A Business-Oriented Approach to the Design of Feedback Loops for Performance Management †

Yixin Diao and Joseph L. Hellerstein and Sujay Parekh

IBM Research Division, IBM T.J. Watson Research Center, Hawthorne, NY 10532

This paper proposes an approach to automated enforcement of service level agreements (SLAs) by constructing information technology (IT) level feedback loops (e.g., admission control, CPU scheduling, load balancing) that achieve business objectives, especially maximizing SLA profits. We develop a framework in which profits are determined by revenues accrued for services delivered (e.g., completed transactions) and rebates to customers if services are unavailable or violate response time constraints. A methodology is described for profit-oriented controller design, and the methodology is applied to a Lotus Notes email server. The methodology relies heavily on modeling design choices to reduce the need for running live experiments. We show that simple linear models suffice to capture the dynamics of the email server we study, even for a time varying workload. In addition, our studies suggest that faster controllers (those with short settling times) consistently provide the best profits since they violate response time constraints less often and they do not sacrifice as much revenue. Last, our studies show that the selection of controller reference value has a significant impact on profits.

Keywords: profit model, control theory, controller design, business management

1 Introduction

Recent years have witnessed a dramatic growth in the number of electronically based service providers (SP) (e.g., Internet Service Providers, Application Service Providers, Management Service Providers, and Storage Service Providers). With this growth has come a widespread interest in service level agreements (SLAs) and automation of their enforcement. Many SLAs include specifications of: (1) revenue that is accrued to the SP for services delivered and (2) costs that are incurred by the SP in the form of rebates to customers if response time constraints are violated or the service is unavailable. This paper describes an approach to automating SLA enforcement in information technology (IT) feedback loops (e.g., admission control, CPU scheduling, load balancing) using business objectives, especially maximizing SLA profits (the difference between revenues and costs).

Much of the control structure used in IT consists of feedback loops. For example, Figure 1 displays a feedback loop (the "online flow") for controlling response time in a Lotus Notes email server. Mail users interact with the server to retrieve, browse, catalogue, and send their mail. Administrators specify policies in the form of desired response times (also called *reference values*). The difference between the reference value and measured response times is the *control error*. This is used by the controller to compute the setting of *MaxUsers*, which determines the number of users allowed to log in. Since this controls the load on the system and thus also the throughput and performance of the server, it has direct implications in terms of the SLA. The process of translating the business-level SLA goal (profits) into a manipulable IT-level design variable (i.e., desired response time) is the "Design Phase", and is the main focus of this paper.

†Please address all correspondence to Joseph L. Hellerstein (hellers@us.ibm.com, (914) 784-7506).

O. Festor and A. Pras (Eds.) : 12th International Worshop on Distributed Systems: Operations and Management DSOM'2001 Nancy France, October 15-17, 2001.

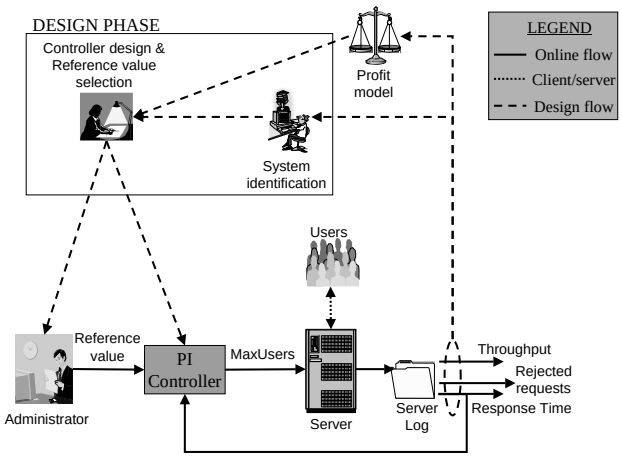

Fig. 1: Admission control feedback loop for Lotus Notes email server and the associated controller design process

