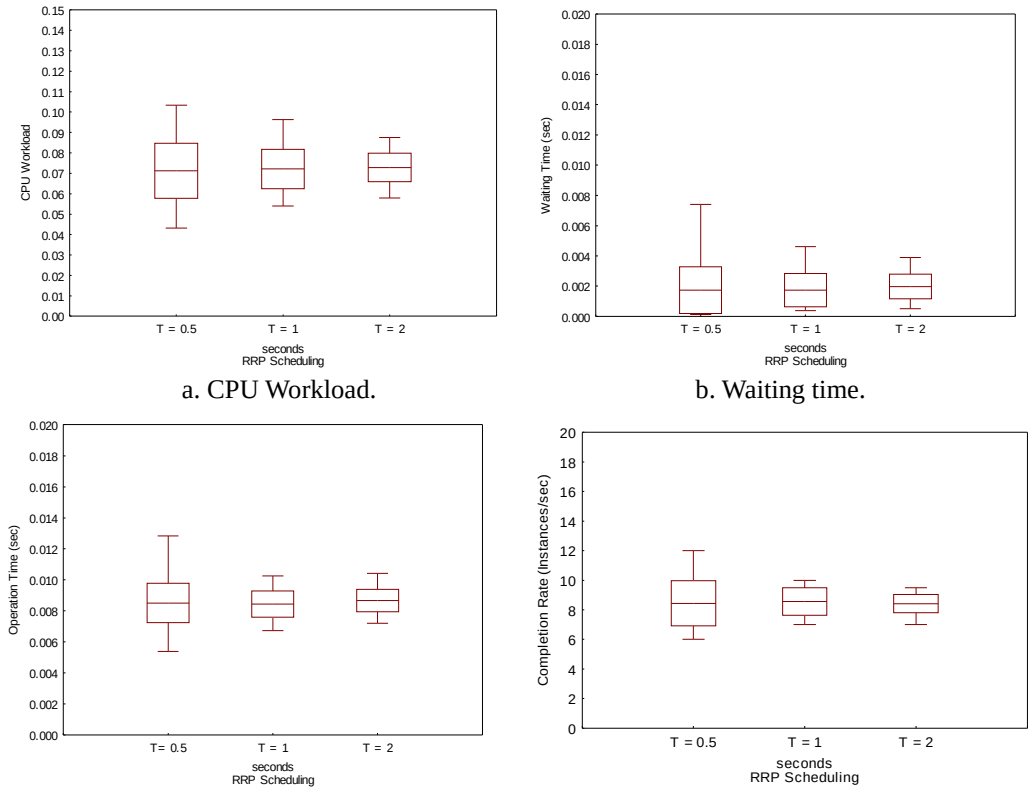
Based on the levels of each factor in Table 22, 240 experimental conditions (2*5*3*2*4) are required. However, experimental conditions (cases) can be combined given that more than one service type competes for the disk in each case. For example, when having five services competing for the disk, the five different levels of block size can be run in one experimental condition (case), combining five cases into only one. By using the same logic to combine cases, the total number of cases was reduced to 40 x 2 = 80. Each case is replicated 10 times. The length of each simulation run is 100 seconds. The services profiles used in the cases can be found in the appendix section.

4.8 Results and Discussions

In this section, the impacts of service parameters (factors) on processor and disk workload and performance are presented. Workload and performance models capturing these impacts are provided.

4.8.1 Impacts of service parameters on processor workload and performance

The framework described in section 4.5 is used to estimate workload and performance metrics of services at the processor based on the data collected from the experimental conditions (cases) in section 4.7.1. For each of the 10 simulation runs (replicates) for each case, workload and performance metrics are obtained, using equations 1-5, for each of the services competing for the processor. Figure 14 shows the effect of using different values of T when estimating the workload and performance metrics for service 1 in case 24 run 1. T is the length of the period used for calculating the workload and performance metrics. As it can be seen from Figure 14, the mean value of the workload and performance metrics estimated for service 1 of case 24 run 1 is not sensitive to the length of the period (T). Similar effect with T is observed for the workload and performance metrics of services in all cases. This effect increases the confidence for using these metrics as estimates of service workload and performance at the processor.



a. CPU Workload. b. Waiting time.

c. Operation time. d. Completion rate.

Figure 14. The effect of *T* on service workload and performance metrics at processor.

The metrics are estimated based on the average of individual service instances observed during *n* periods of length *T*, as *T* increases, the number of service instances observed in each period increases and this increase in the number of service instances observed tends to reduce the variance of the workload and performance metrics estimated for each service.

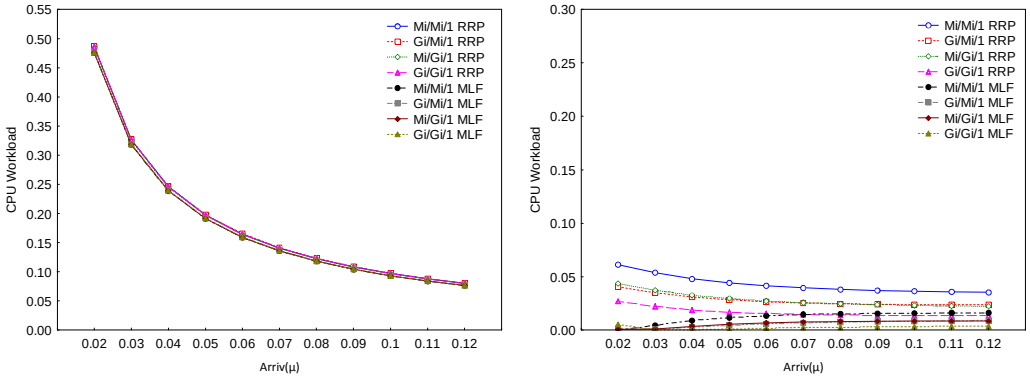
The impacts of service parameters on the mean and standard deviation of the workload and performance metrics for individual services are discussed next. The length of the period, *T*, is set to 2 seconds. Since each simulation run for each of the cases is run for 100 seconds, by setting *T* equal to 2 seconds each simulation run is divided into 50 periods and the information observed during

these periods is used to estimate the workload and performance metrics of services (Equations 1-5).There is no statistically significant impact on service workload and performance metrics with a change in the relation of execution time – service priority. To better understand the impacts of service priorities and workload intensity (WI_{CPU}) parameters on individual service workload and performance, these two parameters were combined to obtain: ρ_i, ρ_{HP} and ρ_{SP}. ρ_i is the workload intensity due to the service type i and is estimated using Eq. 11, but considering only service type i ($\rho_i = \lambda_i/\mu_i$). ρ_{HP} is the workload intensity due to the services with higher priority than service type i and is estimated using Eq. 11, but considering only service types with higher priority than service type i. ρ_{SP} is the workload intensity due to the services with similar priority to that of service type i and is estimated using Eq. 11, but considering only service types with similar priority to that of service type i.

CPU Workload

Figure 15 shows the impacts of service arrival distribution mean ($Arriv_\mu$) on processor (CPU) workload mean (μ) and standard deviation (σ). The $Arriv_\mu$ has a decrease effect on the processor workload mean (μ) and standard deviation (σ). $M_i/M_i/1$ represents those experimental conditions (cases) with arrival and execution times assuming exponential distributions. $G_i/M_i/1$ represents those cases with arrivals being normal distributed and execution time being exponential distributed. $M_i/G_i/1$ represents those cases with arrivals being exponentially distributed and execution time being normal distributed. $G_i/G_i/1$ represents those

cases with arrival and execution times assuming normal distributions. For constructing Figures 15- 27, service parameters are kept at constant values in each figure. For example for constructing Figure 15, the execution distribution mean of the service (Ex_{μ}=0.009), the workload due to higher priority services (ρ_{HP}=0) and the workload due to same priority services (ρ_{SP}=0) were kept at constant values. Each figure indicates the parameters kept at constant values for its construction.

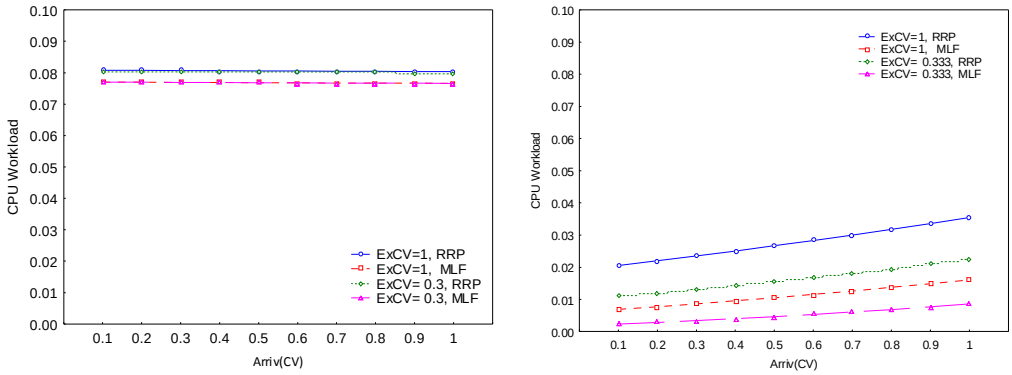


a. Impact of $Arriv_{\mu}$ on CPU Workload (μ) b. Impact of $Arriv_{\mu}$ on CPU Workload (σ)

Constant parameters: Ex_{μ}=0.009, ρ_{HP}=0, ρ_{SP}=0

Figure 15. Impacts of arrival distribution mean ($Arriv_{\mu}$) on CPU workload.

Figure 16 shows the impacts of service arrival distribution's coefficient of variation ($Arriv_{CV}$) on processor (CPU) workload mean and standard deviation. The coefficient of variation (CV) is a normalized measure of the dispersion of a probability distribution and is defined as the ratio of the standard deviation (σ) to the mean (μ). Exponential distributions has a CV = 1, and normal distributions are restricted to have a CV = 1/3. $Arriv_{CV}$ has no significant effect on the mean (μ) of processor workload, but it has an increasing effect on the standard deviation (σ) of processor workload.

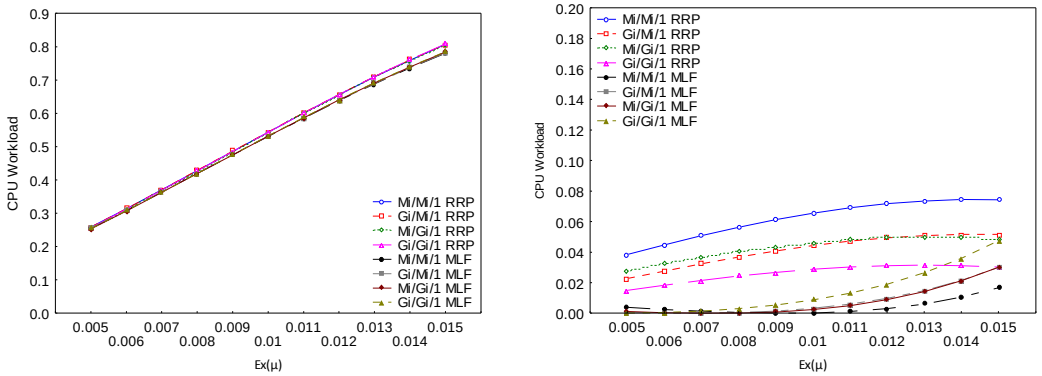


a. Impact of $Arriv_{CV}$ on CPU Workload (μ) b. Impact of $Arriv_{CV}$ on CPU Workload (σ)

Constant parameters: ρ = 0.075, $Arriv_{\mu}$=0.12, Ex_{μ}=0.009, ρ_{HP}=0, ρ_{SP}=0.

Figure 16. Impacts of arrival distribution CV ($Arriv_{CV}$) on CPU workload.

Figure 17 shows the impacts of service execution distribution mean (Ex_{μ}) on processor (CPU) workload mean (μ) and standard deviation (σ). Ex_{μ} has an increasing effect on CPU workload mean (μ) and standard deviation (σ).

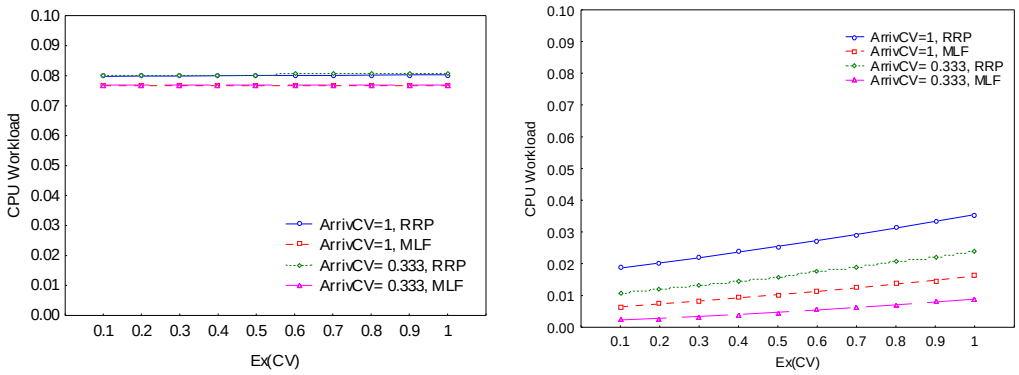


a. Impact of Ex_{μ} on CPU Workload (μ) b. Impact of Ex_{μ} on CPU Workload (σ)

Constant parameters: $Arriv_{\mu}$=0.2, ρ_{HP}=0, ρ_{SP}=0.

Figure 17. Impacts of execution distribution mean (Ex_{μ}) on CPU workload.

Figure 18 shows the impacts of service execution distribution CV (Ex_{CV}) on processor (CPU) workload mean (μ) and standard deviation (σ). Ex_{CV} has no significant effect on CPU workload mean (μ), but it has an increasing effect on CPU workload standard deviation (σ).



a. Impact of Ex_{CV} on CPU Workload (μ) b. Impact of Ex_{CV} on CPU Workload (σ)

Constant parameters: ρ = 0.075, $Arriv_{\mu}$=0.12, Ex_{μ}=0.009, ρ_{HP}=0, ρ_{SP}=0.

Figure 18. Impacts of execution distribution CV (Ex_{CV}) on CPU workload.

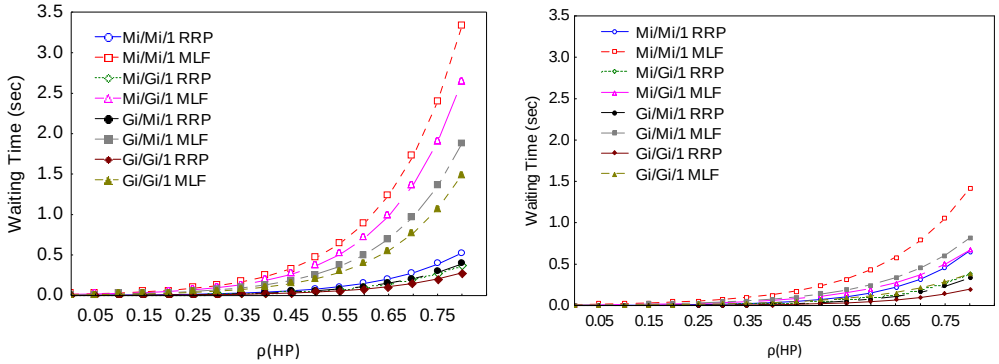
Table 23 summarizes the service parameters effects on CPU workload mean (μ) and standard deviation (σ) with the parameters: workload intensity due to higher priority services (ρ_{HP}), workload intensity due to services with similar priority (ρ_{SP}), arrival distribution mean ($Arriv_{\mu}$), arrival distribution CV ($Arriv_{CV}$), execution distribution mean (Ex_{μ}), and execution distribution CV (Ex_{CV}). From Figures 15-18, it can be observed the mean (μ) and standard deviation (σ) of the CPU workload metric depends on the arrival and execution time distributions assumed. The mean (μ) and standard deviation (σ) of the CPU workload metric tend to be higher with RRP algorithm but only the effect on standard deviation of CPU workload with RRP is statistically significant.

Table 23. Service parameters effects on CPU workload mean (μ) and standard deviation (σ).

Parameters effects on:	**ρ_{HP}**	**ρ_{SP}**	**$Arriv_{\mu}$**	**$Arriv_{CV}$**	**Ex_{μ}**	**Ex_{CV}**
CPU Workload (μ)	-	-	↓	-	↑	-
CPU Workload (σ)	-	-	↓	↑	↑	↑

Waiting Time

Figure 19 shows the increasing effects on service waiting time mean (μ) and standard deviation (σ) due to the increase of the processor workload by higher priority services (ρ_{HP}).

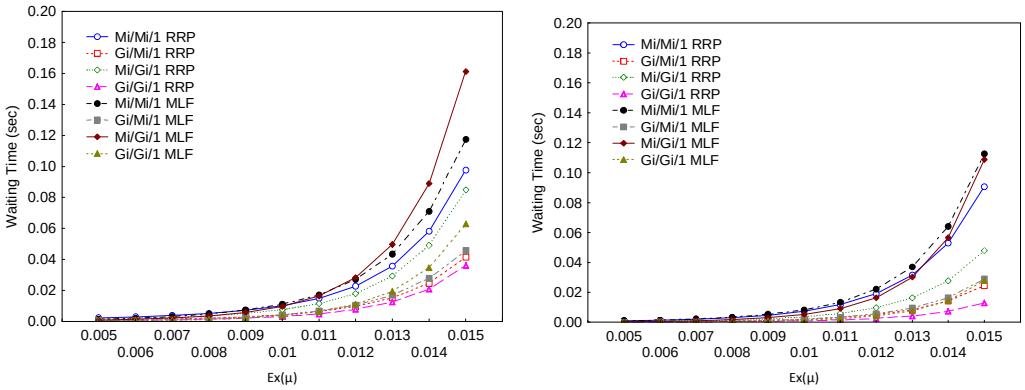


a. Impact of ρ_{HP} on Waiting Time (μ) b. Impact of ρ_{HP} on Waiting Time (σ)

Constant parameters: ρ = 0.075, $Arriv_{\mu}$=0.12, Ex_{μ}=0.009, ρ_{SP}=0.

Figure 19. Impacts of workload by higher priority services (ρ_{HP}) on Waiting Time.

