Visualizing the underlying trends of component latencies affecting service operation performance.

Sid Kargupta
Sue Black

School of Electronics and Computer Science

Copyright © [2009] IEEE. Reprinted from ACTEA 2009: 2009 International Conference on Advances in Computational Tools for Engineering Applications, Zouk Mosbeh, Lebanon, 14 - 23 July 2009. IEEE, pp. 565-568. ISBN 9781424438334.

Visualizing the underlying Trends of Component Latencies affecting Service Operation Performance

Sid Kargupta[1], Member *IEEE* and Sue Black[2]
[1]EMC Consulting, London SE1 9EU, UK
[2]Dept. of Information & Software Systems, University of Westminster, Harrow, Middx HA1 3TP, UK

Abstract - **This paper presents a technology agnostic method for extracting the underlying distinct patterns of variations in the overall Performance of a Service Operation for changes to different application components supporting the Service Operation in a computer based Service Provider-Consumer contract. This short paper advocates that visualizing these patterns would help in early projection of the operation's performance due to modification of the application components/processing catering to the operation, without the need of repetitive performance and load testing of the whole service. Lookup datasets against different component configurations are created to associate the variability of component processing impedances to the service operation's Performance and best fit regression types are applied to enable trend extrapolation and interpolation.**

I. INTRODUCTION

Service Operations of a Service Provider are supported by underlying application components hosted on top of a layer of system components. For every Service Operation, the *activities* of these components cumulatively impact the Performance of the operation. The application components are often modified due to changes in business requirements while the underlying system remains the same. This research verifies a higher level pattern based projection of certain non-functional features of a Service Operation for modifications to the supporting application components. The Service Operation's supporting application components are decomposed into atomic activities or Delay Points (activities introducing delay to the overall performance) like in-memory Data Processing, File I/O, Database Interaction, XML processing etc., which interface with the system resources and impact the overall performance of the Service Operation. This research attempts to highlight that the total delay (or *Impedance*) for each type of Delay Points across all the supporting application components influence the Service Operation's performance 'P' in a distinct pattern i.e.

$$P = f(\bullet I_{DLPi})\ [i=1 \text{ to } n]$$

where I_{DLP1} is the Impedance by a particular Delay Point of Component1. Atomicity of Delay Points is very important as Delay Point types determine their nature of system resource usage, which then manifests as the Delay Point impact pattern. Delay Points should not overlap. This research emphasizes on variations to application components and Delay Points instead of inbound workload.

II. PROBLEM STATEMENT AND MOTIVATION

The changes to underlying application components inevitably influence Service Performance. Significant research is undertaken to highlight lack of performance measuring provision, devise ways to measure efficiency of services and dynamic selection of services based on user specified preferences [1, 2, 3]. Further research [4, 5, 6] has explored data transport mechanisms, which are important factors in Web Service performances. Much research has been performed towards measuring and predicting throughput, response time and congestion using queuing network principles. Ways to model, analyze and plan for web performance problems have been illustrated in details [7]. High performance website design techniques involving redundant hardware, load balancing, web server acceleration and efficient management of dynamic data [8] have been discussed. Methods are devised for dynamic selection of services based on user specified preferences and to predict performance of component based services depending on the underlying technology platforms [9]. Further research [10, 11, 12] has explored various methods of component based performance evaluation with *top-down* approach focusing on inbound workload, profiling, software containers, UMLs and transactions. However, we lack in visual pattern aided methods to determine impact of the underlying application level component modifications on service performance. Often application developers find it convenient to analyze application level outputs than system resource or service level diagnostics, for which other human resources are required. Hence, it will be helpful to explore *generic*, application level, *bottom-up* methods to assess during development the impact of application component modifications at Delay Point granularity on other non-functional features.