Feedback loops have been widely studied in the field of control theory (e.g., [Oga97]). Typically, they are analyzed in terms of stability, steady state error, and transient response (e.g., rise time, overshoot, and settling time). Our interest is in a business oriented analysis, in particular, how to design controllers based on SLA profits (hereafter, just profits) of SPs. This has motivated our development of a framework that addresses the SLA costs and revenues associated with controller characteristics. Analysis based on this framework has yielded several results:

- A framework for constructing profit models by considering the revenues from completed transactions (throughput) and the costs from rejecting requests and from violating the response time constraint
- A methodology for designing controllers and specifying the reference value (see "design phase" in Figure 1) that maximize profits, which is based on a dynamic system model obtained through the system identification technique
- A simple first-order linear model may be sufficient to explain the basic dynamics needed to design and assess controllers (at least in the system we study), especially under heavy loads
- The choice of the reference value is affected by the profit model parameters, that is, a larger reference value is desired for the profit model more favorable to the revenues, but a smaller reference value is needed if the cost of violating SLAs is dominant
- Fast controllers (with a proper reference value) usually produce larger profits than slow controllers and oscillatory controllers (at least in the system we study)

Several areas of work relate to our efforts. Menasce et al. [MAFM00] describe an implementation that manages web server resources based on maximizing revenue (e.g., responding within 8 seconds so that users are not discouraged). However, they do not study the characteristics of control actions that are preferred nor is the choice of profit model considered. Liu et al. [LSW01] consider the optimization problem to maximize the SLA profits based on queueing-theoretic formulas, but the dynamic workload characteristics have not been addressed. Others (e.g., [PGH$^+$01], [MWT00], [LSA$^+$00]) have studied the performance characteristics of feedback controllers used for resource management. However, the relationship to profitability is unclear and so the desired controller characteristics cannot be determined. Much work has been done on pricing models for information systems such as pricing bandwidth and transactions (e.g., [FO98], [HL97]). However, these studies do not address the control characteristics that result in greater profits.

The remainder of this paper is organized as follows. Section 2 specifies the profit models we use to compare controllers. Section 3 studies the profits possible with an open-loop system. Section 4 details our approach to controller design. Section 5 applies our design methodology to Lotus Notes. Our conclusions are contained in Section 6.

2 Profit Models

This section describes the profit models we consider. By profit model, we mean the terms and conditions in a service contract that specify the revenues received by a service provider and the costs they incur. We begin with a brief discussion of service level agreements (SLAs). A variety of SLAs have been published. [SLAa] describes template for frame relay SLAs. [SLAc] provides guidelines for the state of Texas, including considerations for response time, availability, and downtime. [SLAb] does the same for the University of Michigan IT. Synthesizing these and other SLAs, our perspective is that of a transaction oriented service with multiple classes of transactions and customers. As a notational convenience, we assume that transactions are unique to each customer so that subscripts refer to the type of transaction. Our profit model only addresses costs and revenues specified in service contracts. Further, our categorization is relatively simple. For example, we do not consider time-of-day or zone based pricing, although these additional factors can be incorporated (with some added notational complexity). Rather, our focus is on core considerations for costs and revenues and their relationship to the choice of feedback controllers.

We consider two ways in which SPs accrue revenue. The first is on a per transaction basis as it is common in many e-commerce environment. That is, if $M_j(t)$ is the number of class j transactions that are processed by the SP during the t-th time period and the requestor is charged r_j for each class j transaction, then during the t-th time period the SP receives revenue of $R_T(t) = \sum_j r_j M_j(t)$. The subscript T is used to indicate that the revenues are obtained on a per transaction basis. Per transaction pricing has operational overheads in terms of logging requests received at the server and postprocessing these logs. Another approach is time-based charges that are independent of the volume of data transferred, such as those used for long distance telephone calls. Here, r_j is in units of money per time interval. So, $R_I(t) = \sum_j r_j$, where the subscript I is used to indicate interval-based revenues.