Figure 20 shows the increasing effects on service waiting time mean (μ) and standard deviation (σ) due to the increase of the execution distribution mean.



a. Impact of Ex_{μ} on Waiting Time (μ) b. Impact of Ex_{μ} on Waiting Time (σ)

Constant parameters: $Arriv_{\mu}$=0.2, ρ_{HP}=0, ρ_{SP}=0.

Figure 20. Impacts of execution distribution mean (Ex_{μ}) on Waiting Time.

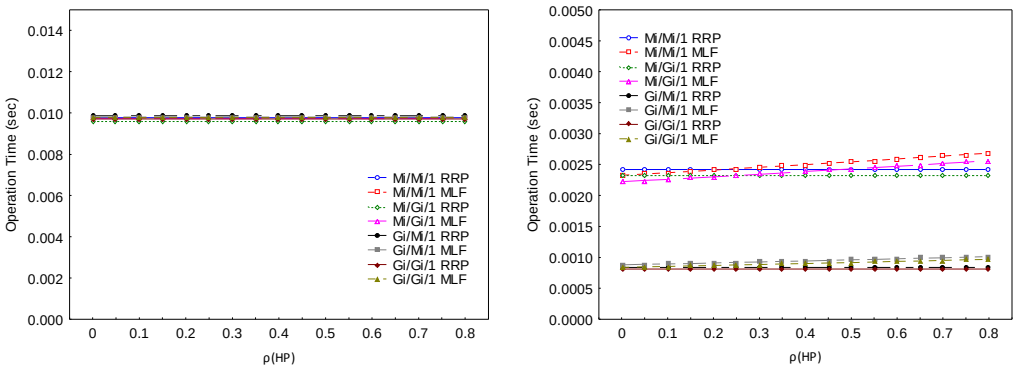
Table 24 summarizes the service parameters effects on waiting time mean (μ) and standard deviation (σ). From Figures 19-20, it can be observed that values for waiting time mean (μ) and standard deviation (σ) depend on the arrival and execution time distributions assumed and the scheduling algorithm. Waiting time mean (μ) and standard deviation (σ) are significantly larger with MLF algorithm. It is important to notice that service parameters such as ρ_{SP}, $Arriv_{\mu}$, $Arriv_{CV}$ and Ex_{CV} have an impact on waiting time mean and standard deviation, but these effects are very small in comparison with the effects of ρ_{HP} and Ex_{μ}. The models presented in section 4.8.2 capture these effects on waiting time mean and standard deviation.

Table 24. Service parameters effects on Waiting Time mean (μ) and standard deviation (σ).

Parameters effects on:	**ρ_{HP}**	**ρ_{SP}**	**$Arriv_{\mu}$**	**$Arriv_{CV}$**	**Ex_{μ}**	**Ex_{CV}**
Waiting Time (μ)	↑	-	-	-	↑	-
Waiting Time (σ)	↑	-	-	-	↑	-

Operation Time

Figure 21 shows the impacts of the workload by higher priority services (ρ_{HP}) on service operation time mean (μ) and standard deviation (σ). ρ_{HP} has no significant effect on the mean (μ) of operation time, but it has a small increasing effect on its standard deviation (σ) when MLF algorithm is used.

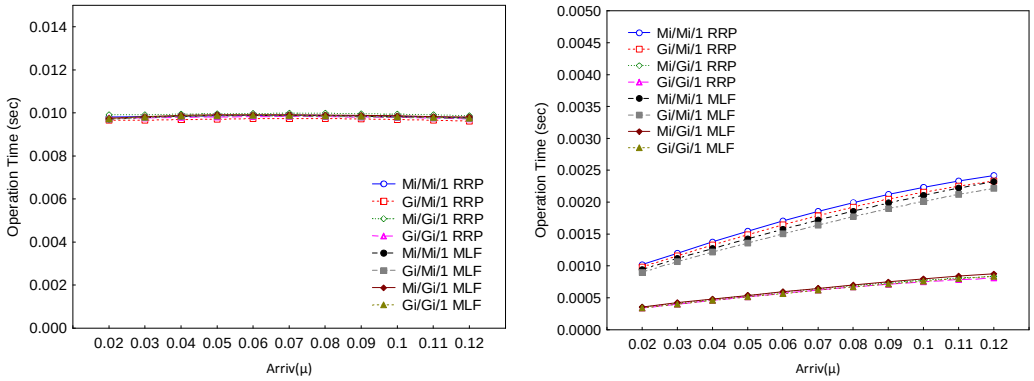


a. Impact of ρ_{HP} on Operation Time (μ) b. Impact of ρ_{HP} on Operation Time (σ)

Constant parameters: $\rho = 0.075$, $Arriv_{\mu}=0.12$, $Ex_{\mu}=0.009$, $\rho_{SP}=0$.

Figure 21. Impacts of workload by higher priority services (ρ_{HP}) on Operation Time.

Figure 22 shows the impacts of service arrival distribution mean ($Arriv_{\mu}$) on operation time mean (μ) and standard deviation (σ). $Arriv_{\mu}$ has no significant effect on operation time mean (μ), but it has a small increasing effect on operation time standard deviation (σ).

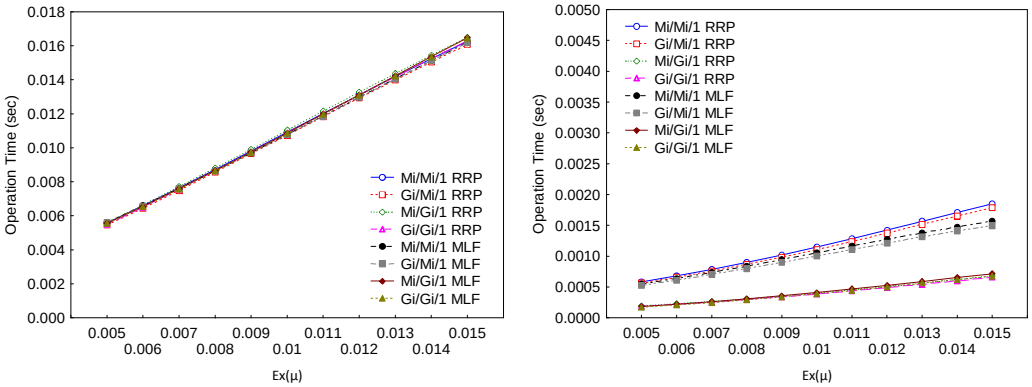


a. Impact of $Arriv_{\mu}$ on Operation Time (μ) b. Impact of $Arriv_{\mu}$ on Operation Time (σ)

Constant parameters: $Ex_{\mu}=0.009$, $\rho_{HP}=0$, $\rho_{SP}=0$.

Figure 22. Impacts of arrival distribution mean ($Arriv_{\mu}$) on Operation Time.

Figure 23 shows the increasing effect on the mean (μ) and standard deviation (σ) of service operation time due to the increase in the execution distribution mean (Ex_{μ}).

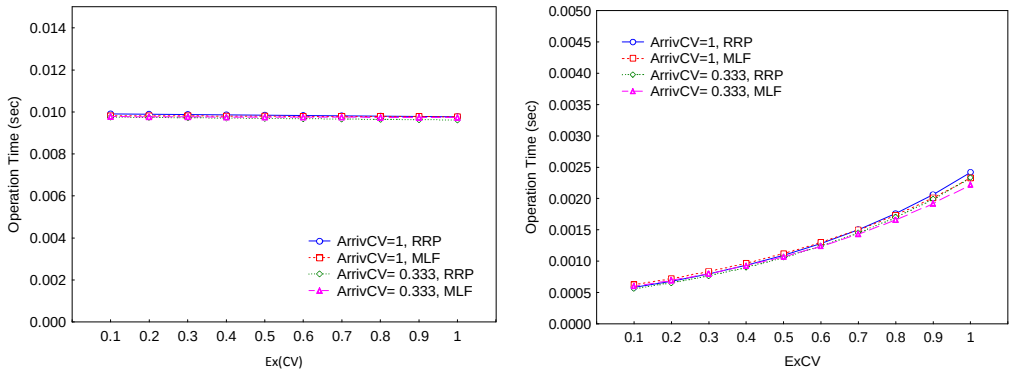


a. Impact of Ex_{μ} on Operation Time (μ) b. Impact of Ex_{μ} on Operation Time (σ)

Constant parameters: $Arriv_{\mu}$=0.2, ρ_{HP}=0, ρ_{SP}=0.

Figure 23. Impacts of execution distribution mean (Ex_{μ}) on Operation Time.

Figure 24 shows the impacts of service execution distribution CV (Ex_{CV}) on mean (μ) and standard deviation (σ) of operation time. Ex_{CV} has no significant effect on operation time mean (μ), but it has a small increasing effect on operation time standard deviation (σ).



a. Impact of Ex_{CV} on Operation Time (μ) b. Impact of Ex_{CV} on Operation Time (σ)

Constant parameters: ρ = 0.075, $Arriv_{\mu}$=0.12, Ex_{μ}=0.009, ρ_{HP}=0, ρ_{SP}=0.

Figure 24. Impacts of execution distribution CV (Ex_{CV}) on Operation Time

Table 25 summarizes the service parameters effects on operation time mean (μ) and standard deviation (σ). From Figures 21-24, it can be observed that values for the standard deviation (σ) of service operation time are affected by the

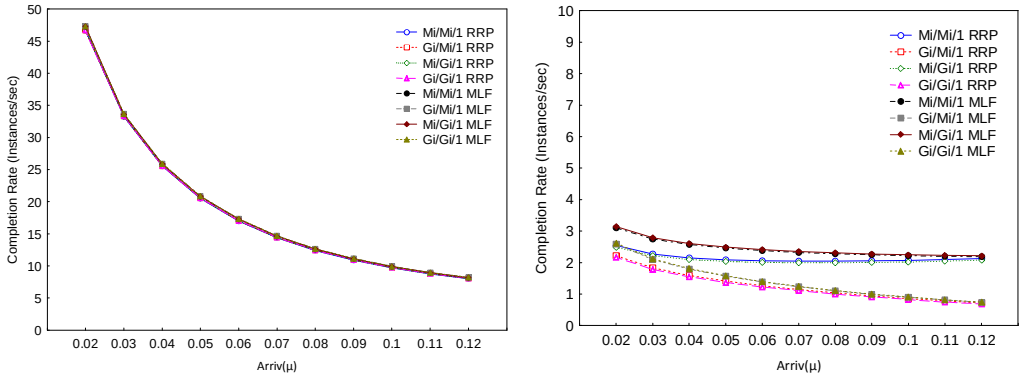
execution time distributions assumed. When the execution time distribution is exponential (CV=1) a larger standard deviation is observed than when the distribution is normal (CV=1/3). This implies the larger the CV of the execution time distribution, the larger the standard deviation (σ) on service operation time. Additionally, the standard deviation (σ) of operation time is statistically larger with MLF scheduling.

Table 25. Service parameters effects on Operation Time mean (μ) and standard deviation (σ).

Parameter Effects on:	**ρ_{HP}**	**ρ_{SP}**	**$Arriv_{\mu}$**	**$Arriv_{CV}$**	**Ex_{μ}**	**Ex_{CV}**
Operation Time (μ)	-	-	-	-	↑	-
Operation Time (σ)	↑(MLF)	-	↑	-	↑	↑

Completion Rate

Figure 25 shows the decreasing effects on completion rate mean (μ) and standard deviation (σ) due to the increase in arrival distribution mean ($Arriv_{\mu}$).

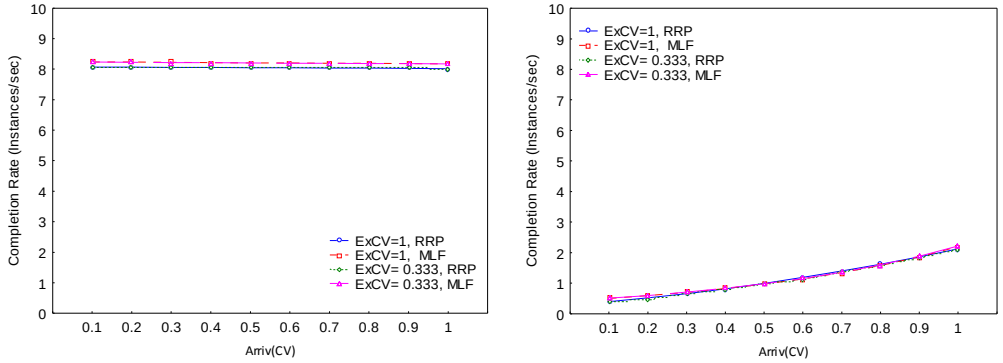


a. Impact of $Arriv_{\mu}$ on Completion Rate (μ) b. Impact of $Arriv_{\mu}$ on Completion Rate (σ)

Constant parameters: Ex_{μ}=0.009, ρ_{HP}=0, ρ_{SP}=0.

Figure 25. Impacts of arrival distribution mean ($Arriv_{\mu}$) on Completion Rate.

Figure 26 shows the impacts of service arrival distribution CV ($Arriv_{CV}$) on completion rate mean (μ) and standard deviation (σ). $Arriv_{CV}$ has no significant effect on completion rate mean (μ), but it has an increasing effect on completion rate standard deviation (σ).

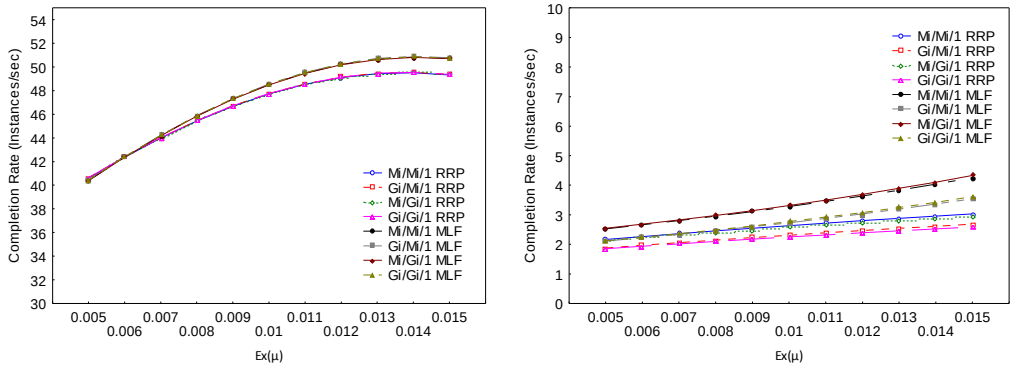


a. Impact of $Arriv_{CV}$ on Completion Rate (μ) b. Impact of $Arriv_{CV}$ on Completion Rate (σ)

Constant parameters: ρ = 0.075, $Arriv_{\mu}$=0.12, Ex_{μ}=0.009, ρ_{HP}=0, ρ_{SP}=0.

Figure 26. Impacts of arrival distribution CV ($Arriv_{CV}$) on Completion Rate.

Figure 27 shows the increasing effects on completion rate mean (μ) and standard deviation (σ) due to the increase in execution distribution mean (Ex_{μ}).



a. Impacts of Ex_{μ} on Completion Rate (μ) b. Impacts of Ex_{μ} on Completion Rate (σ)

Constant parameters: $Arriv_{\mu}$=0.2, ρ_{HP}=0, ρ_{SP}=0.

Figure 27. Impacts of execution distribution mean (Ex_{μ}) on Completion Rate.

Table 26 summarizes the service parameters effects on completion rate mean (μ) and standard deviation (σ). From Figures 25-27, it can be observed that values for completion rate mean (μ) and standard deviation (σ) tend to be slightly larger with MLF algorithm, but these effects with MLF scheduling are not statistically significant.

Table 26. Service parameters effects on Completion Rate mean (μ) and standard deviation (σ).

Parameters effects on:	ρ_{HP}	ρ_{SP}	$Arriv_{\mu}$	$Arriv_{CV}$	Ex_{μ}	Ex_{CV}
Completion Rate (μ)	-	-	↓	-	↑	-
Completion Rate (σ)	-	-	↓	↑	↑	-

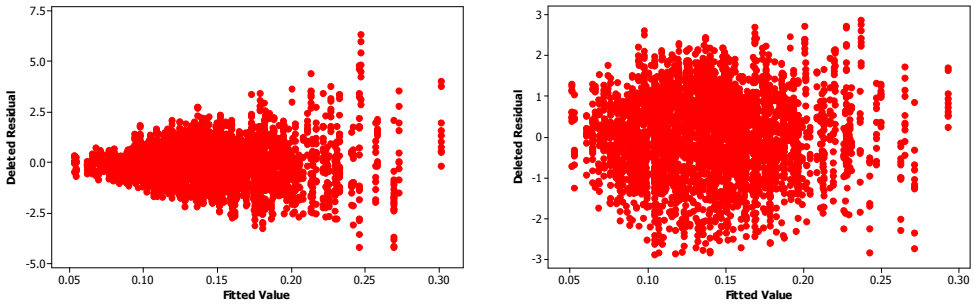
In general, the increase of workload due to higher priority services (ρ_{HP}) increases the service waiting time mean and standard deviation, and the standard deviation of service operation time when MLF scheduling is used. The arrival distribution mean impacts service workload mean and standard deviation, completion rate mean and standard deviation, and the standard deviation of operation time. Increasing arrival distribution mean decreases service workload and completion rate means and standard deviations, and increases the standard deviation of service operation time. The larger the coefficient of variation (CV) for the arrival distribution, the larger the standard deviations of service workload and completion rate metrics. An increase in the mean of the service execution distribution increases the means and standard deviations of service workload, waiting time, operation time and completion rate metrics. The larger the coefficient of variation (CV) for the execution distribution, the larger the standard

deviations of service workload and operation time metrics. Using RRP scheduling increases the standard deviation of services' workload on processor. Services, especially those with long execution times, tend to wait more time for the processor with MLF, since they stay in the lowest priority queue longer time, waiting for services in higher priority queues to complete execution.