III. AIMS AND OBJECTIVES

Typically, impact analysis of modifications to systems/services involves rigorous load/performance testing and validation. Services may comprise of various types of underlying application components. These components may

978-1-4244-3834-1/09/$25.00 © 2009 IEEE

perform different activities like complex data processing, intense file I/O, database interactions and others. Based on the premise that under a given load and platform condition, these different components obey distinct patterns in impacting service operation performance, this paper attempts to extract and highlight the following through empirical results:

- The *distinct* patterns by which the different Delay Point processing times of the components (aka Delay Point Impedances I_{DLP}) vary with variations to their respective processing intensities (PI_{DLP}).
- The distinct patterns by which the different Delay Point Impedances of the components impact overall Service Operation Performance.

The objective was to create lookup datasets and graphs corresponding to the different system configurations to facilitate extrapolation/interpolation of data. This was to project the effect of systems modifications on overall performance and also facilitate fine tuning of modified components to maintain Quality of Service (QoS). Performance was tested against the variability of the Delay Points of application components. We believe this pattern based approach will lead to better systems/underlying application design upfront.

IV. PROPOSED METHODOLOGY

To increase precision of the model and *standardize* request resource requirements, partitioning of the request load is achieved by constraining the model and method to Service Operation level. Different Service Operations from the same Provider may have different resource requirements.

A Java based Service Framework comprising of Web Services, Servlets, RMI Server, Socket Server, a multi-threaded Web Service Client, utility components and external configuration files was created to simulate a Service Provider – Consumer contract with provision to vary the backend services' processing. The patterns are measured at a Service Operation level to maintain parity of type and volume of data across requests. The experiments are performed under nearly identical load conditions with co-located application components (i.e. local inter-component calls) for precision in pattern detection. The scope of the proposed method is confined to formal Service Contracts with Services accessed over dedicated networks (not public network) with controlled network traffic. Hence, at runtime, there should not be any unpredictable fluctuation of network bandwidth or latency impacting performance.

A. Pattern Extraction

A series of experiments were performed by incrementally varying the processing intensities at various Delay Points of the Service Provider while keeping the rest of the configuration the same, observing all the constraints mentioned previously. For the purpose of the experiments, some illustrative Delay Points with activities like Data Processing, File I/O, Request Authentication and Request Authorization involving XML parsing etc. were created. Empirical data for the variations of overall Service Performance (P) against variability of Impedances of different Delay Points (I_{DLP}) at the different application components was collected. For every configuration, the Delay Point's performance factor ($PF_{DLP} = P/I_{DLP}$) was recorded. As nominal error factor is acceptable, for simplifying the model, we looked to applying the best fit regression types or Piecewise Linear models on the collected data. The different regression types verified were *Linear*, *Exponential*, *Polynomial* and *Power*. Data obtained for the different features substituted for x and y in respective models. As derived errors were nominal, we applied best fit regression types on the collected empirical data to model P, I_{DLP}, PI_{DLP} and PF_{DLP}. The consistent patterns affirmed that for a given service load band, from a projected PF_{DLP} for a modified component Delay Point's actual I_{DLP}, P could be projected by

$$P = PF_{DLP} \times I_{DLP} + e$$

where 'e' is the error factor. The results clearly revealed the *distinct* patterns in variations of Service Operation Performance for variations in each of the different Delay Point Impedances across the application components. It was also observed that different Delay Point Impedances obeyed *distinct* patterns of variability for changes to respective activity/processing intensities.

Observations were made by varying the levels of activities at the Data Processing, File I/O, Authentication and Authorization Delay Points.

B. Data Processing Pattern Extraction

Some illustrative components are created with Data Processing, File I/O, XML Processing and other Delay Points. Keeping the rest of the configuration constant, the Data Processing Delay Point intensities of the components were incrementally varied. Empirical data for *actual* overall 'P', the *actual* average Data Processing Delay Point impedance (I_{DP}) and the Data Processing Performance Factor ($PF_{DP} = P/I_{DP}$) was recorded. The following data models 'I_{DP}' versus 'P', 'I_{DP}' versus 'PF_{DP}' and 'PI_{DP}' versus 'I_{DP}' showed *distinct* trends in variation, which were consistent but *not* purely linear. Accepting approximation error, for simplicity, best fit regression types for *Linear*, *Exponential, Polynomial, Power* and also *Piecewise Linear* models were verified. For 'I_{DP}'(x_i) versus 'P'(y_i), pattern line with *Polynomial* regression of 3rd order was the best fit:

$$y_i = -4E{+}06x_i^3 + 86037x_i^2 - 791.72x_i + 4.4125$$

For 'I_{DP}'(x_i) versus 'PF_{DP}'(y_i) pattern line with *Power* regression was best fit:

$$y_i = Ax_i^B,\ B = b,\ A = e^a,\ a \text{ and } b \text{ are best fit coefficients}$$

