Silverlining: A Simulator to Forecast Cost and Performance in the Cloud

Lawrence Chung, The University of Texas at Dallas  
Nary Subramanian, The University of Texas at Tyler  
Thomas Hill, The University of Texas at Dallas  
Grace Park, The University of Texas at Dallas

Abstract. A key question for a Chief Information Officer (CIO) would be the future run-time cost and performance of complex business application software, before deciding to migrate it to a cloud. It is difficult for CIOs to accurately estimate cloud cost and performance in a fast and inexpensive manner. In this article, we describe “Silverlining”, a simulator for estimating the cost and performance of a cloud service before migration, to help the CIO not only with go/no-go decisions but also with the budgeting for an appropriate cloud configuration.

Introduction

A software system’s operational behavior can be characterized by not only its functional requirements - what the system does – but also its non-functional requirements (NFRs) - how usefully and useably the system executes its functions. To forecast the behavior of a software system in the cloud, we explore two primary run-time NFRs - cost and performance. The objective of the Silverlining Simulator is to predict the operational cost and performance of a system by building a model to imitate the operation of the software system under study. The simulation model needs a description of the basic cloud infrastructure topology (resource capacity) and a step-by-step depiction of the system operation workload (resource usage). The Google cloud infrastructure, employed in the case study, uses the following two primary classes of compute resources (shown in Figure 3): The Google App Engine (GAE) and the CloudSQL Database Engine. The CloudSQL Database Engine was simulated using three cloud configurations (Low-cost/low-power-CPU server - D1, medium-cost/medium-power-CPU server - D16 and high-cost/high-power-CPU server - D32). The low-cost/low-power-CPU server - D1 provided adequate throughput to satisfy the management’s goal.

Silverlining Simulation Process

In order to provide cloud forecasting capability, Silverlining - our simulator - must first be primed with appropriate base information (examples in Figures 4 and 5). For this purpose, cost and performance goals are obtained from stakeholder requirements for the system, oftentimes in terms of service level agreements (SLAs) (See Figure 1 Step1), and their interdependencies are analyzed by means of a notational convention, called Softgoal Interdependency Graph (SIG), which is intended for representing and reasoning about NFRs.

Next the characteristics of the intended software application is estimated (Step 2), e.g., using varying workloads. The characteristics of the intended software application are then loaded as input into the simulator (Step 3/4), and the simulator would output (Step 5) the cost and performance estimates for executing the software system, with varying workloads and cloud configurations, on the cloud. With proper adjustments for differences from the standard (Step6), the data from the simulator can be used to estimate the cost and performance of the cloud, as well as choosing among the available cloud configurations according to the particular cost and performance business goals that a CIO may have.

Now, an important question is if and how much Silverlining is reliable – i.e., the accuracy of the simulation results. For Silverlining, experiments for a typical application were run on Google cloud (called Google App Engine, or GAE), with varying workloads and cloud configurations (such as platforms and infrastructure characteristics), for a variety of benchmark data, and, using the same workloads and cloud configurations, the results from Silverlining were compared against the benchmark data. The comparison showed the two sets of data were very close for the typical application (or class of applications) that was used for the experiments. Of course, more experiments would be needed, in order for Silverlining to help a CIO and a cloud service provider assess and predict the cost and performance of a variety of (classes) of software applications, as well as choose among alternative cloud platforms and configurations, or, if needed, even adjust cost and performance business goals.

In this paper, a case study of a Vehicle Management System (VMS) Display-Status is presented, which has been in operation for almost three decades and will continue to be for many decades to come. This particular VMS is supposed to manage close to 100,000 vehicles, while carrying out a variety of tasks, such as keeping track of their locations and status (e.g., in normal operation or maintenance, or in emergency repair, moving or stationed), scheduling their routes, allocating crews, dispatching them, compiling statistics, reporting on work progress, etc.

Figure 1. Silverlining simulation modeling framework steps.
In the following, the process of using Silverlining for the case study is described in a piecemeal manner.

The Vehicle Management CIO is contemplating the migration of the VMS to the cloud. Among other things, two requirements are of critical concerns - the total system operation costs cannot exceed $3,000 per month and the Display-Status application must perform at 300 transactions per minute (tpm). The performance goals and VMS operational characteristics are used as input to the simulation forecaster. The CIO can run the simulation on a local laptop computer and see the cost and performance estimates immediately, i.e., without the time-consuming and costly development, tailoring, installation, configuration and testing the software system, as well as without (the use of) any real cloud or hardware equipment.