4.8.2 Workload and performance models for processor

Tables 27-28 provide the service workload and performance models for the processor with RRP and MLF scheduling algorithms respectively. These models accurately capture the impacts of service parameters on workload and performance metrics described in the previous section (4.8.1). Multiple linear regression was used to build the models, polynomial (ρ^2, $Arriv^2_{\mu}$, Ex^2_{μ}) and interaction terms ($Arriv_{\mu.CV}$, $Ex_{\mu.CV}$) were included when necessary to increase model performance. Natural log (Ln) and square root (Sq) transformations were applied to the workload and performance metrics with the similar purpose. When analyzing the residuals for the regression models, for some of the metrics, it was found the residuals had non-constant variance. The common pattern identified in the residuals appears in Figure 27a, where the variance of the residuals increases with the fitted values. To correct this inequality of variance problem, weighted least square regression (WLS) was applied (Montgomery, Peck and Vining 2006). This approach incorporates weights into the least squares calculation. The main concern with weighted regression is how to find the proper weights to be used, but for processor and disk experiments the replicates for each of the experimental

conditions (cases) are used to estimate these weights. These replicate runs were used to calculate the average value at each service condition and its variance. The weights were defined as the inverse of the variance observed at any service condition. Figure 28 shows the residuals vs. fitted values plots for the standard deviation (σ) of service workload on processor before (Figure 27 a) and after using weighted least square regression (Figure 27 b) when RRP scheduling algorithm is used. Weighted least square regression corrected the inequality of variance of the residuals.



a. Before weighted regression b. After weighted regression

Figure 28. Deleted residuals vs. fitted values CPU workload standard deviation (σ) model.

The predictive performance of the models in Tables 27-28 was estimated in terms of the R-square. R-square is a common measure for the goodness-of-fit for regression models and measures the square correlation coefficient between the predicted and observed responses. The closest the R-square value to one, the better the fitness of the model. Ten fold cross-validation (Tan, Kumar and Steinbach 2006) is used to estimate the predictive R-square value for the models (Tables 26-27). In *k* cross-validation the data is partitioned randomly into *k* equally-sized subsets, at each of the *k* folds *k*-1 subsets (train data) are used to

build a regression model, and then this model is used to estimate the predicted response values for the remaining subset (test data). Each *k* subset is used for testing once. The predictive R-square is obtained by comparing the observed responses vs. the predicted responses obtained from cross-validation. The predictive R-sq values obtained using cross-validation show the regression models are a good fit for the data and provide the confidence to use these models for prediction of service workload and performance. The estimates obtained from these models can be used for taking workload and performance management decisions.

Table 27. Regression models for service workload and performance at processor with RRP algorithm.

Metrics	Regression model	R-sq (pred)	WLS
CPU Workload (μ)	**Sq U_{μ}** = 0.239 + 1.02 ρ - 0.367 ρ^2 - 1.61 $Arriv_{\mu}$ - 0.000853 $Arriv_{CV}$ + 4.12 $Arriv^2_{\mu}$ + 15.5 Ex_{μ} + 0.00383 Ex_{CV} - 0.304 $Ex_{\mu.CV}$ - 443 Ex^2_{μ}	0.998	-
CPU Workload (σ)	**Sq U_{σ}** = - 0.00173 + 0.306 ρ - 0.27 ρ^2 + 0.119 $Arriv_{\mu}$ + 0.0719 $Arriv_{CV}$ - 0.183 $Arriv_{\mu.CV}$ + 0.625 $Arriv^2_{\mu}$ + 5.91 Ex_{μ} + 0.024 Ex_{CV} + 3.72 $Ex_{\mu.CV}$ - 189 Ex^2_{μ}	0.921	Y
Waiting Time (μ)	**Ln Wt_{μ}** = - 8.60 - 6.91 ρ + 9.82 p^2 + 6.24 ρ_{HP} + 0.869 ρ_{SP} - 13.5 $Arriv_{\mu}$ - 7.98 $Arriv_{\mu.CV}$ + 1.38 $Arriv_{CV}$ + 57.9 $Arriv^2_{\mu}$ + 447 Ex_{μ} + 0.879 Ex_{CV} - 45.4 $Ex_{\mu.CV}$ - 8643 Ex^2_{μ}	0.915	Y
Waiting Time (σ)	**Ln Wt_{σ}** = - 11.7 - 1.73 ρ + 7.14 ρ^2 + 7.34 ρ_{HP} + 1.38 ρ_{SP} + 7.79 $Arriv_{\mu}$ + 2.20 $Arriv_{CV}$ - 11.9 $Arriv_{\mu.CV}$ + 16.1 $Arriv^2_{\mu}$ + 356 Ex_{μ} + 1.12 Ex_{CV} - 10.8 $Ex_{\mu.CV}$ - 8719 Ex^2_{μ}	0.89	Y
Operation Time (μ)	**Sq Opt_{μ}** = 0.0362 + 0.00181 ρ + 0.0455 $Arriv_{\mu}$ - 0.293 $Arriv^2_{\mu}$ + 7.98 Ex_{μ} + 0.00113 Ex_{CV} - 0.0846 $Ex_{\mu.CV}$ - 135 Ex^2_{μ}	0.996	Y
Operation Time (σ)	**Ln Opt_{σ}** = - 10.3 - 1.19 ρ + 0.987 ρ^2 + 11.3 $Arriv_{\mu}$ + 0.0515 $Arriv_{CV}$ - 37.1 $Arriv^2_{\mu}$ + 269 Ex_{μ} + 1.72 Ex_{CV} - 14.8 $Ex_{\mu.CV}$ - 6450 Ex^2_{μ}	0.975	-
Completion Rate (μ)	**Ln cr_{μ}** = 3.80 + 1.85 ρ - 1.03 ρ^2 - 20.4 $Arriv_{\mu}$ - 0.0600 $Arriv_{\mu.CV}$ + 55.4 $Arriv^2_{\mu}$ - 21.5 Ex_{μ}	0.994	Y
Completion Rate (σ)	**Sq cr_{σ}** = 1.28 + 0.689 ρ - 0.093 ρ^2 - 7.34 $Arriv_{\mu}$ + 7.69 $Arriv_{\mu.CV}$ + 4.94 $Arriv^2_{\mu}$ + 2.27 Ex_{μ} + 3.04 $Ex_{\mu.CV}$ - 413 Ex^2_{μ}	0.681	Y

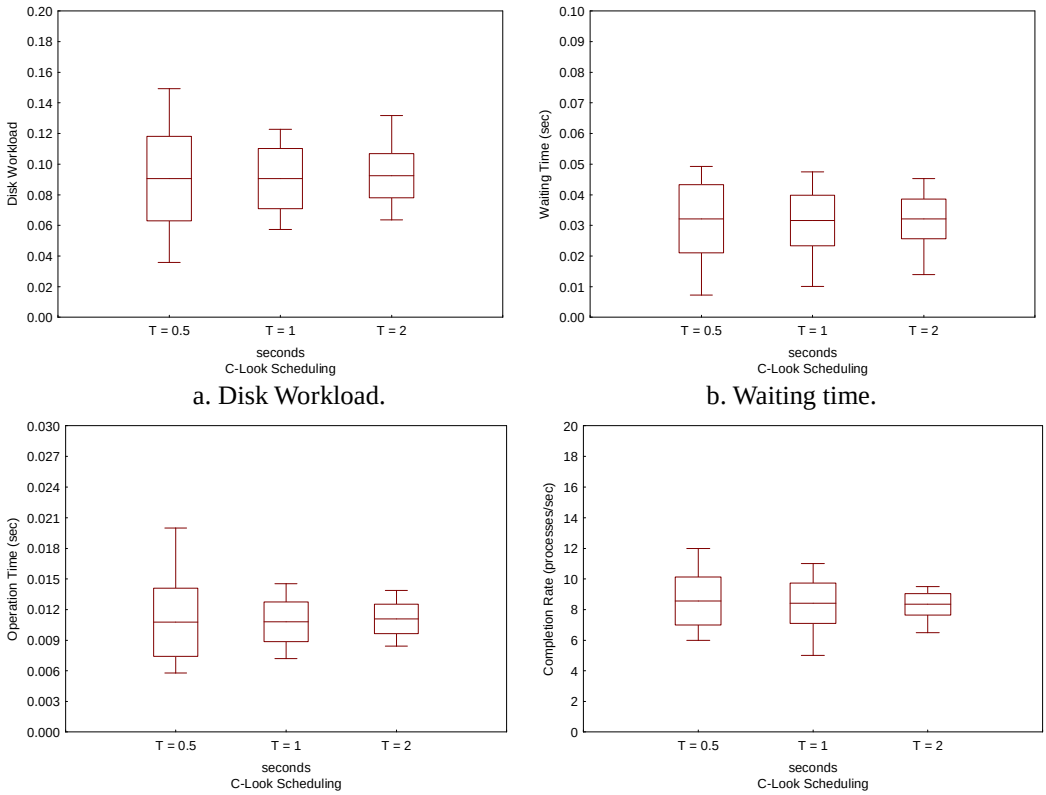
Table 28. Regression models for service workload and performance at processor with MLF algorithm.

Metrics	Regression model	R-sq (pred)	WLS
CPU Workload (μ)	**Sq U_{μ}** = 0.238 + 1.02 ρ - 0.369 ρ^2 - 1.6 $Arriv_{\mu}$ - 0.000742 $Arriv_{CV}$ + 4.08 $Arriv^2_{\mu}$ + 15.09 Ex_{μ} + 0.00493 Ex_{CV} - 0.549 $Ex_{\mu.CV}$ - 462 Ex^2_{μ}	0.998	-
CPU Workload (σ)	**Sq U_{σ}** = 0.00219 - 0.265 ρ - 0.221 ρ^2 - 0.039 $Arriv_{\mu}$ + 0.0694 $Arriv_{CV}$ - 0.173 $Arriv_{\mu.CV}$ + 8.01 Ex_{μ} + 0.029 Ex_{CV} + 2.5 $Ex_{\mu.CV}$ - 268 Ex^2_{μ}	0.932	Y
Waiting Time (μ)	**Ln Wt_{μ}** = - 10.3 - 9.88 ρ + 9.32 p^2 + 6.59 ρ_{HP} - 0.692 ρ_{SP} - 33.4 $Arriv_{\mu}$ - 5.27 $Arriv_{\mu.CV}$ + 1.45 $Arriv_{CV}$ + 204 $Arriv^2_{\mu}$ + 898 Ex_{μ} + 1.51 Ex_{CV} - 131 $Ex_{\mu.CV}$ - 15326 Ex^2_{μ}	0.856	-
Waiting Time (σ)	**Ln Wt_{σ}** = - 12.1 - 6.09 ρ + 6.58 ρ^2 + 5.9 ρ_{HP} - 14.2 $Arriv_{\mu}$ + 2.22 $Arriv_{CV}$ - 9.14 $Arriv_{\mu.CV}$ + 158 $Arriv^2_{\mu}$ + 671 Ex_{μ} + 1.99 Ex_{CV} - 129 $Ex_{\mu.CV}$ - 5125 Ex^2_{μ}	0.734	Y
Operation Time (μ)	**Sq Opt_{μ}** = 0.0394 - 0.0125 ρ + 0.011 ρ^2 - 0.0248 $Arriv_{\mu}$ + 0.00256 $Arriv_{\mu.CV}$ + 8.27 Ex_{μ} + 0.0016 Ex_{CV} - 0.205 $Ex_{\mu.CV}$ - 138 Ex^2_{μ}	0.995	-
Operation Time (σ)	**Ln Opt_{σ}** = - 10.6 - 0.402 ρ + 0.179 ρ_{HP} + 12.7 $Arriv_{\mu}$ + 0.0697 $Arriv_{CV}$ - 37.2 $Arriv^2_{\mu}$ + 273 Ex_{μ} + 1.89 Ex_{CV} - 47.7 $Ex_{\mu.CV}$ - 5049 Ex^2_{μ}	0.974	-
Completion Rate (μ)	**Ln cr_{μ}** = 3.77 + 1.66 ρ - 0.777 ρ^2 - 20.7 $Arriv_{\mu}$ - 0.068 $Arriv_{\mu.CV}$ + 56.5 $Arriv^2_{\mu}$ - 4.9 Ex_{μ} + 0.071 $Ex_{\mu.CV}$ - 827 Ex^2_{μ}	0.995	-
Completion Rate (σ)	**Ln cr_{σ}** = 0.706 + 0.689 ρ - 15.8 $Arriv_{\mu}$ + 13.6 $Arriv_{\mu.CV}$ + 9.19 $Arriv^2_{\mu}$ + 19.5 Ex_{μ} - 1.86 $Ex_{\mu.CV}$	0.829	Y

4.8.3 Impacts of service parameters and factors on disk workload and performance

Similar to processor experiments, the framework described in section 4.5 is used to estimate workload and performance metrics of services at the disk based on the data collected from the experimental conditions (cases) in section 4.7.2. For each of the 10 simulation runs (replicates) for each case, workload and performance metrics are obtained, using equations 1-5, for each of the services competing for the disk. Figure 29 shows the effect of using different values of *T* when estimating the workload and performance metrics for service 1 in case 8 run

1. *T* is the length of the period used for calculating the workload and performance metrics. As it can be seen from Figure 29, the mean value of the workload and performance metrics estimated for service 1 of case 8 run 1 is not sensitive to the length of the period (*T*). Similar effect with *T* is observed for the workload and performance metrics of services in all cases. This effect increases the confidence for using these metrics as estimates of service workload and performance at the disk.



a. Disk Workload. b. Waiting time.

c. Operation time. d. Completion rate.

Figure 29. The effect of *T* on service workload and performance metrics at disk.

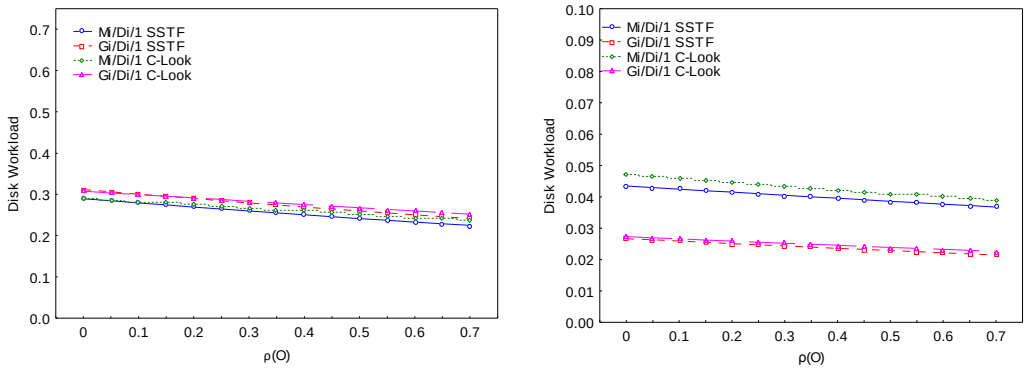
The metrics are estimated based on the average of individual service instances observed during *n* periods of length *T*, as *T* increases, the number of service instances observed in each period increases and this increase in the

number of service instances observed tends to reduce the variance of the workload and performance metrics estimated for each service. The impacts of service parameters on the mean and standard deviation of the workload and performance metrics for individual services are discussed next. The length of the period, *T*, is set to 2 seconds. Since each simulation run (replicate) for each of the cases is run for 100 seconds, by setting *T* equal to 2 seconds each simulation run is divided into 50 periods and the information observed during these periods is used to estimate the workload and performance metrics of services (Equations 1-5). To understand the impact of workload intensity (WI_{Disk}) on individual service workload and performance, workload intensity is decomposed in ρ_i and ρ_O. ρ_i is the workload intensity due to the service type i and is estimated just as in Eq. 12, but considering only service type i. ρ_O is the workload intensity due to other services competing for the disk and is estimated using Eq. 12, but considering all service types except service type i.

Disk Workload

Figure 30 shows the impacts of the workload imposed by other services (ρ_O) on the mean (μ) and standard deviation (σ) of the service workload at disk. The workload imposed by other services has a decreasing effect on the mean (μ) and standard deviation (σ) of the service workload at disk. This effect is caused by the reduction in access time due to the increase of service instances in queue. When the number of service instances in queue increases, the seek distance per disk access is reduced since the disk has more service instances to choose from,

thus reducing the access time. $M_i/D_i/1$ represents those experimental conditions (cases) with exponential arrival distributions. $G_i/D_i/1$ represents those cases with arrivals being normal distributed. For constructing Figures 30-44, service parameters are kept at constant values in each figure. For example, for constructing Figure 30 the arrival distribution mean of the service ($Arriv_\mu$=0.05) and the block size were kept (B= 0.032) at constant values. Each figure indicates the parameters kept at constant values for its construction.

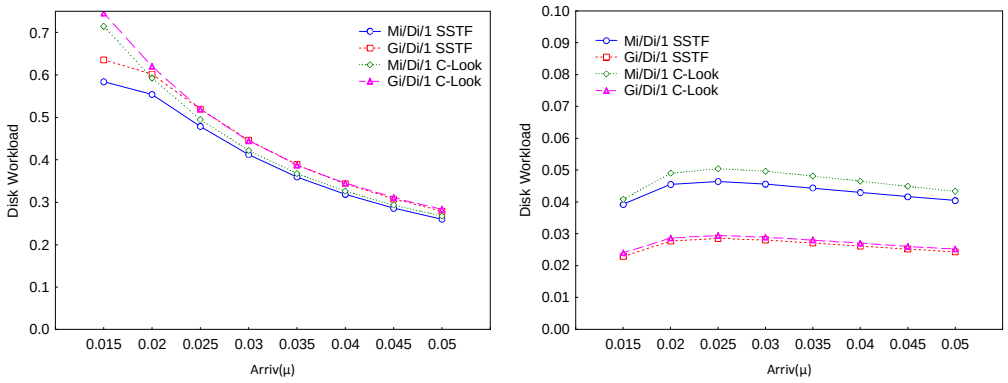


a. Impact of ρ_O on Disk Workload (μ) b. Impact of ρ_O on Disk Workload (σ)

Constant parameters: $Arriv_\mu$=0.05, B= 0.032.

Figure 30. Impacts of workload by other services (ρ_O) on service workload at disk.

Figure 31 shows the impacts of service arrival distribution mean ($Arriv_\mu$) on disk workload mean (μ) and standard deviation (σ). The arrival distribution mean of the service has a decreasing effect on the processor workload mean (μ). The standard deviation (σ) of disk workload varies within a certain range with the arrival distribution mean ($Arriv_\mu$), first increasing with $Arriv_\mu$ and then showing a slow decreasing effect.