For ‘PI_{DP}’(x_i) versus ‘I_{DP}’(y_i), pattern line with *Polynomial* regression of 3rd order was the best fit:

$$y_i = -4E\text{-}07x_i^3 + 2E\text{-}05x_i^2 + 0.0001x_i + 0.0004$$

Tests are performed to validate the extracted patterns. Results affirmed (with some approximation errors) the *distinct* underlying patterns of variations in ‘P’ due to changes in application components/Delay Points under a given load. From a *projected* value of ‘IF_{DP}’ corresponding to a given *actual* ‘I_{DP}’, we could also *project* ‘P’:

$$P = PF_{DP} \times I_{DP} + e$$

where ‘e’ is the error factor. Figures 1, 2 and 3 present the empirical graphs of ‘I_{DP}’ versus ‘P’, ‘I_{DP}’ versus ‘PF_{DP}’ and ‘PI_{DP}’ versus ‘I_{DP}’. Pattern validation is highlighted.

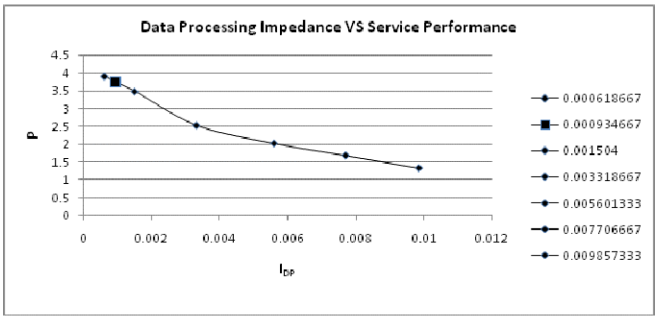

Figure 1: Empirical Data Graph for ‘I_{DP}’ versus ‘P’

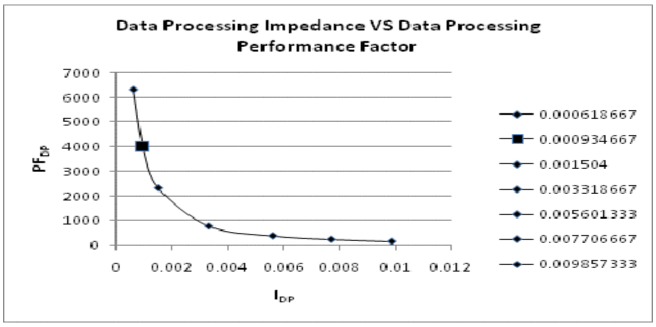

Figure 2: Empirical Data Graph for ‘I_{DP}’ versus ‘PF_{DP}’

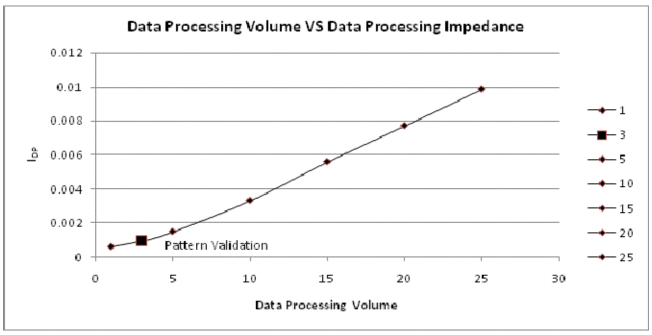

Figure 3: Empirical Data Graph for ‘PI_{DP}’ versus ‘I_{DP}’