We address two kinds of contractual terms that impose costs on SPs. The first occurs if there is a degradation in service quality. Specifically, we assume that for the j-th transaction class there are contractual terms that specify a **response time constraint** W_j such that if the response time of a class j transaction exceeds W_j during a time interval, the SP incurs a penalty of c_j. A transaction oriented model results in costs $C_T(t) = \sum_j c_j N_j(t)$, where $N_j(t)$ is the number of class j transactions whose response times exceed W_j in the t-th interval. It may be that a transaction oriented cost model is impractical since considerable operational overhead may be involved with gathering per transaction SLA violations (e.g., response times must be collected for each transaction). An alternative is to use a probing station (e.g., such as what Keynote provides) to sample response times. This results in an interval-based approach. The cost model is $C_I(t) = \sum_j c_j \phi_j(t)$, where $\phi_j(t)$ is an indicator that is 1 only if the definition of "exceeds within an interval" (e.g., average value, maximum value, 95th percentile) is satisfied in the t-th interval. Note that $\phi_j(t) = 1$ implies that $N_j > 0$. There is a second kind of contractual cost as well–the penalties incurred if the service is unavailable. In the transaction oriented model, this results in a cost of $C'_T(t) = \sum_j c'_j N'_j(t)$, where c'_j is the cost for refusing to serve a transaction and $N'_j(t)$ is the number of refused transactions. The interval oriented model is $C'_I(t) = \sum_j c'_j \phi'_j(t)$, where $\phi'_j(t)$ is an indicator that specifies if the t-th interval meets the criteria for rejected transactions. As before, $\phi'_j(t) = 1$ implies that $N'_j > 0$.

Tab. 1: Profit model components

	Interval	**Transaction**
$R(t)$	$\sum_j r_j$	$\sum_j r_j M_j(t)$
$C(t)$	$\sum_j c_j \phi_j(t)$	$\sum_j c_j N_j(t)$
$C'(t)$	$\sum_j c'_j \phi'_j(t)$	$\sum_j c'_j N'_j(t)$

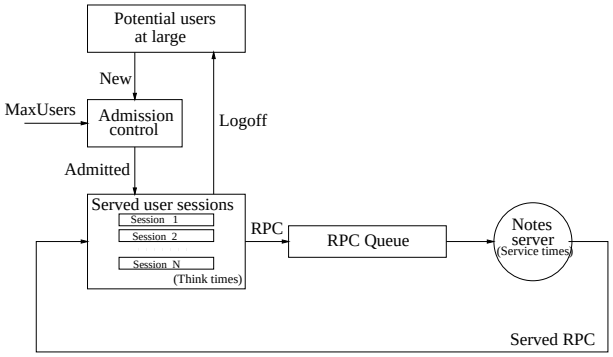

Fig. 2: Schematic of the Simulation Model of the Lotus Notes eMail Server

The various components of the profit models for both interval and transaction-based models are summarized in Table 1.

Having two different models (transaction-based and interval-based) for each of the three components of profit means that there are eight different profit models. A profit model can be specified by a string of the form $\{T,I\}^3$. For example, TII refers to the model for which revenue is transaction based, response time violations are interval based, and availability is interval based. That is $P_{TII} = \sum_t R_T(t) + C_I(t) + C_I'(t)$. While there are eight potential profit models, there are some constraints. Consider a model that uses interval-based revenue. Then, the customer should be very concerned if $c_j' = 0$ since the SP can optimize profits by refusing all transactions. This follows from the observation that: (1) SP revenues are unaffected by the number of completed transactions and (2) the SP incurs no cost if we never violate the response time constraint.

Several generalizations are possible using the foregoing framework. First, it may be that a transaction begins in class j and then, based on the resources consumed, becomes a class k transaction. (This is common in multi-level feedback control systems.) Thus, a more complex pricing policy can be used. Another generalization is to consider different pricing models for each transaction class. That is, class j might be TII and class k might be IIT.