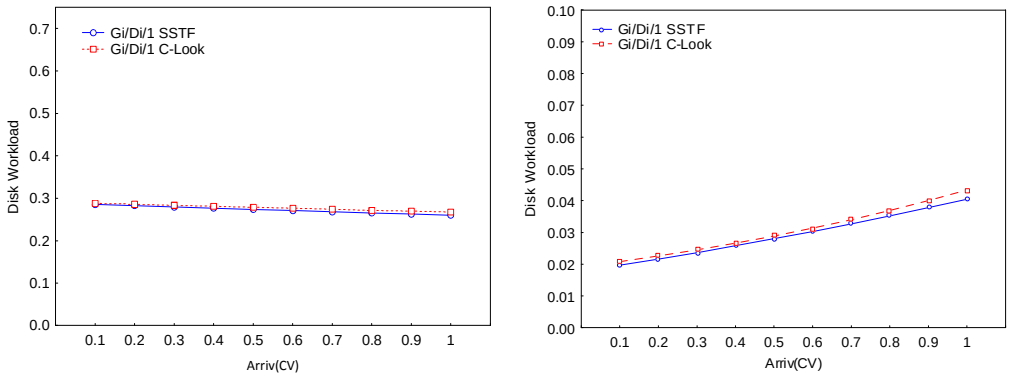


a. Impact of $Arriv_{\mu}$ on Disk Workload (μ) b. Impact of $Arriv_{\mu}$ on Disk Workload (σ)

Constant parameters: B=0.032, ρ_O=0.3.

Figure 31. Impacts of arrival distribution mean ($Arriv_{\mu}$) on Disk Workload.

Figure 32 shows the impacts of the coefficient of variation of the service arrival distribution ($Arriv_{CV}$) on disk workload mean (μ) and standard deviation (σ). The coefficient of variation (CV) is a normalized measure of the dispersion of a probability distribution and is defined as the ratio of the standard deviation (σ) to the mean (μ). Exponential distributions has a CV = 1, and normal distributions for disk experiments are restricted to have a CV = 1/3. The arrival distribution CV of the service tends to decrease the mean (μ) and increase the standard deviation (σ) of disk workload, but only the increasing effect on the standard deviation (σ) of disk workload is statistically significant.

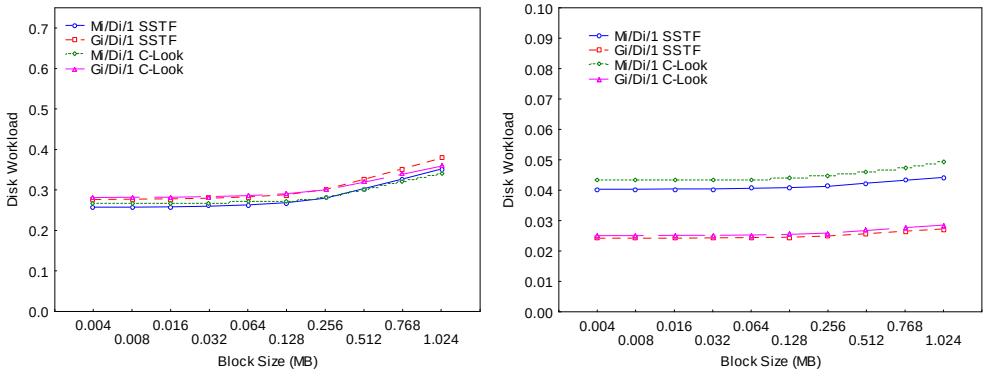


a. Impact of $Arriv_{CV}$ on Disk Workload (μ) b. Impact of $Arriv_{CV}$ on Disk Workload (σ)

Constant parameters: $Arriv_{\mu}$ =0.05, B=0.032, ρ_O=0.3.

Figure 32. Impacts of arrival distribution CV ($Arriv_{CV}$) on Disk Workload.

Figure 33 shows the impacts of block size (B) on disk workload mean (μ) and standard deviation (σ). B has an increasing effect on disk workload mean (μ) and standard deviation (σ) due to the increase of transfer time.



a. Impact of B on Disk Workload (μ) b. Impact of B on Disk Workload (σ)

Constant parameters: $Arriv_{\mu}$=0.05, ρ_O=0.3.

Figure 33. Impacts of block size (B) on Disk Workload.

Table 29 shows the impacts on the mean (μ) and standard deviation (σ) of disk workload with service parameters: workload imposed by other services (ρ_o), arrival distribution mean ($Arriv_{\mu}$), arrival distribution CV ($Arriv_{CV}$), and block size (B). From Figures 30-33, it can be observed the mean (μ) and standard

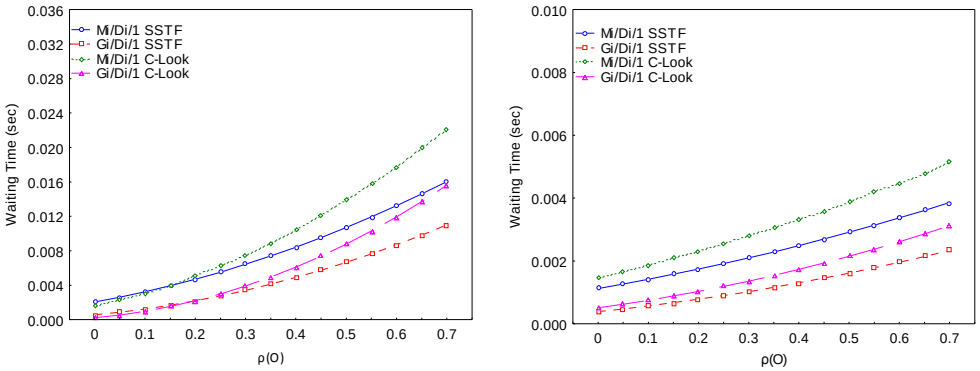
deviation (σ) of disk workload depend on the arrival distribution assumed. When the arrival distribution is exponential, the disk workload mean (μ) is smaller than with the normal distribution, but the standard deviation (σ) of disk workload tends to be larger than with normal distribution. The exponential distribution has larger coefficient of variation (CV) than the normal distribution, and larger $Arriv_{CV}$ tends to increase the standard deviation (σ) of disk workload (see Figure 32). The standard deviation (σ) of the disk workload is significantly larger with C-Look algorithm.

Table 29. Service parameters effects on Disk Workload mean (μ) and standard deviation (σ).

Parameters effects on:	ρ_o	$Arriv_{\mu}$	$Arriv_{CV}$	**B**
Disk Workload (μ)	↓	↓	↓	↑
Disk Workload (σ)	↓	Λ	↑	↑

Waiting Time

Figure 34 shows the increasing effect of the workload imposed by other services (ρ_O) on the mean (μ) and standard deviation (σ) of service waiting time. Increasing the workload imposed by other services (ρ_O) increases the number of service instances in queue, thus increasing the service waiting time mean (μ) and standard deviation (σ).

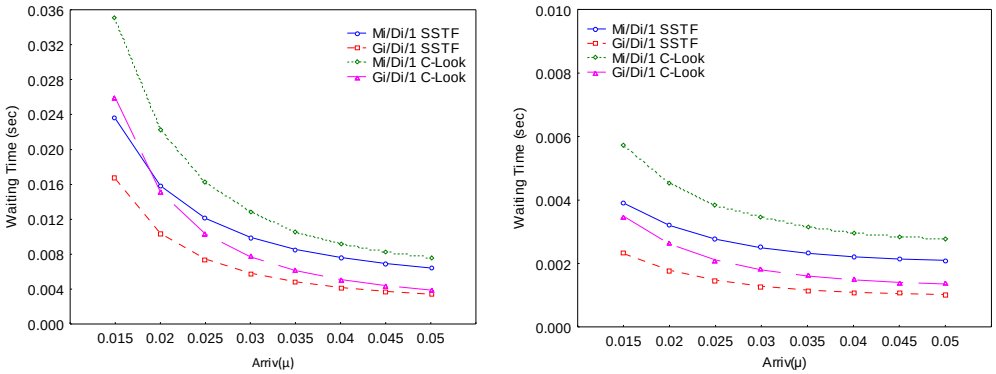


a. Impact of ρ_O on Waiting Time (μ) b. Impact of ρ_O on Waiting Time (σ)

Constant parameters: $Arriv_\mu$=0.05, B= 0.032.

Figure 34. Impacts of workload by other services (ρ_O) on Waiting Time.

Figure 35 shows the decreasing effect of arrival distribution mean ($Arriv_\mu$) on service waiting time mean (μ) and standard deviation (σ). Increasing $Arriv_\mu$ decreases the number of service instances in queue, therefore decreasing the service waiting time mean (μ) and standard deviation (σ).

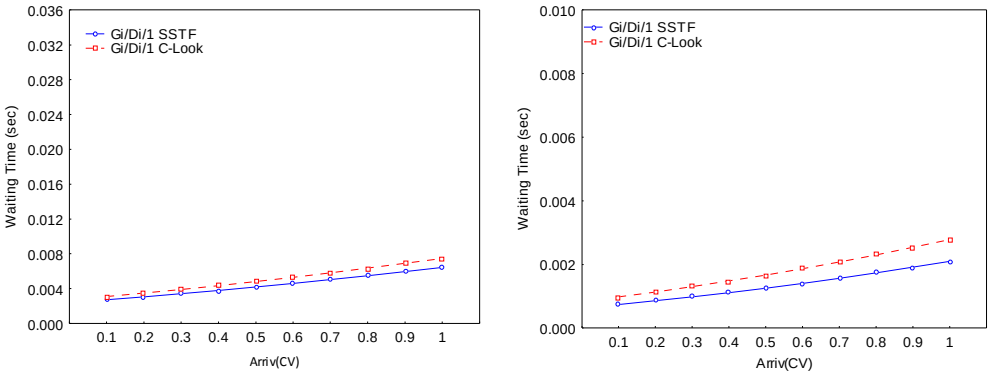


a. Impact of $Arriv_\mu$ on Waiting Time (μ) b. Impact of $Arriv_\mu$ on Waiting Time (σ)

Constant parameters: B=0.032, ρ_O=0.3.

Figure 35. Impacts of arrival distribution mean ($Arriv_\mu$) on Waiting Time.

Figure 36 shows the increasing effect of arrival distribution CV ($Arriv_{CV}$) on service waiting time mean (μ) and standard deviation (σ).

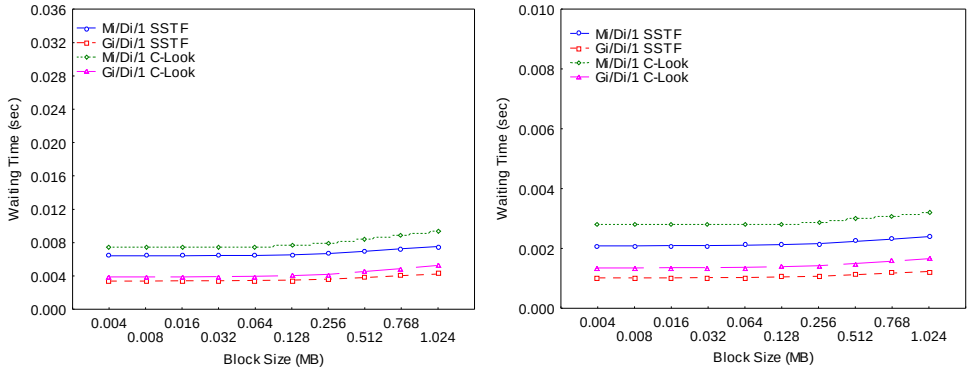


a. Impact of $Arriv_{CV}$ on Waiting Time (μ) b. Impact of $Arriv_{CV}$ on Waiting Time (σ)

Constant parameters: $Arriv_{\mu}$ =0.05, B=0.032, ρ_O=0.3.

Figure 36. Impacts of arrival distribution CV ($Arriv_{CV}$) on Waiting Time.

Figure 37 shows the increasing effect of block size (B) on service waiting time mean (μ) and standard deviation (σ). Increasing B increases the transfer time of service instances, thus increasing the waiting time in queue for service instances requiring the disk.



a. Impact of B on Waiting Time (μ) b. Impact of B on Waiting Time (σ)

Constant parameters: $Arriv_{\mu}$=0.05, ρ_O=0.3.

Figure 37. Impacts of block size (B) on Waiting Time.

Table 30 summarizes the service parameters effects on service waiting time mean (μ) and standard deviation (σ). From Figures 34-37, it can be observed the mean (μ) and standard deviation (σ) of service waiting time depend on the

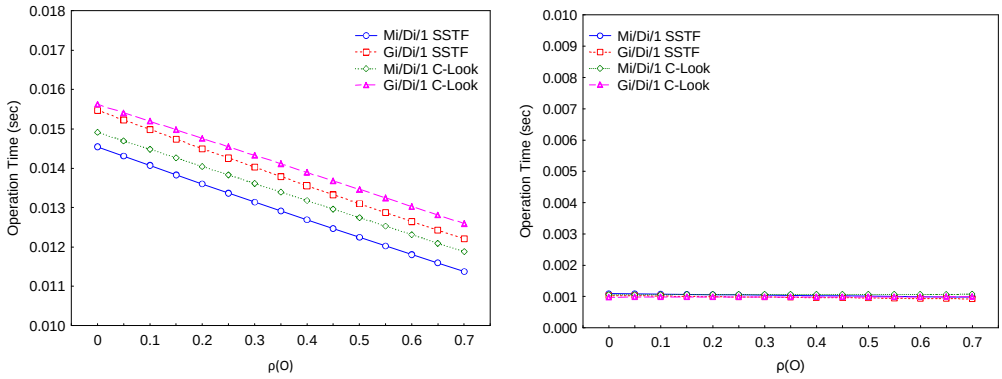
arrival distribution and the scheduling algorithm assumed. When the arrival distribution is exponential, the waiting time mean (μ) and standard deviation (σ) are larger than with normal distribution. This effect is due to the fact that larger $Arriv_{CV}$ tends to increase service waiting time mean (μ) and standard deviation (σ) and the exponential distribution has larger coefficient of variation (CV) than the normal distribution. The service waiting time mean (μ) and standard deviation (σ) are significantly larger with C-Look algorithm.

Table 30. Service parameters effects on Waiting Time mean (μ) and standard deviation (σ).

Parameters effect on:	**ρ_o**	**$Arriv_{\mu}$**	**$Arriv_{CV}$**	**B**
Waiting Time (μ)	↑	↓	↑	↑
Waiting Time (σ)	↑	↓	↑	↑

Operation Time

Figure 38 shows the impacts of the workload by other services (ρ_O) on the mean (μ) and standard deviation (σ) of service operation time. Increasing ρ_O has a decreasing effect on operation time mean (μ). This effect is due to the increase of service instances in queue and the consequent decrease of the access time part of operation time. ρ_O has no effect on the standard deviation (σ) of service operation time.

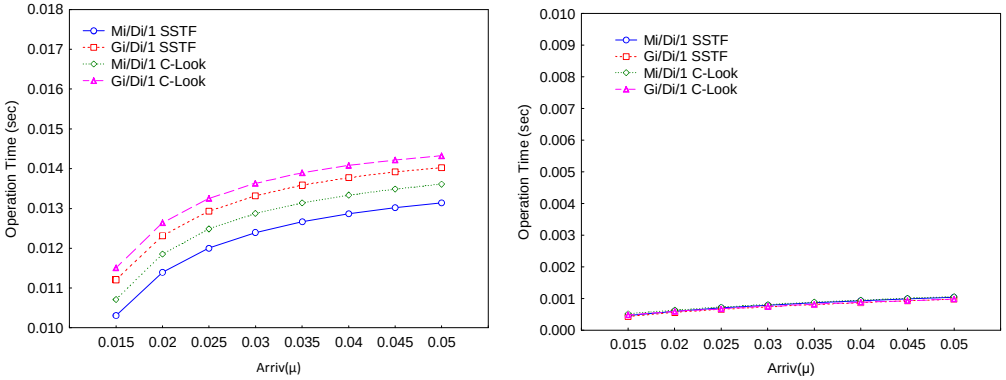


a. Impact of ρ_O on Operation Time (μ) b. Impact of ρ_O on Operation Time (σ)

Constant parameters: $Arriv_\mu$=0.05, B= 0.032.

Figure 38. Impacts of workload by other services (ρ_O) on Operation Time.

Figure 39 shows the increasing effect of the arrival distribution mean ($Arriv_\mu$) on service operation time mean (μ) and standard deviation (σ). Increasing $Arriv_\mu$ has an increasing effect on operation time mean (μ) and standard deviation (σ). This effect is due to the decrease of service instances in queue as result of increasing $Arriv_\mu$ and the consequent increase of the access time part of service operation time.

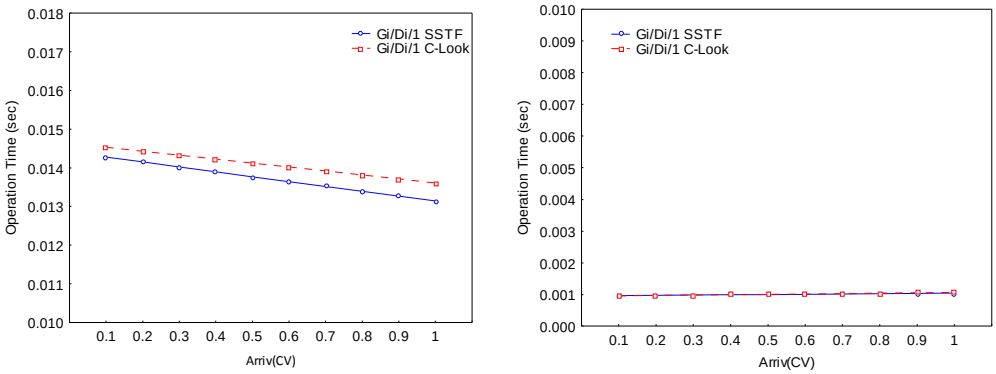


a. Impact of $Arriv_\mu$ on Operation Time (μ) b. Impact of $Arriv_\mu$ on Operation Time (σ)

Constant parameters: B=0.032, ρ_O=0.3.

Figure 39. Impacts of arrival distribution mean ($Arriv_\mu$) on Operation Time.

Figure 40 shows the impacts of the arrival distribution CV ($Arriv_{CV}$) on service operation time mean (μ) and standard deviation (σ). Increasing $Arriv_{CV}$ has a decreasing effect on operation time mean (μ). $Arriv_{CV}$ has no significant effect on the standard deviation (σ) of service operation time.