C. File I/O Pattern Extraction

Keeping the rest of the configuration constant, the File I/O Delay Point intensities of the components were incrementally varied. Empirical data for *actual* overall ‘P’, the *actual* average File I/O Delay Point impedance (I_{FIO}) and the File I/O Performance Factor ($PF_{FIO} = P/I_{FIO}$) was recorded. As with Data Processing, *distinct* patterns are extracted for all the above non-functional features by varying the File I/O Delay Points. But the functions had *different* values from the Data Processing Delay Point patterns. For ‘I_{FIO}’(x_i) versus ‘P’(y_i) and ‘I_{FIO}’(x_i) versus ‘PF_{FIO}’(y_i) pattern line with *Power* regression was best fit. For ‘PI_{FIO}’(x_i) versus ‘I_{FIO}’(y_i), pattern line with *Polynomial* regression of 3rd order was the best fit:

$$y_i = 6E\text{-}06x_i^3 - 6E\text{-}05x_i^2 + 0.0673x_i - 0.1018$$

Tests are performed to validate the extracted patterns. Results affirmed (with some approximation errors) the *distinct* underlying patterns of variations in ‘P’ due to changes in the File I/O Delay Points across the application components under a given load. Figures 4, 5 and 6 present the empirical graphs of ‘I_{FIO}’ versus ‘P’, ‘I_{FIO}’ versus ‘PF_{FIO}’ and ‘PI_{FIO}’ versus ‘I_{FIO}’. Pattern validation is highlighted.

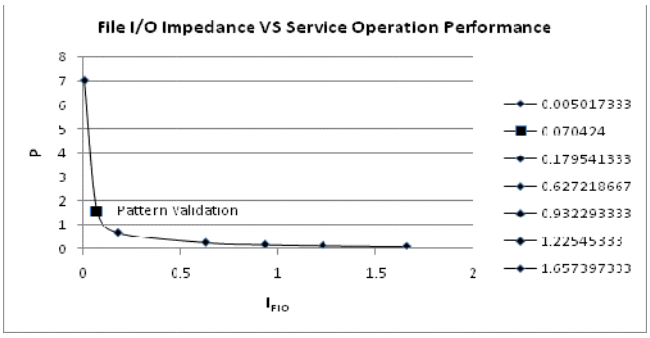

Figure 4: Empirical Data Graph for ‘I_{FIO}’ versus ‘P’

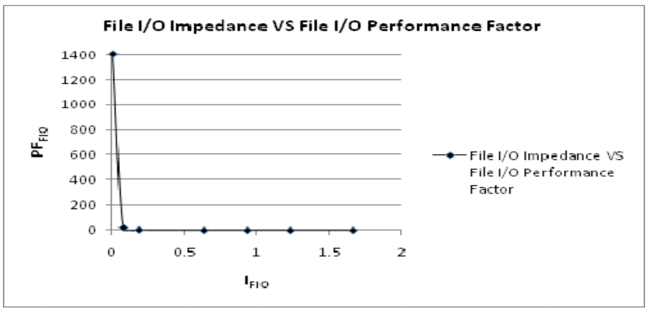

Figure 5: Empirical Data Graph for ‘I_{FIO}’ versus ‘PF_{FIO}’

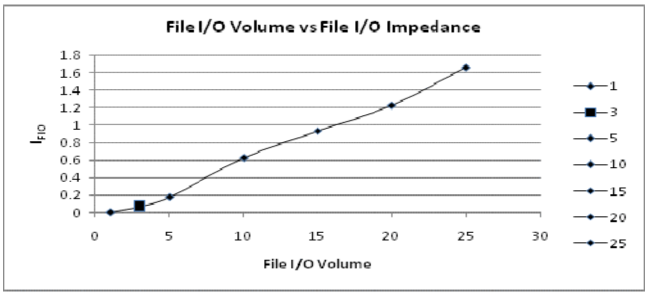

Figure 6: Empirical Data Graph for 'PI_{FIO}' versus 'I_{FIO}'

D. Authentication/Authorization Pattern Extraction

As with the Data Processing and File I/O Delay Points, the Authentication and Authorization Delay Point intensities of the components were incrementally varied as well. Empirical data for *actual* overall 'P', the *actual* average Authentication/Authorization Delay Point impedance (I_{AA}) and the Authentication/Authorization Performance Factor ($PF_{AA} = P/I_{AA}$) was recorded. As with the previous Delay Points, *distinct* patterns are extracted for all the above non-functional features by varying the Authentication / Authorization Delay Points.