Step 1: Set stakeholder cost and performance goals
First, a VMS Display-Status application SIG (Softgoal Interdependency Graph) [1] is developed to represent a subset of the business and the system non-functional requirements (NFRs). Figure 2-1 shows these NFRs as business (soft-)goals – goals for which there is no absolute criteria for their complete satisfaction - along with their sub-goals, and Figure 2-2 shows the system goals of the VMS, which are traceable to each other and the business (soft-) goals.

As shown in Figure 2-1, the final goal of the CIO of the organization is to maximize profits by lowering cost with good performance of service. To achieve the business goal, the CIO may consider migrating to a cloud, while exploring several options. The options being considered include a Consolidated Cloud, in which the whole VMS system is operated in the cloud. A Hybrid Cloud is another option, in which the VMS Data center manages important and highly confidential information and the cloud manages less important and less vulnerable information.

Softgoals Each option has its own pros and cons, and the CIO can carry out an estimated tradeoff analysis in terms of potential benefits and risks that each option is likely to bring about. For example, a Consolidated Cloud is estimated to be better for Lower cost[Service] (green ++), but worse for Security[Service] (red -), than a Hybrid Cloud (green S+, i.e., some +). The CIO can decide to choose one of the options that best meets (i.e., satisfies) the particular business goals as a business strategy, after confirming the tradeoff relationships between business goals and strategies by executing the Silverlining Simulator. In this case, we may assume, for the purpose of illustration, that the most important business goal is “Lower Cost”, hence consequently the CIO selecting a Consolidated Cloud. By the way, due to the space limitation, we show the simulation results for only the VMS Consolidated Cloud, as indicated by solid lines; other ones whose simulation results are not shown in this paper are indicated by dotted lines.

Softgoals, such as Good Performance and Lower Cost[VMS Consolidated Cloud] in the upper line in Figure 2-2, come from the business strategy on the right-hand side portion on the bottom of Figure 2-1. In consideration of the business strategy, a system analyst can then traceably establish the system’s concrete goals, such as Throughput and Cost, which respectively are quantified as 300 transactions per minute and $1.5 per hour, as shown in Figure 2-2. The alignment between business (soft-)goals and system (soft-)goals is represented by "eql" (equal relationship), as in “Good Performance[VMS CC] eql Good Performance[Throughput=300]” and “Lower Cost[VMS CC] eql Lower Cost[CostPerHour=$1.5/h]”. Note that, in the piecemeal illustrations, Lower Cost[VMS CC] is considered more important than performance, when selection decisions are made.

To achieve the system goals, a system analyst finds out which configuration elements of a system infrastructure are needed, such as “WAN Media” and “Server Capacity”, by estimating the application workload of Step2. There may be several operational options to achieve the system goals for each system element, and a system analyst can also choose one option, using our simulator results. As in Figure 2-2, the...
"WAN media" has options such as Fiber, Cable, and Google USA-wide [2] which have their own round trip time.

Using the simulator and the WANTimeResponse formula, as in Figure 2-3, a system analyst can obtain results as shown in Table 1. In this case, achieving all three subgoals of Good Performance [Throughput=300] will imply achieving Good Performance [Throughput=300] as well. Moreover, Good Performance [CPU Processing Waiting Time] and Good Performance [CPU Processing Time] are estimated to be achievable, via the use of a Cloud with any of the three different capacities (D1, D16 and D32) being considered (Of course, this can be confirmed, or denied, by simulation results). This would, then, imply that Good Performance [Throughput=300] is estimated to be achievable. In other words, if we estimate that D1 achieves Good Performance [Throughput=300] is estimated to be achievable via the use of a Cloud with any of the three different capacities (D1, D16 and D32) being considered (Of course, this can be confirmed, or denied, by simulation results).

From the perspective of the goal [CostPerHour=$1.5/h], the costs of Fiber, Cable, Google USA-wide, which respectively are 1.9, 1.75, and 1.0, have contributions - , S-, + respectively. Hence, Google USA-wide achieves the system goal of Good Performance, and at the same with the lowest cost among the options considered.