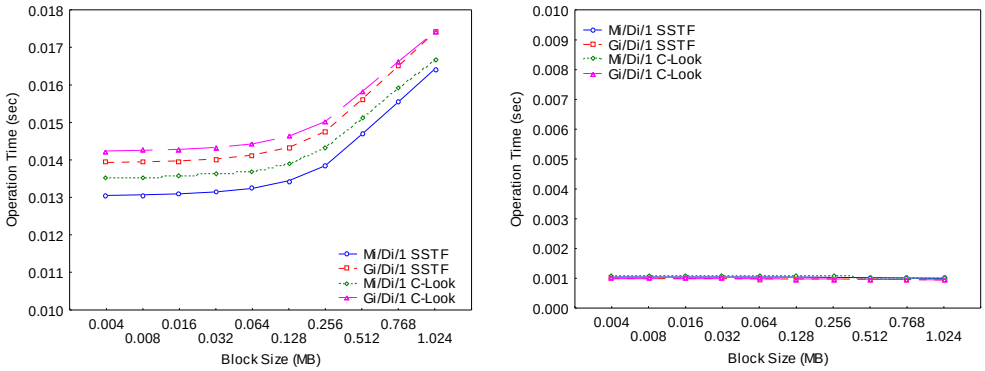


a. Impact of $Arriv_{CV}$ on Operation Time (μ) b. Impact of $Arriv_{CV}$ on Operation Time (σ)

Constant parameters: $Arriv_{\mu}$ =0.05, B=0.032, ρ_O=0.3.

Figure 40. Impacts of arrival distribution CV ($Arriv_{CV}$) on Operation Time.

Figure 41 shows the impacts of the block size (B) on service operation time mean (μ) and standard deviation (σ). Increasing B has an increasing effect on operation time mean (μ) due to the increase in the transfer time part of the operation time. B has no significant effect on the standard deviation (σ) of service operation time.



a. Impact of B on Operation Time (μ) b. Impact of B on Operation Time (σ)

Constant parameters: $Arriv_{\mu}$=0.05, ρ_O=0.3.

Figure 41. Impacts of block size (B) on Operation Time.

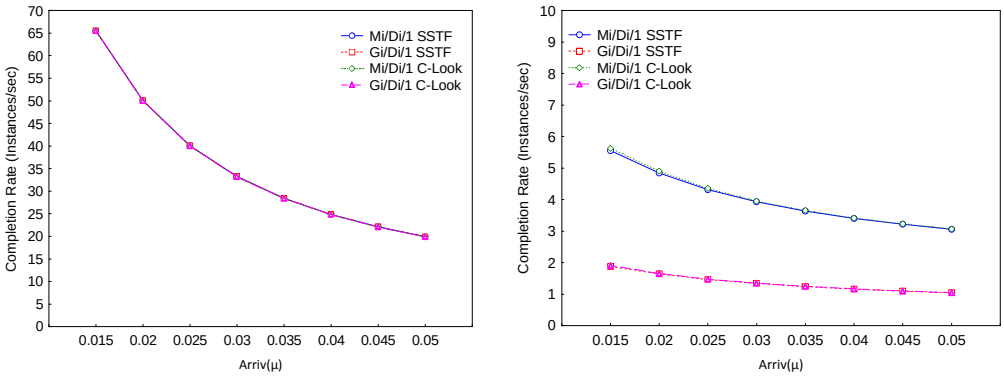
Table 31 summarizes the service parameters effects on service operation time mean (μ) and standard deviation (σ). From Figures 38-41, it can be observed the mean (μ) of service operation time depends on the arrival distribution and the scheduling algorithm assumed. When the arrival distribution is exponential the operation time mean (μ) is smaller than with normal distribution. Larger $Arriv_{CV}$ tends to decrease the access time part of operation time (see Figure 40) and the exponential distribution (CV=1) has larger CV than the normal distribution (CV= 1/3). The mean (μ) and standard deviation (σ) of the service operation time are significantly larger with C-Look algorithm.

Table 31. Service parameters effects on Operation Time mean (μ) and standard deviation (σ).

Parameters effects on:	**ρ_o**	**$Arriv_{\mu}$**	**$Arriv_{CV}$**	**B**
Operation Time (μ)	↓	↑	↓	↑
Operation Time (σ)	-	↑	-	-

Completion Rate

Figure 42 shows the decreasing effect on completion rate mean (μ) and standard deviation (σ) due to the increase in arrival distribution mean ($Arriv_{\mu}$).

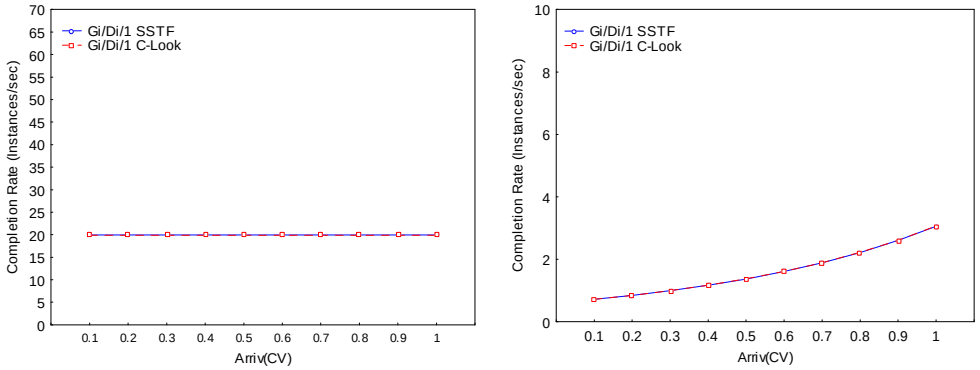


a. Impact of $Arriv_{\mu}$ on Completion Rate (μ) b. Impact of $Arriv_{\mu}$ on Completion Rate (σ)

Constant parameters: B=0.032, ρ_O=0.3.

Figure 42. Impacts of arrival distribution mean ($Arriv_{\mu}$) on Completion Rate.

Figure 43 shows the impacts of arrival distribution CV ($Arriv_{CV}$) on service completion rate mean (μ) and standard deviation (σ). $Arriv_{CV}$ has no significant effect on completion rate mean (μ), but it has a small increasing effect on completion rate standard deviation (σ).

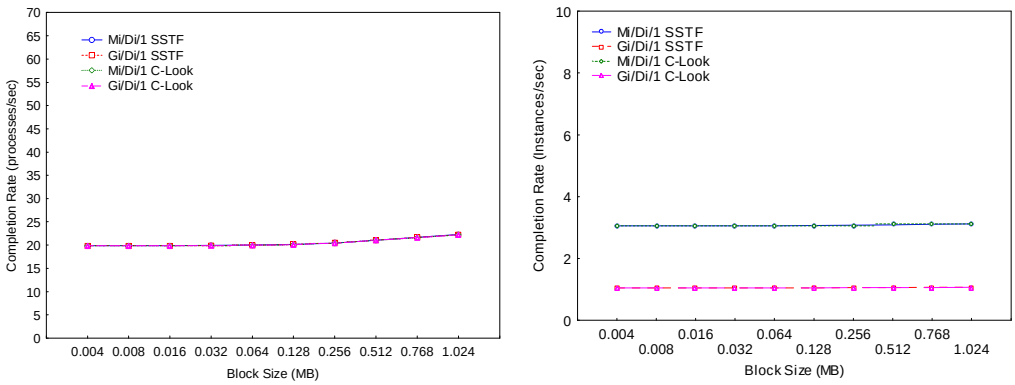


a. Impact of $Arriv_{CV}$ on Completion Rate (μ) b. Impact of $Arriv_{CV}$ on Completion Rate (σ)

Constant parameters: $Arriv_{\mu}$ =0.05, B=0.032, ρ_O=0.3.

Figure 43. Impacts of arrival distribution CV ($Arriv_{CV}$) on Completion Rate.

Figure 44 shows the impacts of block size (B) on service completion rate mean (μ) and standard deviation (σ). B has a small increasing effect on service completion rate mean (μ). This small effect is a consequence of the waiting time increase for service instances in queue due to the increase of transfer time. The increase in waiting time increases the number of service instances in queue, thus reducing service access time and increasing completion rate.



a. Impact of B on Completion Rate (μ) b. Impact of B on Completion Rate (σ)

Constant parameters: $Arriv_{\mu}$=0.05, ρ_O=0.3.

Figure 44. Impacts of block size (B) on Completion Rate.

Table 32 summarizes the service parameters effects on completion rate mean (μ) and standard deviation (σ). From Figures 42-44, it can be observed the standard deviation (σ) of completion rate is higher when the arrival distribution is exponential. $Arriv_{CV}$ has an increasing effect on the standard deviation (σ) of completion rate (see Figure 43), and exponential distribution (CV=1) has larger CV than the normal distribution (CV= 1/3).

Table 32. Service parameters effects on Completion Rate mean (μ) and standard deviation (σ).

Parameters effects on:	**ρ_o**	**$Arriv_{\mu}$**	**$Arriv_{CV}$**	**B**
Completion Rate (μ)	-	↓	-	↑(small)
Completion Rate (σ)	-	↓	↑	-

In general, the increase of workload due to other priority services (ρ_O) increases the service waiting time mean and standard deviation, but on the other hand decreases service operation time mean, and disk workload mean and standard deviation. The increase of arrival distribution mean increases operation time mean and standard deviation, but decreases the means and standard deviations of service waiting time and completion rate, and the mean of disk workload. The standard deviation of disk workload varies within a certain range with the arrival distribution mean, first increasing and then showing a slow decreasing effect. The larger the coefficient of variation (CV) for the arrival distribution the larger the mean and standard deviation of service waiting time, and the larger the standard deviation of disk workload and service completion rate. The arrival distribution CV has a decreasing effect on operation time mean. Increasing block size increases transfer time and therefore increase the means and standard deviations of disk workload and service waiting time, and the means of service operation time and completion rate. The standard deviation of disk workload and the mean and standard deviation of service operation time tend to be larger with C-Look algorithm than with SSTF algorithm. Service waiting time mean and standard deviation are also larger with C-Look algorithm. These effects were expected since SSTF algorithm is more efficient in reducing the seek time

which is the major component of operation time. A problem with SSTF algorithm occurs in high workload intensity conditions and localized access pattern, where services instances with larger track distances from the current read-write head location tend to have large waiting times (Thomasian and Liu 2002). This effect is not perceived in the experimental conditions since full disk capacity is never reached and random access patterns are assumed for services.

4.8.4 Workload and performance models for disk

Tables 33-34 provide the service workload and performance models for disk with C-Look and SSTF scheduling algorithms respectively. These models accurately capture the impacts of service parameters on workload and performance metrics described in the previous section (4.8.3). Multiple linear regression was used to build the models, polynomial and interaction terms (ρ^2, $Arriv^2_{\mu}$, B^2, $Arriv_{\mu.CV}$) were included when necessary to increase model performance. Natural log (Ln) and square root (Sq) transformations were applied to the workload and performance metrics with the same purpose. Similar to the regression models for processor workload and performance metrics, when analyzing the residuals for the regression models, for some of the metrics, it was found the residuals had non-constant variance. The common pattern identified in the residuals appears in Figure 27a, where the variance of the residuals increases with the fitted values. Weighted least square (WLS) regression was applied (Montgomery, Peck and Vining 2006) to correct this inequality of variance problem. Weights were defined as the inverse of the variance observed at any

point, and were estimated using the replicate runs for each of the experimental conditions (cases). Ten fold cross-validation (Tan, Kumar and Steinbach 2006) was used to estimate the predictive R-square values for the models (Tables 33-34). The predictive R-sq values obtained using cross-validation show the regression models are a good fit for the data and provide the confidence to use these models for prediction of service workload and performance at disk. The estimates obtained from these models can be used for taking workload and performance management decisions.

Table 33. Regression models for service workload and performance at disk with C-Look algorithm.

Metric	Regression model	R-sq (pred)	WLS
Disk Workload (μ)	**Sq U_{μ}** = 0.361 + 0.956 ρ - 0.398 ρ^2 - 0.0751 ρ_O - 0.692 $Arriv_{\mu}$ - 0.0273 $Arriv_{CV}$ + 0.109 $Arriv_{\mu.CV}$ + 0.562 $Arriv^2_{\mu}$ + 0.0194 B	0.998	-
Disk Workload (σ)	**Ln U_{σ}** = - 3.98 + 1.45 ρ - 1.44 ρ^2 - 0.27 ρ_O - 3.87 $Arriv_{\mu}$ + 0.794 $Arriv_{CV}$ + 0.405 $Arriv_{\mu.CV}$ + 3.87 $Arriv^2_{\mu}$ + 0.0879 B	0.934	-
Waiting Time (μ)	**Sq Wt_{μ}** = - 0.0449 + 0.143 ρ + 0.016 ρ^2 + 0.156 ρ_O + 0.206 $Arriv_{\mu}$ + 0.0394 $Arriv_{CV}$ - 0.104 $Arriv_{\mu.CV}$ - 0.196 $Arriv^2_{\mu}$	0.981	-
Waiting Time (σ)	**Sq Wt_{σ}** = - 0.0154 + 0.0692 ρ - 0.0182 ρ^2 + 0.0475 ρ_O + 0.263 $Arriv_{\mu}$ + 0.0249 $Arriv_{CV}$ - 0.0211 $Arriv_{\mu.CV}$ - 0.302 $Arriv^2_{\mu}$	0.932	Y
Operation Time (μ)	**Opt_{μ}** = 0.0167 - 0.00273 ρ - 0.00137 ρ^2 - 0.00433 ρ_O - 0.00118 $Arriv_{CV}$ + 0.00312 $Arriv_{\mu.CV}$ - 0.00705 $Arriv^2_{\mu}$ + 0.00335 B	0.947	Y
Operation Time (σ)	**Ln Opt_{σ}** = - 6.86 - 1.23 ρ + 0.288 ρ^2 + 0.0232 ρ_O + 4.43 $Arriv_{\mu}$ + 0.116 $Arriv_{CV}$ - 4.88 $Arriv^2_{\mu}$ + 0.0334 B	0.93	-
Completion Rate (μ)	**Sq cr_{μ}** = 2.71 + 8.25 ρ - 2.46 ρ^2 -0.0479 ρ_O - 6.7 $Arriv_{\mu}$ + 8.79 $Arriv^2_{\mu}$ - 0.191 B	0.999	-
Completion Rate (σ)	**Ln cr_{σ}** = - 0.65 + 1.37 ρ - 0.53 ρ^2 - 3.68 $Arriv_{\mu}$ + 1.63 $Arriv_{CV}$ - 0.481 $Arriv_{\mu.CV}$ + 4.91 $Arriv^2_{\mu}$ - 0.0518 B	0.975	-

Table 34. Regression models for service workload and performance at disk with SSTF algorithm.

Metric	Regression model	R-sq (pred)	WLS
Disk Workload (μ)	**Ln U_{μ}** = - 1.55 + 3.3 ρ - 2 ρ^2 - 0.364 ρ_O - 7.3 $Arriv_{\mu}$ - 0.128 $Arriv_{CV}$ + 0.471 $Arriv_{\mu.CV}$ + 7.01 $Arriv^2_{\mu}$ + 0.193 B - 0.0232 B^2	0.998	Y
Disk Workload (σ)	**Sq U_{σ}** = 0.119 + 0.142 ρ - 0.133 ρ^2 - 0.0241 ρ_O - 0.161 $Arriv_{\mu}$ + 0.0713 $Arriv_{CV}$ - 0.0704 $Arriv_{\mu.CV}$ + 0.216 $Arriv^2_{\mu}$ + 0.00523 B	0.933	Y
Waiting Time (μ)	**Sq Wt_{μ}** = - 0.0255 + 0.108 ρ + 0.00853 ρ^2 + 0.116 ρ_O + 0.2 $Arriv_{\mu}$ + 0.0366 $Arriv_{CV}$ - 0.111 $Arriv_{\mu.CV}$ - 0.19 $Arriv^2_{\mu}$ - 0.000884 B	0.982	-
Waiting Time (σ)	**Sq Wt_{σ}** = - 0.0138 + 0.0577 ρ - 0.0169 ρ^2 + 0.0408 ρ_O + 0.257 $Arriv_{\mu}$ + 0.0219 $Arriv_{CV}$ - 0.0245 $Arriv_{\mu.CV}$ - 0.31 $Arriv^2_{\mu}$	0.956	Y
Operation Time (μ)	**Sq Opt_{μ}** = 0.13 - 0.0123 ρ - 0.00632 ρ^2 - 0.0199 ρ_O - 0.0106 $Arriv_{\mu}$ - 0.00652 $Arriv_{CV}$ + 0.0222 $Arriv_{\mu.CV}$ - 0.0229 $Arriv^2_{\mu}$ + 0.0147 B	0.959	Y
Operation Time (σ)	**Ln Opt_{σ}** = - 6.86 - 1.02 ρ - 0.155 ρ_O + 4.98 $Arriv_{\mu}$ + 0.0875 $Arriv_{CV}$ - 5.65 $Arriv^2_{\mu}$ + 0.0276 B	0.926	-
Completion Rate (μ)	**Sq cr_{μ}** = 2.71 + 8.25 ρ - 2.46 ρ^2 -0.0463 ρ_O - 6.7 $Arriv_{\mu}$ + 8.79 $Arriv^2_{\mu}$ - 0.191 B	0.999	-
Completion Rate (σ)	**Ln cr_{σ}** = - 0.633 + 1.31 ρ - 0.499 ρ^2 - 3.76 $Arriv_{\mu}$ + 1.63 $Arriv_{CV}$ - 0.494 $Arriv_{\mu.CV}$ + 4.98 $Arriv^2_{\mu}$ - 0.046 B	0.976	-

4.9 Conclusions

When dealing with competing service requests with specific performance (QoS) requirements in service-based systems (SBS), the system must determine if its limited resources can accommodate the service requests and provide the performance (QoS) levels required. Understanding the impacts of services on resource workload and service performance becomes necessary to ensure that services are provided at the required performance (QoS) levels and system resources are managed efficiently. Previous studies (Vazhkudai and Schopf 2002; Doyle, et al. 2003; Abrahao and Zhang 2004; Shivam, Babu and Chase 2006; Sun and Ifeachor 2006; Harada, Ushio and Nakamoto 2007; Kjaer, Kihl and

Robertsson 2009; Kan, Sun and Ifeachor 2010; Kang and Suh 2011) focused on modeling system dynamics for individual services, covering specific resources or performance metrics. Workload and performance models of services are required at a more comprehensive, system-wide scale independent of services functional and non-functional requirements. To address these needs, a framework is proposed in this part of the dissertation to estimate the impacts of service workload and performance at individual resources considering the usage profiles of the services competing for the resource and the resource-sharing schemes. Simulation models for processor and disk components were designed to collect the specific service and resource information required by the framework. Simulation provides the modeling flexibility that other modeling techniques such as queuing theory lack. The simulation models incorporate hardware (e.g. speed, capacity) and software (e.g. access, allocation, scheduling) characteristics of each resource that can be customized to model different hardware and software configurations. Two performance (QoS) metrics were investigated: completion rate and response time. Response time was further brokendown into waiting time and operation time. Resource workload was defined as the proportion of time the resource was busy executing service instances (requests). Experimental conditions (cases) were run using processor and disk models to investigate the impacts of various service parameters (e.g. arrival distribution, execution time distribution, priority, workload intensity, scheduling algorithm) on the workload and performance metrics.