3 Open-Loop Profits

To motivate the need for feedback control, this section studies the profits of an open-loop (no feedback) approach to specifying *MaxUsers*. By open-loop, we mean that *MaxUsers* is a constant.

In [PGH+01], controller design is performed by running experiments against a product level Notes server, an approach that provides accuracy but is quite time consuming. Fortunately, we were able to construct a fairly accurate queueing simulation of the Notes server. We use this as our live system (even though it is simulated). The structure of the simulated system is displayed in Figure 2. Potential users must pass through admission control, which is gated by the *MaxUsers* parameter. Once admitted, users cycle through two states: (1) thinking and (2) waiting for a response to an RPC (remote procedure call) submitted to the Notes Server. Admission control rejects users if the number of users present in the system exceeds *MaxUsers*. Actually, the details here are a bit more involved since admission control is exercised only for *OpenDB* RPCs. However, subsequent RPCs will fail unless the *OpenDB* succeeds. Our simulation addresses these details. Once admitted to the Notes system, users remain for the duration of their session. We use data from the testbed described in [PGH+01] to obtain think times, service times, and session lengths for the simulations. The duration of our simulation is 6 hours, including a one hour warm-up period. We found that the average response times produced by the simulation are within 1% of those of the testbed with the production Notes server.

To analyze the profits of the open-loop system, we simplify matters somewhat. First, only a single class is considered, and so we drop the subscript j. Also, we use transaction-based revenues, and interval based costs for SLA violations. For simplicity, we set c' to 0 to ignore the cost of refusing to serve a transaction. This is valid in the sense that we can build this cost into the transaction-based revenue model as loss of

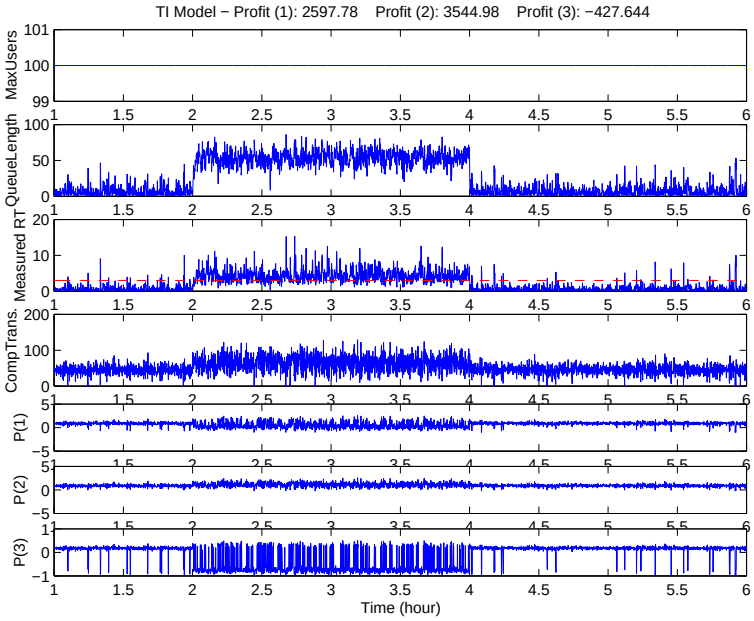

Fig. 3: Behavior of the Open Loop System

profit from rejecting the transactions. We refer to this as the TI model. That is,

$$P_{TI} = \sum_t (rM(t) - c\phi(t)). \tag{1}$$

We set the response time constraint, W, to 3 seconds, and we use the following three sets of parameters for the TI model: (1) $r = r_0$ and $c = c_0$ to indicate that both the revenue and cost are important. (2) $r = r_0$ and $c = c_0/5$ to indicate that the cost is relatively less important compared to Case 1. (3) $r = r_0/5$ and $c = c_0$ to indicate that the cost is relatively more important compared to Case 1. Here, we have $r_0 = 0.02$, and $c_0 = 1$.