As in the above descriptions, A SIG serves as a notational convention for exploring options for achieving business and system NFR goals, such as cost and performance. At the end, it also serves as a record of the rationale for the various design decisions made, in consideration of tradeoff analysis (pluses and minuses) [3]. In Figure 2-1, for example, the pros (green lines) and cons (the red lines) from different options represent the contributions the different options make towards the NFR (soft-) goals. A Claim, “Lower Cost is the most important”, is the rationale for the particular decision made on the VMS Consolidated Cloud. Similarly in Figure 2-2, the selection on Google USA-wide has the rationale with the claim “Achieve the Throughput Goal with the Lowest Cost.”
Step 4: Generate simulation experiment XML

The Google App Script can generate XML to describe a simulation/forecaster experiment. A selected sample of the XML, in Figure 5, is highlighted in red to demonstrate XML’s ability to communicate a complete simulation experiment description to multiple discrete event simulation modeling frameworks supporting multiple cloud provider infrastructures. The important additional annotations are explained with the following XML tag descriptions:

- `<apptitle>`: Display-Status
- `<workloadmip>`: 10
- `<requestmsgbytes>`: 1000
- `<responsemsgbytes>`: 1000000
- `<instancetype>`: 10
- `<instanceclass>`: F1
- `<costperinsthour>`: 0.08
- `<sqlinstanceclass>`: D1
- `<sqlcostperinsthour>`: 0.10
- `<sqlcostpermio>`: 0.10
- `<sqlcoststoragegbyteperm>`: 0.24

The XML is designed to describe a complete simulation for the following open source discrete event simulation frameworks: SimPy [9], CloudSim [10] and OMNeT++ [11].

Step 5: Run Simulation experiment

Simulation models are created for understanding the behavior of a complex system without actually constructing the system. Simulation eliminates the time and expense that are needed to: design, code and test the software system, not to mention the physical hardware equipment either.

Figure 6 presents the results of using an open source discrete event simulation framework, SimPy, for a particular simulation experiment, which corresponds to the Google App Engine cloud characteristics for 300 concurrent users. The report is formatted to closely align with Google’s monthly invoice format. An explanation of the significant report data elements follows:

- **Section I.** Latency-goal (3 seconds), Throughput-goal (300 transactions per minute), Op-hours (operation hours per day 24) and #-users (number of concurrent users 300), Display-Status application SIM-THROUGHPUT-PER-MIN (throughput per minute 367.86), and Txn-workload-% (the percent of total transactions dedicated to the Display-Status application 10).

- **Section II.** The estimated daily frontend compute resource usage for 24 hours per day CHARGE is $83.35 daily or $2,500.39 per 30-day month.

- **Section III.** The estimated daily database (D1 CloudSQL) resource usage for one database instance, read-write operations and storage is $12.40 per 30-day month is $372.20. Here, D1 is the cheapest, hence with the lowest computing, infrastructure of Google App Engine, among 32 different infrastructures which enable the use of SQL.

The simulation run of 4-hours-simulation-clock-time, using a laptop computer (1.30GHz), took 2 minutes execution time at a very low cost. The simulation input variables can be modified to describe application workload changes and alternative cloud infrastructure configurations at a minimum cost.
as-a-Service (PaaS) on this cloud with an indication of the class of Infrastructure-as-a-Service (IaaS) to be used. We then ran several experiments on this cloud system to collect cost and performance data. This data seeds the simulator-framework SimPy, which is written in the Python programming language, and the simulator is then able to predict, to a high degree of accuracy, the cost and performance of operating a similar software system to the cloud.

We have described the essentials of our system, Silverlining, in this paper and interested readers are referred to some of the references at the end of this article for further details. Further research in this area relates to identifying adjustment factors when the cloud to be migrated to is not exactly like the GAE, to creating a revised local stand-alone GUI front-end for the simulator (the current Google App Script GUI requires internet connectivity to operate properly), and to porting the simulator to other domains, in order to cover a wide range of software applications and a variety of clouds. The simulator source code can be obtained by a requesting email to any of the authors.

Acknowledgments
This work wouldn’t have been possible without the generous Grant from Google and the untiring help of Michael Brauwerman at Google. Five Silverlining research team members also deserve special recognition — Dr. Sam Supakkul, Dr. Rutvij Mehta, Alan Anderson, Yishuai Li and John McCain (NTT Data) — for collaborating, encouraging and leading the way.

Step 6: Validate distance to cloud benchmark
After a review of the simulation report, one question remains — “Is the simulation throughput of 368 transactions-per-minute (tpm) a reasonable result?” The Silverlining Lab at the University of Texas at Dallas maintains the results of cloud TPC-C [7] benchmarks, ranging from 10 to 6000 concurrent users and throughput ranging from 12.5 tpmC to 7029 tpmC, for two kinds of platforms — Java and Python — and for a variety of CloudSQL and NoSQL infrastructures [12].

Figure 7 plots (in blue) a subset of the cloud TPC-C benchmark to show 320 users with 338.1 transactions-per-minute throughput, in the table on the bottom. The VMS Display-Status simulation shows 300 users with 368 transactions-per-minute (in red) throughput, with a similar application profile (similar workload, service times, database activity and cloud infrastructure configuration), in the graph on the top.