For processor, the results show the increase in workload intensity due to higher priority services (ρ_{HP}) mainly increases service waiting time due to the increase in the number of service instances in queue waiting for processor. The increase in arrival distribution mean ($Arriv_{\mu}$**)** decreases the frequency of service instances arriving at the processor queue, thus decreasing the means and standard deviations of service workload and completion rate, and increasing the standard deviation of service operation time. The larger the coefficient of variation (CV) of the arrival distribution ($Arriv_{CV}$), the larger the standard deviations of service workload and completion rate metrics. The mean of the execution distribution (Ex_{μ}) directly increases service operation time and thus increases service workload, waiting time, and completion rate. The larger the coefficient of variation (CV) of the execution distribution (Ex_{CV}), the larger the standard deviations of service workload and operation time metrics. The scheduling algorithm has an impact on service workload and performance metrics at processor. Using round robin priority preemptive (RRP) scheduling increases the standard deviation of services' workload on processor. Service waiting time, especially for those services with long execution times, tend to be larger with MLF scheduling since services with long execution times stay in the lowest priority queue longer time, waiting for services in higher priority queues to complete execution.

For disk, the results show the increase in workload intensity due to other services (ρ_O) directly increases the number of service instances in queue, thus increasing service waiting time mean and standard deviation, but on the other

hand, this increase in the number of service instances in queue decreases the disk access time of services thus decreasing the disk workload mean and standard deviation, and the service operation time mean. The increase in arrival distribution mean ($Arriv_{\mu}$**)** decreases the frequency of service instances arriving at the disk queue, thus decreasing the means and standard deviations of service waiting time, completion rate, and the mean of disk workload, but increasing the disk access time of services and consequently the mean and standard deviation of service operation time. The larger the coefficient of variation (CV) of the arrival distribution ($Arriv_{CV}$), the larger the mean and standard deviation of service waiting time, and the larger the standard deviations of disk workload and completion rate metrics. Additionally, a decrease effect on operation time mean is observed with the increase in $Arriv_{CV}$. Increasing the block size (B) increases the transfer time and thus increases the means of disk workload, waiting time, operation time and completion rate metrics, and increases the standard deviations of disk workload and waiting time metrics. The scheduling algorithm has an impact on service workload and performance metrics at disk. In general, service operation and waiting times tend to be longer with C-Look algorithm due to the longer access time in comparison with SSTF scheduling. These effects were expected since SSTF algorithm is more efficient in reducing the seek time part of the access time. The effect of longer waiting times for services instances with larger track distances from the current read-write head position when using SSTF scheduling, is not observed in the experimental conditions since full disk capacity is never reached and random access patterns are assumed for services.

Regression models were built to capture the above impacts of service parameters on workload and performance metrics. The predictive R-sq values obtained using cross-validation provide the confidence to use these models for prediction of service workload and performance at processor and disk resources. Independent of the number of services competing for the resource and/or the profiles of the services competing for the resource, the workload and performance models obtained can be used to estimate the workload and performance of services, but if the hardware or software characteristics of the resource change, for example using a different scheduling algorithm, the workload and performance models will no longer be valid and the model coefficients for the service parameters will need to be re-estimated according to the new hardware and software configuration of the resource. If required, the estimates obtained from workload and performance models at individual resources can be aggregated to obtain the workload and performance of services through multiple system resources. Although service workload and performance models are built only for processor and disk resources, the framework presented in this study is general applicable to model service workload and performance at other system resources (e.g. network) assuming an appropriate model of the resource incorporating major resource hardware and software characteristics is available.

CONCLUSIONS

The dynamicity, flexibility and loosely-coupled capability of service-based systems (SBS) cause service performance to become one of the most challenging aspects in SBS. Service performance is important for customer satisfaction and loyalty, therefore it is critical to IT service providers. Resource management is also critical to IT service providers since the availability and further allocation of system resources to services impact their performance. Previous studies have identified the value of modeling system dynamics to guide resource allocation in achieving the required service performance (Wu and Woodside 2004; Stewart and Shen 2005; Zhang, Bivens and Rezek 2007), but a general approach is not yet established to capture system dynamics under a wide variety of service conditions and independently of service functional and nonfunctional requirements.

This dissertation develops two methods to understand and model the cause-effect relations of service-related activities on resources workload and service performance.

Chapter 2 presents an empirical method to analyze and model the impacts of services on system activities, resources workload and service performance. The method requires the collection of system-wide dynamics data and the application of statistical analyses to extract the information required. The results show that the method is capable to: 1) uncover the impacts of services on resource workload and service performance, 2) identify interaction effects of multiple services running concurrently, 3) gain insights about resource and performance tradeoffs of

services, and 4) build service workload and performance models capturing system dynamics.

Chapter 3 presents a study to investigate the impacts services, security mechanisms and cyber attacks on resources workload and service performance. System dynamics data is collected under two services (voice communication and motion detection), two security mechanisms (data encryption and intrusion detection) and five cyber attacks (ARP poison, ping flood, vulnerability scan, fork bomb and remote dictionary). The results show the information obtained using the empirical method presented in Chapter 2 can be used to: 1) uncover interaction effects of service, security mechanism and cyber attacks, 2) identify tradeoffs within the limits of system resources, and 3) develop general/specific strategies for system survivability. The results obtained in Chapters 2 and 3 by using the empirical method to capture system dynamics provide useful knowledge of services, security mechanisms and cyber attacks that can be used for IT service providers for resource and performance management of services, and even system survivability.

Chapter 4 presents a general framework to estimate the impacts of service workload and performance at individual resources based on the usage profiles of the services competing for the resource and the resource-sharing schemes. This framework overcomes the limitations of the empirical method due to the time and effort required for experimental set-up, data collection and analysis for each service configuration of interest. Processor and disk models were designed to collect the service and resource information required by the framework. The

framework is used to investigate the impacts of various service parameters (e.g. arrival distribution, execution time distribution, priority, workload intensity, scheduling algorithm) on resource workload and performance metrics. The results show the framework can be used to: 1) uncover the impacts of service parameters on workload and performance metrics, and 2) build service workload and performance models at individual resources. The estimates for service workload and performance metrics at individual resources can later be aggregated to obtain workload and performance estimates of services through multiple system resources.

The empirical method and the theoretical framework represent two distinct alternatives to analyze and model the impacts of services on system resources and performance for SBS. The empirical method involves the experimental set-up and data collection under each service condition (configuration) of interest and further data analyses in order to build the workload and performance models. On the other hand by using the framework, service workload and performance models are provided for processor and disk resources under specific hardware (e.g. speed, capacity) and software (e.g. access, allocation, scheduling) characteristics. These models can be used to estimate service workload and performance at processor and disk based on the profiles of the services competing for the resources. If the profiles of the services competing for the resource change, the estimates for service workload and performance will change according to the workload and performance models. If the hardware or software characteristics of the resource change, for example using a different scheduling algorithm, then the workload

and performance models are no longer valid and need to be re-estimated according to the new hardware and software configuration of the resource.

The workload and performance models of services obtained through either the empirical method or the general framework can be used for efficient management of resource workload and service performance. These workload and performance models can be incorporated into service standardization for modeling, composition, monitoring, optimization and management stages of SBS.

Future work includes: 1) exploring the relation between service activities, system resources and service performance when system reaches a saturation point, 2) using the framework for the development of service workload and performance models for additional system resources including network, memory, video card, etc., 2) the evaluation of aggregated service workload and performance through multiple resources, and 4) the inclusion of the inter-component communication between resources.

REFERENCES

Abdel-qader, Fadi M. *Examples to create your Conferencing System in .NET, C# VOIP & Video Conferencing Systems using H.323 and TAPI 3.* 01 31, 2007. http://www.codeproject.com/KB/IP/Video_Voice_Conferencing.aspx (accessed 10 09, 2010).

Abrahao, Bruno, and Alex Zhang. *Characterizing application workloads on CPU utilization for utility computing.* Technical report, Palo Alto, CA, USA: HP Laboratories, 2004.

Atighetchi, M., P. Pal, F. Webber, R. Schantz, C. Jones, and J. Loyal. "Adaptive cyber defense for survival and intrusion tolerance." *IEEE Internet Computing, vol. 8, no. 6*, 2004: 25-33.

Bashandy, Ahmed R., Edwin K. P. Chong, and Arif Ghafoor. "Generalized quality-of-service routing with resource allocation." *IEEE Journal in Selected Areas of Communications, vol. 23, no. 2*, 2005: 450-463.

Chen, Yan, Toni Farley, and Nong Ye. "QoS requirements of network applications over the internet." *Information, Knowledge and System Management, vol. 4, no. 1*, 2004: 55-76.

Curbera, Francisco, Matthew Duftler, Rania Khalaf, William Nagy, Nirmal Mukhi, and Sanjiva Weerawarana. "Unraveling the web services web: an introduction to SOAP, WSDL, and UDDI." *IEEE Internet Computing, vol. 6, no. 2*, 2002: 86-93.

Daemen, J., and V. Rijmen. *The Design of Rijndael: AES- The Advanced Encryption Standard.* NJ, USA: Springer-Verlag New York Inc., 2001.

Doyle, Ronald P., Jeffrey Chaseq, Omer M. Asad, Wei Jin, and Amin M. Vahdat. "Model-based resource provisioning in a web service utility." *Fourth Symposium on Internet Technologies and Systems*, 2003.

Gross, Donald, and Carl Harris. *Fundamental of queueing theory.* New York, USA: John Wiley & Sons, Inc., 1998.

Harada, Fumiko, Toshimitsu Ushio, and Yukikazu Nakamoto. "Adaptive Resource Allocation Control for Fair QoS Management." *IEEE Transactions on Computers, vol. 56, no. 3*, 2007: 344-357.

Hu, Liang, Jian Cao, and Zhiping Gu. "Modeling semantic web service using semantic templates." *International Conference on Semantics, Knowledge and Grid*, 2008: 165-172.

Jacob, Bruce, Spencer Ng, and David Wang. *Memory Systems: Cache, DRAM, Disk.* USA: Morgan Kaufmann, 2008.

Kan, A, L Sun, and E Ifeachor. "Learning models for video quality over wireless local area networks and universal mobile telecommunication system networks." *IET Communications, vol. 4, no. 12*, 2010: 1389-1403.

Kang, K., and H. J. Suh. "Adaptive buffer control to minimize delay and guarantee service reliability." *IET Communications, vol. 5, no. 1*, 2011: 110-118.

Kirillov, Andrew. *Motion Detection algorithms.* 03 27, 2007. http://www.codeproject.com/KB/audio-video/Motion_Detection.aspx (accessed 10 09, 2010).

Kjaer, Martin Ansbjerg, Maria Kihl, and Anders Robertsson. "Resource allocation and disturbance rejection in web servers using SLAs and virtualized servers." *IEEE Transactions on Network and Service Management, vol. 6, no. 4*, 2009: 226-239.

Lakshminarasimhan, Deepak. *Wavelet based Cyber Attack Detection.* Thesis presented for the degree of Master in Science in Industrial Engineering, Tempe, Az, USA: Arizona State University, 2005.

Lamparter, Steffen, Anupriya Ankolekar, and Rudi Studer. "Preference-based selection of highly configurable web services." *International World Wide Web Conference*, 2007: 1013-1022.

Lee, Chen, John Lehoczky, Dan Siewiorek, Ragunathan Rajkumar, and Jeff Hansen. "A Scalable solution to the multi-resource QoS problem." *IEEE Real-Time Systems Symposium*, 1999: 315-326.

Li, W., L. Z. Shu, and Y. Feng. "A dynamic survivability reconfiguration framework based on QoS." *Proceedings of the 2009 International Conference on Advanced Computer Control.* Singapore, 2009. 103-106.

Li, Ying, Kewei Sun, Jie Qiu, and Ying Chen. "Self-reconfiguration of service-based systems: a case study for service level agreements and resource optimization." *IEEE International Conference on Web Services, vol. 1*, 2005: 266-273.

Lipson, H. F., and D. A. Fisher. "Survivability - a new techincal and business perspective." *Proccedings of the 1999 Workshop on New Security Paradigms.* Caledon Hills, Ontario CA, 1999. 33-39.

Liu, Yan, I. Gorton, and Liming Zhu. "Performance prediction of service-oriented applications on an enterprise service bus." *31st Annual International Computer Software and Applications Conference.* Beijing, China, 2007. 327-334.

Liu, Zhen, Laura Wynter, Cathy H. Xia, and Fan Zhang. "Parameter inference of queueing models for IT systems using end to end measurements." *Perfromance Evaluation, vol. 63*, 2006: 36-60.

Lumb, C. R., Schindler, J., Ganger, G. R., and Nagle D. F. "Towards higher disk head utilization: extracting free bandwidth from busy disk drives." *Symposium on Operating System Design and Implementation*. San Diego, CA, USA. 87-102.

McLachlan, G., and Peel, D. *Finite Mixture Models.* New York, USA: John Wiley & Sons, 2000.

Mann, H. B., and D. R. Whitney. "On a test of whether one of two random variables is sthocasthically larger than the other." *Annals of Mathematical Statistics, vol. 18, no. 1*, 1947: 50-60.

Microsoft. *Windows Perfomance Counters by Object.* 04 2009. http://technet.microsoft.com/en-us/library/cc783073%28WS.10%29.aspx (accessed 10 10, 2010).

Montgomery, D. C., E. A. Peck, and G. G. Vining. *Introduction to linear regression models.* New York, USA: John Wiley & Sons, 2006.

Montgomery, Douglas C. *Design and Analysis of Experiments, Sixth Edition.* John Wiley & Sons, Inc., 2005.

Montoro, Massimiliano. *Cain & Abel software v4.9.30.* 04 21, 2009. http://www.oxid.it/index.html (accessed 10 10, 2010).

Olejnik, S., and J. Algina. "Generalized Eta and Omega Square Statistics: Measures of effect size for some common research designs." *Psychological Methods, vol. 8, no. 4,* 2003: 434-447.

Paralel Data Lab. *The DiskSim Simulation Environment (v4.0).* 02 01, 2011. http://www.pdl.cmu.edu/DiskSim/ (accessed 12 22, 2011).

Riska, Alma, and Erik Riedel. "Disk drive level workload characterization." *Proceeding of USENIX Annual Technicall Conference.* Boston, MA, USA, 2006. 97-102.

Russinovich, Mark E., and David A. Solomon. *Microsoft Windows Internals: Microsoft Windows Server 2003, Windows XP and Windows 2000.* Redmond, Washington, USA: Microsoft Press Divison of Microsoft Corporation, 2005.

Shivam, Piyush, Shivnath Babu, and Jeffrey S. Chase. "Learning application models for utility resource planning." *IEEE International Conference on Autonomic Computing,* 2006: 255-264.

Silberschatz, Abraham, Peter B. Galvin, and Greg Gagne. *Operation Systems Concepts 8th Edition.* USA: John Wiley & Sons, Inc., 2009.

Sourcefire, Inc. *Snort.* http://www.snort.org/ (accessed 10 09, 2010).

Staikopoulos, Athanasios, Owen Cliffe, Razvan Popescu, Julian Padget, and Siobhan Clarke. "Template-based adaptation of semantic web services with model-driven engineering." *IEE Transactions on Services Computing, vol. 3, no. 2,* 2010: 116-128.

StatSoft, Inc. *STATISCA.* 2010. http://www.statsoft.com/ (accessed 10 10, 2010).

Stewart, C., and K. Shen. "Performance modeling and system management for multi-component online services." *2nd Symposium on Networked Systems Design & implementation.* 2005. 71-84.

Subrata, Riky, Albert Y. Zomaya, and Bjorn Landfeldt. "A cooperative game framework for QoS guided job allocation schemes in grids ." *IEEE Transactions on Computers, vol. 57, no. 10*, 2008: 1413-1422.

Sun, Lingfen, and Emmanuel C. Ifeachor. "Voice quality prediction models and their application in VoIP networks." *IEEE Transactions on Multimedia, vol. 8, no. 4*, 2006: 809-820.

Tan, P.N., V. Kumar, and M. Steinbach. *Introduction to data mining.* Pearson Addison Wesley, 2006.

Thomasian, Alexander, and Chang Liu. "Disk scheduling policies with lookahead." *ACM SIGMETRICS Performance Evaluation Review, vol. 30, no. 2*, 2002: 31.

Tools4ever. *Free Ping v2.0.* 04 2009. http://www.tools4ever.com/products/free/freeping/ (accessed 10 10, 2010).

Tran, Vuong Xuan, Hidekazu Tsuji, and Ryosuke Masuda. "A new QoS ontology and its QoS-based ranking algorithm for web services." *Simulation Modelling Practice and Theory, vol. 17, no. 8*, 2009: 1378-1398.

Vazhkudai, Sudharshan, and Jennifer M. Schopf. "Using disk throughput data in predictions of end-to-end grid data transfers." *International Worlkshop on Grid Computing*, 2002: 291-304.

Wang, Xia, Tomas Vitvar, Mick Kerrigan, and Ioan Toma. "A QoS-aware selection model for semantic web services." *International Conference on Service Oriented Computing*, 2006: 390-401.