V. DATA VALIDATION

After creating the datasets and graphical patterns from the experiments, tests were performed to validate the patterns. The empirical results proved the integrity of the previously obtained patterns. The results supported that one could interpolate (hence also extrapolate) the derived dataset/patterns to predict possible impact on Service Performances due to changes to application components during construction stage. The results demonstrated that *distinct* underlying patterns are obeyed for variations in Service Performance due to changes to underlying application components at a Service Operation level under a given load condition. The results also highlighted that variations of the different component Delay Point Impedances due to changes in processing intensities follow *distinct* trends.

VI. FURTHER WORK

For precision, Delay Point atomicity needs to be increased e.g. file type specific File I/O Delay Point. Model calibration needs to be verified. More Delay Points need to be tested e.g. database interaction/contention has not been verified yet. Delay Points were varied one type at a time but real world component modifications will be more complex with multiple Delay Point types modified simultaneously. For this, data patterns need to be created that will facilitate projection of 'P' for any arbitrary combination of Delay Point Impedances. Minimizing components system resource sharing by spreading the service framework would be good. All of these should enhance overall method precision.

REFERENCES

[1] Shiping Chen, Bo Yan, Zic J., Ren Liu and Nq A., Evaluation and Modeling of Web Services Performance, *Networking Technol. Lab., CSIRO ICT Centre,* Sept. 2006, 437-444.

[2] Menasce D.A., Performance and Availability of Internet Data Centers, *Internet Computing, IEEE.* George Mason University, Fairfax, VA, USA, May-June 2004, 94 - 96.

[3] D. Ardagna and B. Pernici, Adaptive Service Composition in Flexible Processes, *IEEE Transactions on Software Engineering, Vol. 33, No.6,* June 2007

[4] Alex Ng, Optimising Web Services Performance with Table Driven XML, *Proceedings of the 2006 Australian Software Engineering Conference (ASWEC'06) 1530-0803/06* © 2006 IEEE.

[5] Khoi Anh Phan, Zahir Tari, Peter Bertok, Optimizing Web Services Performance by Using Similarity-based Multicast Protocol, *Proceedings of the European Conference on Web Services (ECOWS'06) 0-7695-2737-X/06* © 2006.

[6] Tanakorn Wichaiwong and Chuleerat Jaruskulchai, A Simple Approach to Optimize Web Services Performance, *Proceedings of the Third International Conference on Next Generation Web Services Practices* 2007.

[7] Daniel A. Menasce, Virgilio A. F. Almeida: Capacity Planning for Web Services. Metrics, Models and Methods. Prentice Hall PTR, Upper Saddle River, NJ 07458, 2002.

[8] Arun Iyengar, Jim Challenger, Daniel Dias and Paul Dantzig: High Performance Web Site Design Techniques. Web Design, IEEE Internet Computing, March-April 2000.

[9] Yan Liu, Alan Fekete and Ian Gorton, Design-Level Performance Prediction of Component-Based Applications, IEEE Transactions on Software Engineering, Vol.31, No.11, 2005.

[10] Connie U. Smith and Lloyd G. Williams, Performance Engineering Evaluation of Object-Oriented Systems with SPE.ED, Computer Performance Evaluation: Modelling Techniques and Tools, No.1245,Springer-Verlag, Berlin 1997.

[11] Christopher Stewart and Kai Shen, Performance Modeling and System Management for Multi-component Online Services, Proceedings of the 2nd Symposium on Networked Systems Design and Implementation, May2-4, Boston, MA, USA, 2005.

[12] K.S. Jasmine and R. Vasantha, Design Based Performance Prediction of Component Based Software Products, Proceedings of the World Academy of Science, Engineering and Technology, Vol 24, ISSN 1307-6884, October 2007.