As seen in Figure 7, the comparison shows that the VMS simulation result is indeed very close (distance 320 to 368 throughput) the corresponding benchmark result. Hence, the answer to the above question, “Is the simulation throughput of 368 transactions-per-minute (tpm) a reasonable result?”, would be in the affirmative.

Conclusions: What the Silverlining Simulator Can Do
A common problem confronting just about every CIO is the cost and performance issues, when migration of software to the cloud is considered - a CIO needs to budget the cost and ensure that the performance of the system is not unduly compromised by the migration. This should be true whether the cloud is public, private, or hybrid, since the cost and performance in a cloud-based system is related to the configuration of the resources - different servers and platforms - supporting the intended software system. To aid the IT department and specifically the CIO, we have developed a simulator, Silverlining, that can forecast the cost and performance of a Google GAE and CloudSQL cloud infrastructure.

Our simulator is benchmarked on the GAE with an industry standard benchmark application, TPC-C, which covers a class of online transaction processing (OLTP) relational database-centric applications, and we deployed a configurable Platform-as-a-Service (PaaS) on this cloud with an indication of the class of Infrastructure-as-a-Service (IaaS) to be used. We then ran several experiments on this cloud system to collect cost and performance data. This data seeds the simulator-framework SimPy, which is written in the Python programming language, and the simulator is then able to predict, to a high degree of accuracy, the cost and performance of operating a similar software system to the cloud.

We have described the essentials of our system, Silverlining, in this paper and interested readers are referred to some of the references at the end of this article for further details. Further research in this area relates to identifying adjustment factors when the cloud to be migrated to is not exactly like the GAE, to creating a revised local stand-alone GUI front-end for the simulator (the current Google App Script GUI requires internet connectivity to operate properly), and to porting the simulator to other domains, in order to cover a wide range of software applications and a variety of clouds. The simulator source code can be obtained by a requesting email to any of the authors.

Acknowledgments
This work wouldn’t have been possible without the generous Grant from Google and the untiring help of Michael Brauwerman at Google. Five Silverlining research team members also deserve special recognition — Dr. Sam Supakkul, Dr. Rutvij Mehta, Alan Anderson, Yishuai Li and John McCain (NTT Data) — for collaborating, encouraging and leading the way.
REFERENCES

2. Grigorik, Google, Making the Web Faster at Google and Beyond Presentation, November 2012.

ABOUT THE AUTHORS

Prof. Lawrence Chung is in Computer Science at University of Texas at Dallas. Working in Requirements Engineering and Architecture, he was the principal author of “Non-Functional Requirements in Software Engineering”, and has been involved in developing “RE-Tools” (a multi-notational tool for RE) with Sam Supakkul, “HOPE” (a smartphone application for disabilities) with Rutvij Mehta, and “Silverlining” (a cloud forecaster) with Tom Hill. He received his Ph.D. in Computer Science in 1993 from University of Toronto.

The University of Texas at Dallas
800 W Campbell Road
Richardson, Texas 75080
Phone: 972-883-2178
E-mail: chung@utdallas.edu

Nary Subramanian is an Associate Professor of Computer Science at The University of Texas at Tyler, Tyler, Texas. Dr. Subramanian received his Ph.D. in Computer Science from The University of Texas at Dallas. He is a Faculty Fellow for Service Learning at UT Tyler’s Center for Teaching Excellence and Innovation. He has over fifteen years’ experience in industry in engineering, sales, and management. His research interests include software engineering, systems engineering, and security engineering.

The University of Texas at Tyler
3900 University Blvd.
Tyler, Texas 75799
Phone: 903-566-7309
E-mail: nsubramanian@uttyler.edu

Thomas Hill became an EDS/HP Fellow Emeritus, when he retired as the Director of EDS/HP Fellows and Distinguished Engineering after 40 years of service. He served as a systems engineer, since becoming an Air Force officer in 1963, building the first joint intelligence agency time-sharing system. He joined EDS in 1970, as a systems engineer. He has a BS and MBA in Information Technology and received his Ph.D. in software engineering from The University of Texas at Dallas.

The University of Texas at Dallas
800 W Campbell Road
Richardson, Texas 75080
Phone: 469-767-5011
E-mail: tom.hill.fellow@gmail.com

Grace (Eun-jung) Park is a Ph.D. Student of Computer Science at The University of Texas at Dallas. She has about fifteen years of industrial experience in software engineering, and she is an IT professional engineer in her native country. Her research interests are Requirements Engineering, System/Software Architecture, and Cloud Computing.

The University of Texas at Dallas
800 W Campbell Road
Richardson, Texas 75080
Phone: 469-450-3940
E-mail: exp130530@utdallas.edu