Wu, Xiuping, and Murray Woodside. "Perfromance modeling from software components." *4th International Workshop on Software and performance.* Redwood Shores, CA, USA, 2004.

Xiao, K., et al. "A workflow-based non-intrusive approach for enhancing the survivability of critical infrastructures in cyber environment." *Proceedings of the 3rd International Workshop on Software Engineering for Secure Systems.* Washington, DC, USA, 2007. 20-26.

Ye, N., C. Newman, and T. Farley. "A system-fault-risk framework for cyber attack classification." *Information, Knowledge, Systems Management, vol. 5, no. 2*, 2005: 135-151.

Ye, Nong. "QoS-centric stateful resource management in information systems." *Information Systems Frontiers, vol. 4, no. 2*, 2002: 149-160.

Ye, Nong. *Secure computer and networks systems: Modeling, analysis and design.* London, UK: John Wiley & Sons Ltd, 2008.

Ye, Nong, Stephen S. Yau, Dazhi Huang, Mustafa Baydogan, Billibaldo Martinez Aranda, and Auttawut Roontiva. "Cause-effect dynamics of computer and network systems for QoS." *Industrial Engineering Research Conference.* Cancun, Mexico: A. Jhonson and J. Milller eds., 2010.

Yi, X., and Y. Zhang. "Survivability of information systems." *Procceedings of the 5th International Conference on Information, Communications and Signal Processing.* Bangkok, Thailand, 2005. 1551-1555.

Yu, H., D. Zheng, B. Y. Zhao, and W. Zheng. "Understanding user behavior in large-scale video on demand systems." *European Conference in Computer Systems.* Leuven, Belgium, 2006. 333-344.

Yu, Hongliang, Dongdong Zheng, Ben Y. Zhao, and Weimin Zheng. "Understanding user behavior in large-scale video on demand systems." *European Conference on Computer Systems, vol. 40, no. 4*, 2006.

Zhang, F., P. K. Verma, and S. Cheng. "Pricing, resource allocation and quality of service in multi-class networks with competitive market model." *IET Communications, vol. 5, no. 1*, 2011: 51-60.

Zhang, L. J., W. Wang, L. Guo, W. Yang, and Y. T. Yang. "A survivability quantitative analysis model for network system based on attack graph." *Proceedings of the 6th International Conference on Machine Learning and Cybernetics.* Hong Kong, 2007. 19-22.

Zhang, Liang-Jie, Jia Zhang, and Hong Cai. *Services Computing.* New York, USA: Springer, 2007.

Zhang, R., A. Bivens, and I. Rezek. "Efficient statistical performance modeling for autonomic, service-oriented systems." *IEEE International Parallel and Distributed Symposium.* Long beach, CA, USA., 2007. 1-10.

Zuo, Y., and B. Panda. "Unifying strategies and tactics: a survivability framework for countering cyber attacks." *Proceedings of the IEEE International Conference on Intelligence and Security Informatics.* Dallas, Texas, USA, 2009. 119-1124.

APPENDIX

SERVICE PROFILES OF EXPERIMENTAL CONDITIONS FOR PROCESSOR AND DISK MODELS

Tables A1-A3 show the service profiles for each of the experimental conditions (cases) run for the processor model. Cases were run first using RRP and then using MLF algorithms.

Table A1. Service Profiles for processor experiments, 2-Services competition.

		Service Profile 1		**Service Profile 2**	
Case	***WI***	**Arrival Dist.**	**Service Dist.**	**Arrival Dist.**	**Service Dist.**
1	0.5	expo(0.04)	expo(0.01)	expo(0.02)	expo(0.005)
2	0.5	expo(0.04)	expo(0.005)	expo(0.02)	expo(0.0075)
3	0.7	expo(0.04)	expo(0.01)	expo(0.02)	expo(0.009)
4	0.7	expo(0.04)	expo(0.005)	expo(0.02)	expo(0.0115)
5	0.9	expo(0.04)	expo(0.02)	expo(0.02)	expo(0.008)
6	0.9	expo(0.04)	expo(0.005)	expo(0.02)	expo(0.0155)
7	0.5	expo(0.04)	norm(0.01,0.0033)	expo(0.02)	norm(0.005,0.0017)
8	0.5	expo(0.04)	norm(0.005,0.0017)	expo(0.02)	norm(0.0075,0.0025)
9	0.7	expo(0.04)	norm(0.01,0.0033)	expo(0.02)	norm(0.009,0.003)
10	0.7	expo(0.04)	norm(0.005,0.0017)	expo(0.02)	norm(0.0115,0.0038)
11	0.9	expo(0.04)	norm(0.02,0.0067)	expo(0.02)	norm(0.008,0.0027)
12	0.9	expo(0.04)	norm(0.005,0.0017)	expo(0.02)	norm(0.0155,0.0052)
13	0.5	norm(0.04,0.0133)	expo(0.01)	norm(0.02,0.0067)	expo(0.005)
14	0.5	norm(0.04,0.0133)	expo(0.005)	norm(0.02,0.0067)	expo(0.0075)
15	0.7	norm(0.04,0.0133)	expo(0.01)	norm(0.02,0.0067)	expo(0.009)
16	0.7	norm(0.04,0.0133)	expo(0.005)	norm(0.02,0.0067)	expo(0.0115)
17	0.9	norm(0.04,0.0133)	expo(0.02)	norm(0.02,0.0067)	expo(0.008)
18	0.9	norm(0.04,0.0133)	expo(0.005)	norm(0.02,0.0067)	expo(0.0155)
19	0.5	norm(0.04,0.0133)	norm(0.01,0.0033)	norm(0.02,0.0067)	norm(0.005,0.0017)
20	0.5	norm(0.04,0.0133)	norm(0.005,0.0017)	norm(0.02,0.0067)	norm(0.0075,0.0025)
21	0.7	norm(0.04,0.0133)	norm(0.01,0.0033)	norm(0.02,0.0067)	norm(0.009,0.003)
22	0.7	norm(0.04,0.0133)	norm(0.005,0.0017)	norm(0.02,0.0067)	norm(0.0115,0.0038)
23	0.9	norm(0.04,0.0133)	norm(0.02,0.0067)	norm(0.02,0.0067)	norm(0.008,0.0027)
24	0.9	norm(0.04,0.0133)	norm(0.005,0.0017)	norm(0.02,0.0067)	norm(0.0155,0.0052)

Table A2. Service Profiles for processor experiments, 5-Services competition.

		Service Profile 1		Service Profile 2	
Case	***WI***	**Arrival Dist.**	**Service Dist.**	**Arrival Dist.**	**Service Dist.**
25	0.5	expo(0.12)	expo(0.012)	expo(0.1)	expo(0.009)
26	0.5	expo(0.12)	expo(0.004)	expo(0.1)	expo(0.006)
27	0.7	expo(0.12)	expo(0.012)	expo(0.1)	expo(0.011)
28	0.7	expo(0.12)	expo(0.008)	expo(0.1)	expo(0.009)
29	0.9	expo(0.12)	expo(0.018)	expo(0.1)	expo(0.016)
30	0.9	expo(0.12)	expo(0.008)	expo(0.1)	expo(0.01)
31	0.5	expo(0.12)	norm(0.012,0.004)	expo(0.1)	norm(0.009,0.003)
32	0.5	expo(0.12)	norm(0.004,0.0013)	expo(0.1)	norm(0.006,0.002)
33	0.7	expo(0.12)	norm(0.012,0.004)	expo(0.1)	norm(0.011,0.0037)
34	0.7	expo(0.12)	norm(0.008,0.0027)	expo(0.1)	norm(0.009,0.003)
35	0.9	expo(0.12)	norm(0.018,0.006)	expo(0.1)	norm(0.016,0.0053)
36	0.9	expo(0.12)	norm(0.008,0.0027)	expo(0.1)	norm(0.01,0.0033)
37	0.5	norm(0.12,0.04)	expo(0.012)	norm(0.1,0.0333)	expo(0.009)
38	0.5	norm(0.12,0.04)	expo(0.004)	norm(0.1,0.0333)	expo(0.006)
39	0.7	norm(0.12,0.04)	expo(0.012)	norm(0.1,0.0333)	expo(0.011)
40	0.7	norm(0.12,0.04)	expo(0.008)	norm(0.1,0.0333)	expo(0.009)
41	0.9	norm(0.12,0.04)	expo(0.018)	norm(0.1,0.0333)	expo(0.016)
42	0.9	norm(0.12,0.04)	expo(0.008)	norm(0.1,0.0333)	expo(0.01)
43	0.5	norm(0.12,0.04)	norm(0.012,0.004)	norm(0.1,0.0333)	norm(0.009,0.003)
44	0.5	norm(0.12,0.04)	norm(0.004,0.0013)	norm(0.1,0.0333)	norm(0.006,0.002)
45	0.7	norm(0.12,0.04)	norm(0.012,0.004)	norm(0.1,0.0333)	norm(0.011,0.0037)
46	0.7	norm(0.12,0.04)	norm(0.008,0.0027)	norm(0.1,0.0333)	norm(0.009,0.003)
47	0.9	norm(0.12,0.04)	norm(0.018,0.006)	norm(0.1,0.0333)	norm(0.016,0.0053)
48	0.9	norm(0.12,0.04)	norm(0.008,0.0027)	norm(0.1,0.0333)	norm(0.01,0.0033)
Cases	***WI***	**Service Profile 3**		**Service Profile 4**	
25	0.5	expo(0.08)	expo(0.008)	expo(0.06)	expo(0.006)
26	0.5	expo(0.08)	expo(0.0065)	expo(0.06)	expo(0.007)
27	0.7	expo(0.08)	expo(0.01)	expo(0.06)	expo(0.009)
28	0.7	expo(0.08)	expo(0.009)	expo(0.06)	expo(0.0095)
29	0.9	expo(0.08)	expo(0.014)	expo(0.06)	expo(0.012)
30	0.9	expo(0.08)	expo(0.012)	expo(0.06)	expo(0.014)
31	0.5	expo(0.08)	norm(0.008,0.0027)	expo(0.06)	norm(0.006,0.002)
32	0.5	expo(0.08)	norm(0.0065,0.0022)	expo(0.06)	norm(0.007,0.0023)
33	0.7	expo(0.08)	norm(0.01,0.0033)	expo(0.06)	norm(0.009,0.003)
34	0.7	expo(0.08)	norm(0.009,0.003)	expo(0.06)	norm(0.0095,0.0032)
35	0.9	expo(0.08)	norm(0.014,0.0047)	expo(0.06)	norm(0.012,0.004)
36	0.9	expo(0.08)	norm(0.012,0.004)	expo(0.06)	norm(0.014,0.0047)
37	0.5	norm(0.08,0.0267)	expo(0.008)	norm(0.06,0.02)	expo(0.006)
38	0.5	norm(0.08,0.0267)	expo(0.0065)	norm(0.06,0.02)	expo(0.007)
39	0.7	norm(0.08,0.0267)	expo(0.01)	norm(0.06,0.02)	expo(0.009)
40	0.7	norm(0.08,0.0267)	expo(0.009)	norm(0.06,0.02)	expo(0.0095)
41	0.9	norm(0.08,0.0267)	expo(0.014)	norm(0.06,0.02)	expo(0.012)
42	0.9	norm(0.08,0.0267)	expo(0.012)	norm(0.06,0.02)	expo(0.014)
43	0.5	norm(0.08,0.0267)	norm(0.008,0.0027)	norm(0.06,0.02)	norm(0.006,0.002)
44	0.5	norm(0.08,0.0267)	norm(0.0065,0.0022)	norm(0.06,0.02)	norm(0.007,0.0023)
45	0.7	norm(0.08,0.0267)	norm(0.01,0.0033)	norm(0.06,0.02)	norm(0.009,0.003)

46	0.7	norm(0.08,0.0267)	norm(0.009,0.003)	norm(0.06,0.02)	norm(0.0095,0.0032)
47	0.9	norm(0.08,0.0267)	norm(0.014,0.0047)	norm(0.06,0.02)	norm(0.012,0.004)
48	0.9	norm(0.08,0.0267)	norm(0.012,0.004)	norm(0.06,0.02)	norm(0.014,0.0047)
Cases	***WI***	**Service Profile 5**			
25	0.5	expo(0.04)	expo(0.0044)		
26	0.5	expo(0.04)	expo(0.0084)		
27	0.7	expo(0.04)	expo(0.0086)		
28	0.7	expo(0.04)	expo(0.0109)		
29	0.9	expo(0.04)	expo(0.0086)		
30	0.9	expo(0.04)	expo(0.014)		
31	0.5	expo(0.04)	norm(0.0044,0.0015)		
32	0.5	expo(0.04)	norm(0.00835,0.0028)		
33	0.7	expo(0.04)	norm(0.0086,0.0029)		
34	0.7	expo(0.04)	norm(0.0109,0.0036)		
35	0.9	expo(0.04)	norm(0.0086,0.0029)		
36	0.9	expo(0.04)	norm(0.014,0.0047)		
37	0.5	norm(0.04,0.0133)	expo(0.0044)		
38	0.5	norm(0.04,0.0133)	expo(0.00835)		
39	0.7	norm(0.04,0.0133)	expo(0.0086)		
40	0.7	norm(0.04,0.0133)	expo(0.0109)		
41	0.9	norm(0.04,0.0133)	expo(0.0086)		
42	0.9	norm(0.04,0.0133)	expo(0.014)		
43	0.5	norm(0.04,0.0133)	norm(0.0044,0.0015)		
44	0.5	norm(0.04,0.0133)	norm(0.00835,0.0028)		
45	0.7	norm(0.04,0.0133)	norm(0.0086,0.0029)		
46	0.7	norm(0.04,0.0133)	norm(0.0109,0.0036)		
47	0.9	norm(0.04,0.0133)	norm(0.0086,0.0029)		
48	0.9	norm(0.04,0.0133)	norm(0.014,0.0047)		

Table A3. Service Profiles for processor experiments, 10-Services competition.

		Service Profile 1		**Service Profile 2**	
Cases	***WI***	**Arrival Dist.**	**Service Dist.**	**Arrival Dist.**	**Service Dist.**
49	0.5	expo(0.15)	expo(0.008)	expo(0.14)	expo(0.007)
50	0.5	expo(0.15)	expo(0.003)	expo(0.14)	expo(0.003)
51	0.7	expo(0.15)	expo(0.01)	expo(0.14)	expo(0.01)
52	0.7	expo(0.15)	expo(0.005)	expo(0.14)	expo(0.005)
53	0.9	expo(0.15)	expo(0.012)	expo(0.14)	expo(0.012)
54	0.9	expo(0.15)	expo(0.007)	expo(0.14)	expo(0.008)
55	0.5	expo(0.15)	norm(0.008,0.0027)	expo(0.14)	norm(0.007,0.0023)
56	0.5	expo(0.15)	norm(0.003,0.001)	expo(0.14)	norm(0.003,0.001)
57	0.7	expo(0.15)	norm(0.01,0.0033)	expo(0.14)	norm(0.01,0.0033)
58	0.7	expo(0.15)	norm(0.005,0.0017)	expo(0.14)	norm(0.005,0.0017)
59	0.9	expo(0.15)	norm(0.012,0.004)	expo(0.14)	norm(0.012,0.004)
60	0.9	expo(0.15)	norm(0.007,0.0023)	expo(0.14)	norm(0.008,0.0027)

61	0.5	norm(0.15,0.05)	expo(0.008)	norm(0.14,0.0467)	expo(0.007)
62	0.5	norm(0.15,0.05)	expo(0.003)	norm(0.14,0.0467)	expo(0.003)
63	0.7	norm(0.15,0.05)	expo(0.01)	norm(0.14,0.0467)	expo(0.01)
64	0.7	norm(0.15,0.05)	expo(0.005)	norm(0.14,0.0467)	expo(0.005)
65	0.9	norm(0.15,0.05)	expo(0.012)	norm(0.14,0.0467)	expo(0.012)
66	0.9	norm(0.15,0.05)	expo(0.007)	norm(0.14,0.0467)	expo(0.008)
67	0.5	norm(0.15,0.05)	norm(0.008,0.0027)	norm(0.14,0.0467)	norm(0.007,0.0023)
68	0.5	norm(0.15,0.05)	norm(0.003,0.001)	norm(0.14,0.0467)	norm(0.003,0.001)
69	0.7	norm(0.15,0.05)	norm(0.01,0.0033)	norm(0.14,0.0467)	norm(0.01,0.0033)
70	0.7	norm(0.15,0.05)	norm(0.005,0.0017)	norm(0.14,0.0467)	norm(0.005,0.0017)
71	0.9	norm(0.15,0.05)	norm(0.012,0.004)	norm(0.14,0.0467)	norm(0.012,0.004)
72	0.9	norm(0.15,0.05)	norm(0.007,0.0023)	norm(0.14,0.0467)	norm(0.008,0.0027)
Cases	***WI***	**Service Profile 3**		**Service Profile 4**	
49	0.5	expo(0.14)	expo(0.007)	expo(0.12)	expo(0.006)
50	0.5	expo(0.14)	expo(0.004)	expo(0.12)	expo(0.004)
51	0.7	expo(0.14)	expo(0.009)	expo(0.12)	expo(0.009)
52	0.7	expo(0.14)	expo(0.006)	expo(0.12)	expo(0.006)
53	0.9	expo(0.14)	expo(0.011)	expo(0.12)	expo(0.01)
54	0.9	expo(0.14)	expo(0.008)	expo(0.12)	expo(0.009)
55	0.5	expo(0.14)	norm(0.007,0.0023)	expo(0.12)	norm(0.006,0.002)
56	0.5	expo(0.14)	norm(0.004,0.0013)	expo(0.12)	norm(0.004,0.0013)
57	0.7	expo(0.14)	norm(0.009,0.003)	expo(0.12)	norm(0.009,0.003)
58	0.7	expo(0.14)	norm(0.006,0.002)	expo(0.12)	norm(0.006,0.002)
59	0.9	expo(0.14)	norm(0.011,0.0037)	expo(0.12)	norm(0.01,0.0033)
60	0.9	expo(0.14)	norm(0.008,0.0027)	expo(0.12)	norm(0.009,0.003)
61	0.5	norm(0.14,0.0467)	expo(0.007)	norm(0.12,0.04)	expo(0.006)
62	0.5	norm(0.14,0.0467)	expo(0.004)	norm(0.12,0.04)	expo(0.004)
63	0.7	norm(0.14,0.0467)	expo(0.009)	norm(0.12,0.04)	expo(0.009)
64	0.7	norm(0.14,0.0467)	expo(0.006)	norm(0.12,0.04)	expo(0.006)
65	0.9	norm(0.14,0.0467)	expo(0.011)	norm(0.12,0.04)	expo(0.01)
66	0.9	norm(0.14,0.0467)	expo(0.008)	norm(0.12,0.04)	expo(0.009)
67	0.5	norm(0.14,0.0467)	norm(0.007,0.0023)	norm(0.12,0.04)	norm(0.006,0.002)
68	0.5	norm(0.14,0.0467)	norm(0.004,0.0013)	norm(0.12,0.04)	norm(0.004,0.0013)
69	0.7	norm(0.14,0.0467)	norm(0.009,0.003)	norm(0.12,0.04)	norm(0.009,0.003)
70	0.7	norm(0.14,0.0467)	norm(0.006,0.002)	norm(0.12,0.04)	norm(0.006,0.002)
71	0.9	norm(0.14,0.0467)	norm(0.011,0.0037)	norm(0.12,0.04)	norm(0.01,0.0033)
72	0.9	norm(0.14,0.0467)	norm(0.008,0.0027)	norm(0.12,0.04)	norm(0.009,0.003)
Cases	***WI***	**Service Profile 5**		**Service Profile 6**	
49	0.5	expo(0.12)	expo(0.006)	expo(0.1)	expo(0.005)
50	0.5	expo(0.12)	expo(0.005)	expo(0.1)	expo(0.005)

51	0.7	expo(0.12)	expo(0.008)	expo(0.1)	expo(0.008)
52	0.7	expo(0.12)	expo(0.007)	expo(0.1)	expo(0.007)
53	0.9	expo(0.12)	expo(0.01)	expo(0.1)	expo(0.009)
54	0.9	expo(0.12)	expo(0.009)	expo(0.1)	expo(0.0095)
55	0.5	expo(0.12)	norm(0.006,0.002)	expo(0.1)	norm(0.005,0.0017)
56	0.5	expo(0.12)	norm(0.005,0.0017)	expo(0.1)	norm(0.005,0.0017)
57	0.7	expo(0.12)	norm(0.008,0.0027)	expo(0.1)	norm(0.008,0.0027)
58	0.7	expo(0.12)	norm(0.007,0.0023)	expo(0.1)	norm(0.007,0.0023)
59	0.9	expo(0.12)	norm(0.01,0.0033)	expo(0.1)	norm(0.009,0.003)
60	0.9	expo(0.12)	norm(0.009,0.003)	expo(0.1)	norm(0.0095,0.0032)
61	0.5	norm(0.12,0.04)	expo(0.006)	norm(0.1,0.0333)	expo(0.005)
62	0.5	norm(0.12,0.04)	expo(0.005)	norm(0.1,0.0333)	expo(0.005)
63	0.7	norm(0.12,0.04)	expo(0.008)	norm(0.1,0.0333)	expo(0.008)
64	0.7	norm(0.12,0.04)	expo(0.007)	norm(0.1,0.0333)	expo(0.007)
65	0.9	norm(0.12,0.04)	expo(0.01)	norm(0.1,0.0333)	expo(0.009)
66	0.9	norm(0.12,0.04)	expo(0.009)	norm(0.1,0.0333)	expo(0.0095)
67	0.5	norm(0.12,0.04)	norm(0.006,0.002)	norm(0.1,0.0333)	norm(0.005,0.0017)
68	0.5	norm(0.12,0.04)	norm(0.005,0.0017)	norm(0.1,0.0333)	norm(0.005,0.0017)
69	0.7	norm(0.12,0.04)	norm(0.008,0.0027)	norm(0.1,0.0333)	norm(0.008,0.0027)
70	0.7	norm(0.12,0.04)	norm(0.007,0.0023)	norm(0.1,0.0333)	norm(0.007,0.0023)
71	0.9	norm(0.12,0.04)	norm(0.01,0.0033)	norm(0.1,0.0333)	norm(0.009,0.003)
72	0.9	norm(0.12,0.04)	norm(0.009,0.003)	norm(0.1,0.0333)	norm(0.0095,0.0032)
Cases	***WI***	**Service Profile 7**		**Service Profile 8**	
49	0.5	expo(0.1)	expo(0.005)	expo(0.08)	expo(0.004)
50	0.5	expo(0.1)	expo(0.006)	expo(0.08)	expo(0.006)
51	0.7	expo(0.1)	expo(0.007)	expo(0.08)	expo(0.0065)
52	0.7	expo(0.1)	expo(0.008)	expo(0.08)	expo(0.008)
53	0.9	expo(0.1)	expo(0.009)	expo(0.08)	expo(0.009)
54	0.9	expo(0.1)	expo(0.0095)	expo(0.08)	expo(0.01)
55	0.5	expo(0.1)	norm(0.005,0.0017)	expo(0.08)	norm(0.004,0.0013)
56	0.5	expo(0.1)	norm(0.006,0.002)	expo(0.08)	norm(0.006,0.002)
57	0.7	expo(0.1)	norm(0.007,0.0023)	expo(0.08)	norm(0.0065,0.0022)
58	0.7	expo(0.1)	norm(0.008,0.0027)	expo(0.08)	norm(0.008,0.0027)
59	0.9	expo(0.1)	norm(0.009,0.003)	expo(0.08)	norm(0.009,0.003)
60	0.9	expo(0.1)	norm(0.0095,0.0032)	expo(0.08)	norm(0.01,0.0033)
61	0.5	norm(0.1,0.0333)	expo(0.005)	norm(0.08,0.0267)	expo(0.004)
62	0.5	norm(0.1,0.0333)	expo(0.006)	norm(0.08,0.0267)	expo(0.006)
63	0.7	norm(0.1,0.0333)	expo(0.007)	norm(0.08,0.0267)	expo(0.0065)
64	0.7	norm(0.1,0.0333)	expo(0.008)	norm(0.08,0.0267)	expo(0.008)
65	0.9	norm(0.1,0.0333)	expo(0.009)	norm(0.08,0.0267)	expo(0.009)

66	0.9	norm(0.1,0.0333)	expo(0.0095)	norm(0.08,0.0267)	expo(0.01)
67	0.5	norm(0.1,0.0333)	norm(0.005,0.0017)	norm(0.08,0.0267)	norm(0.004,0.0013)
68	0.5	norm(0.1,0.0333)	norm(0.006,0.002)	norm(0.08,0.0267)	norm(0.006,0.002)
69	0.7	norm(0.1,0.0333)	norm(0.007,0.0023)	norm(0.08,0.0267)	norm(0.0065,0.0022)
70	0.7	norm(0.1,0.0333)	norm(0.008,0.0027)	norm(0.08,0.0267)	norm(0.008,0.0027)
71	0.9	norm(0.1,0.0333)	norm(0.009,0.003)	norm(0.08,0.0267)	norm(0.009,0.003)
72	0.9	norm(0.1,0.0333)	norm(0.0095,0.0032)	norm(0.08,0.0267)	norm(0.01,0.0033)
Cases	***WI***	**Service Profile 9**		**Service Profile 10**	
49	0.5	expo(0.08)	expo(0.004)	expo(0.08)	expo(0.0037)
50	0.5	expo(0.08)	expo(0.006)	expo(0.08)	expo(0.0076)
51	0.7	expo(0.08)	expo(0.005)	expo(0.08)	expo(0.005)
52	0.7	expo(0.08)	expo(0.009)	expo(0.08)	expo(0.0094)
53	0.9	expo(0.08)	expo(0.008)	expo(0.08)	expo(0.0077)
54	0.9	expo(0.08)	expo(0.01)	expo(0.08)	expo(0.0119)
55	0.5	expo(0.08)	norm(0.004,0.0013)	expo(0.08)	norm(0.0037,0.0012)
56	0.5	expo(0.08)	norm(0.006,0.002)	expo(0.08)	norm(0.0076,0.0025)
57	0.7	expo(0.08)	norm(0.005,0.0017)	expo(0.08)	norm(0.005,0.0017)
58	0.7	expo(0.08)	norm(0.009,0.003)	expo(0.08)	norm(0.0094,0.0031)
59	0.9	expo(0.08)	norm(0.008,0.0027)	expo(0.08)	norm(0.0077,0.0026)
60	0.9	expo(0.08)	norm(0.01,0.0033)	expo(0.08)	norm(0.0119,0.004)
61	0.5	norm(0.08,0.0267)	expo(0.004)	norm(0.08,0.0267)	expo(0.0037)
62	0.5	norm(0.08,0.0267)	expo(0.006)	norm(0.08,0.0267)	expo(0.0076)
63	0.7	norm(0.08,0.0267)	expo(0.005)	norm(0.08,0.0267)	expo(0.005)
64	0.7	norm(0.08,0.0267)	expo(0.009)	norm(0.08,0.0267)	expo(0.0094)
65	0.9	norm(0.08,0.0267)	expo(0.008)	norm(0.08,0.0267)	expo(0.0077)
66	0.9	norm(0.08,0.0267)	expo(0.01)	norm(0.08,0.0267)	expo(0.0119)
67	0.5	norm(0.08,0.0267)	norm(0.004,0.0013)	norm(0.08,0.0267)	norm(0.0037,0.0012)
68	0.5	norm(0.08,0.0267)	norm(0.006,0.002)	norm(0.08,0.0267)	norm(0.0076,0.0025)
69	0.7	norm(0.08,0.0267)	norm(0.005,0.0017)	norm(0.08,0.0267)	norm(0.005,0.0017)
70	0.7	norm(0.08,0.0267)	norm(0.009,0.003)	norm(0.08,0.0267)	norm(0.0094,0.0031)
71	0.9	norm(0.08,0.0267)	norm(0.008,0.0027)	norm(0.08,0.0267)	norm(0.0077,0.0026)
72	0.9	norm(0.08,0.0267)	norm(0.01,0.0033)	norm(0.08,0.0267)	norm(0.0119,0.004)

Tables A4-A6 show the service profiles for each of the experimental conditions (cases) run for the disk model. Cases were run first using C-Look and then using SSTF algorithms.

Table A4. Service Profiles for disk experiments, 2-Services competition.

		Service Profile 1		Service Profile 2	
Case	***WI***	**Arrival Dist.**	**Block Size**	**Arrival Dist.**	**Block Size**
1	0.6	expo(0.07)	0.004	expo(0.034)	0.032
2	0.6	expo(0.06)	0.016	expo(0.037)	0.064
3	0.6	expo(0.05)	0.032	expo(0.043)	0.128
4	0.8	expo(0.065)	0.004	expo(0.023)	0.016
5	0.8	expo(0.055)	0.016	expo(0.025)	0.032
6	0.8	expo(0.045)	0.064	expo(0.029)	0.128
7	1	expo(0.04)	0.004	expo(0.021)	0.064
8	1	expo(0.03)	0.016	expo(0.026)	0.128
9	1	expo(0.02)	0.032	expo(0.045)	0.064
10	1.2	expo(0.035)	0.004	expo(0.017)	0.064
11	1.2	expo(0.025)	0.016	expo(0.022)	0.128
12	1.2	expo(0.015)	0.032	expo(0.049)	0.064
13	0.6	norm(0.07,0.0233)	0.004	norm(0.034,0.0113)	0.032
14	0.6	norm(0.06,0.02)	0.016	norm(0.037,0.0123)	0.064
15	0.6	norm(0.05,0.0167)	0.032	norm(0.043,0.0143)	0.128
16	0.8	norm(0.065,0.0217)	0.004	norm(0.023,0.0077)	0.016
17	0.8	norm(0.055,0.0183)	0.016	norm(0.025,0.0083)	0.032
18	0.8	norm(0.045,0.015)	0.064	norm(0.029,0.0097)	0.128
19	1	norm(0.04,0.0133)	0.004	norm(0.021,0.007)	0.064
20	1	norm(0.03,0.01)	0.016	norm(0.026,0.0087)	0.128
21	1	norm(0.02,0.0067)	0.032	norm(0.045,0.015)	0.064
22	1.2	norm(0.035,0.0117)	0.004	norm(0.017,0.0057)	0.064
23	1.2	norm(0.025,0.0083)	0.016	norm(0.022,0.0073)	0.128
24	1.2	norm(0.015,0.005)	0.032	norm(0.049,0.0163)	0.064

Table A5. Service Profiles for disk experiments, 5-Services competition.

		Service 1		Service 2	
Case	***WI***	**Arrival Dist.**	**Block Size**	**Arrival Dist.**	**Block Size**
25	0.6	expo(0.11)	0.004	expo(0.11)	0.016
26	0.8	expo(0.09)	0.004	expo(0.09)	0.016
27	1	expo(0.08)	0.004	expo(0.08)	0.016
28	1.2	expo(0.06)	0.004	expo(0.06)	0.016
29	0.6	norm(0.11,0.0367)	0.004	norm(0.11,0.0367)	0.016
30	0.8	norm(0.09,0.03)	0.004	norm(0.09,0.03)	0.016
31	1	norm(0.08,0.0267)	0.004	norm(0.08,0.0267)	0.016
32	1.2	norm(0.06,0.02)	0.004	norm(0.06,0.02)	0.016
Case	***WI***	**Service 3**		**Service 4**	
25	0.6	expo(0.11)	0.032	expo(0.11)	0.064
26	0.8	expo(0.09)	0.032	expo(0.09)	0.064
27	1	expo(0.08)	0.032	expo(0.08)	0.064
28	1.2	expo(0.06)	0.032	expo(0.06)	0.064
29	0.6	norm(0.11,0.0367)	0.032	norm(0.11,0.0367)	0.064
30	0.8	norm(0.09,0.03)	0.032	norm(0.09,0.03)	0.064
31	1	norm(0.08,0.0267)	0.032	norm(0.08,0.0267)	0.064
32	1.2	norm(0.06,0.02)	0.032	norm(0.06,0.02)	0.064
Case	***WI***	**Service 5**			
25	0.6	expo(0.141)	0.128		
26	0.8	expo(0.075)	0.128		
27	1	expo(0.045)	0.128		
28	1.2	expo(0.05)	0.128		
29	0.6	norm(0.141,0.047)	0.128		
30	0.8	norm(0.075,0.025)	0.128		
31	1	norm(0.045,0.015)	0.128		
32	1.2	norm(0.05,0.0167)	0.128		

Table A6. Service Profiles for disk experiments, 10-Services competition.

		Service 1		Service 2	
Case	***WI***	**Arrival Dist.**	**Block Size**	**Arrival Dist.**	**Block Size**
33	0.6	expo(0.23)	0.004	expo(0.23)	0.008
34	0.8	expo(0.18)	0.004	expo(0.18)	0.008
35	1	expo(0.16)	0.004	expo(0.16)	0.008
36	1.2	expo(0.12)	0.004	expo(0.12)	0.008
37	0.6	norm(0.23,0.0767)	0.004	norm(0.23,0.0767)	0.008
38	0.8	norm(0.18,0.06)	0.004	norm(0.18,0.06)	0.008
39	1	norm(0.16,0.0533)	0.004	norm(0.16,0.0533)	0.008
40	1.2	norm(0.12,0.04)	0.004	norm(0.12,0.04)	0.008
Case	***WI***	**Service 3**		**Service 4**	
33	0.6	expo(0.23)	0.016	expo(0.23)	0.032
34	0.8	expo(0.18)	0.016	expo(0.18)	0.032
35	1	expo(0.16)	0.016	expo(0.16)	0.032
36	1.2	expo(0.12)	0.016	expo(0.12)	0.032
37	0.6	norm(0.23,0.0767)	0.016	norm(0.23,0.0767)	0.032
38	0.8	norm(0.18,0.06)	0.016	norm(0.18,0.06)	0.032
39	1	norm(0.16,0.0533)	0.016	norm(0.16,0.0533)	0.032
40	1.2	norm(0.12,0.04)	0.016	norm(0.12,0.04)	0.032
Case	***WI***	**Service 5**		**Service 6**	
33	0.6	expo(0.23)	0.064	expo(0.23)	0.128
34	0.8	expo(0.18)	0.064	expo(0.18)	0.128
35	1	expo(0.16)	0.064	expo(0.16)	0.128
36	1.2	expo(0.12)	0.064	expo(0.12)	0.128
37	0.6	norm(0.23,0.0767)	0.064	norm(0.23,0.0767)	0.128
38	0.8	norm(0.18,0.06)	0.064	norm(0.18,0.06)	0.128
39	1	norm(0.16,0.0533)	0.064	norm(0.16,0.0533)	0.128
40	1.2	norm(0.12,0.04)	0.064	norm(0.12,0.04)	0.128
Case	***WI***	**Service 7**		**Service 8**	
33	0.6	expo(0.23)	0.256	expo(0.23)	0.512
34	0.8	expo(0.18)	0.256	expo(0.18)	0.512
35	1	expo(0.16)	0.256	expo(0.16)	0.512
36	1.2	expo(0.12)	0.256	expo(0.12)	0.512
37	0.6	norm(0.23,0.0767)	0.256	norm(0.23,0.0767)	0.512
38	0.8	norm(0.18,0.06)	0.256	norm(0.18,0.06)	0.512
39	1	norm(0.16,0.0533)	0.256	norm(0.16,0.0533)	0.512
40	1.2	norm(0.12,0.04)	0.256	norm(0.12,0.04)	0.512
Case	***WI***	**Service 9**		**Service 10**	
33	0.6	expo(0.23)	0.768	expo(0.432)	1.024
34	0.8	expo(0.18)	0.768	expo(0.204)	1.024
35	1	expo(0.16)	0.768	expo(0.088)	1.024
36	1.2	expo(0.12)	0.768	expo(0.136)	1.024
37	0.6	norm(0.23,0.0767)	0.768	norm(0.432,0.144)	1.024
38	0.8	norm(0.18,0.06)	0.768	norm(0.204,0.068)	1.024
39	1	norm(0.16,0.0533)	0.768	norm(0.088,0.0293)	1.024
40	1.2	norm(0.12,0.04)	0.768	norm(0.136,0.0453)	1.024