Figure 3 displays simulation results for $MaxUsers = 100$ with a sampling period of 5 seconds. The total profits obtained during these 5 hours are given in the title of the figure for each set of model parameters. The dashed line in the third plot ("Measured RT") indicates the response time constraint W. Note that we vary user average user think times as follows: Period 1 (Hour 1-2): 10s, Period 2 (Hour 2-4): 3s, and Period 3 (Hour 4-6): 10s. As seen in Figure 3, load is light during periods 1 and 3 (e.g., the queue length is much less than *MaxUsers*) and heavy during period 2. Thus, it is not surprising that in period 2, transaction completions increase, but so do SLA violations. However, the increase in completions is modest because the system reaches saturation. That is, we have a large increase in cost (because of the SLA violations) but only a small increase in revenue. Thus, less profit is generated during this period.

We draw two insights from this example. First, a simple admission control policy can be used to maximize revenue if load is light: Accept all requests for service. However, the policy when load is heavy is considerably more complex since there are trade-offs between gaining revenue from increased transaction completions and avoiding the costs of violating response constraints by rejecting requests for service.

Second, a static setting of *MaxUsers* can be found using trial and error to maximize the profits. However, running a substantial number of experiments on the live system is time consuming. Further, the optimal value of *MaxUsers* will change if there is a change in think times and/or service times. These observations motivate our interest in a feedback controller that automatically adjusts tuning controls as workload changes so as to improve profits.

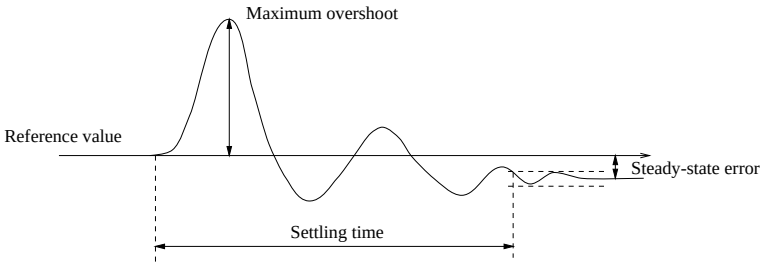

Fig. 4: Transient Response Characteristics

4 Profit-Based Controller Design

This section describes our approach to designing feedback controllers for performance management based on the profit models introduced in Section 2. We begin with a brief review of control theory, and then describe our design methodology. Throughout, we use the Lotus Notes email server as a running example.

4.1 Control Theory Background

Control theory provides a systematic approach to designing controllers. The classical theory assumes a linear, time invariant, deterministic system (although there may be multiple inputs and multiple outputs). The system being controlled has one or more actuators (tuning controls). In our case, the actuator is *MaxUsers*, the maximum number of users that may concurrently retrieve their email. The system outputs one or more metrics that are used by the controller to update the tuning control. In our case, these metrics are response times, throughputs, and rejected requests since these metrics are needed to compute profit. There may also be load disturbances in the system, such as time varying service times and think times. Indeed in Lotus Notes, we are concerned with how to adjust *MaxUsers* to compensate for such load disturbances.

We consider a simple but widely used class of controllers–**proportional, integral (PI) control**. PI controllers compute the value of the tuning control (*MaxUsers* in our case) as

$$K_p(e(t) + \frac{1}{T_i}\int_0^t e(x)dx), \tag{2}$$

where $e(t)$ is the control error (difference between the reference value and the measured value of the controlled metric), and K_p and T_i are the control gains. Intuitively, there is a large adjustment of the tuning control if the absolute value of $e(t)$ is large and/or the sum of past $e(t)$'s is large.

How are controllers evaluated? A primary concern is stability. That is, a bounded load disturbance should not result in an unbounded output (e.g., response time). Clearly, in information systems, unbounded output is impossible without unbounded load. More commonly, the problem is limit cycles in which the controller alternates between extreme settings of the controller without achieving the control objective. Such situations impair both adaptation and profitability.

Another commonly used metric to evaluate controllers is steady state error. In our context, steady-state error means that the response time achieved by the system differs from the reference (desired) value. In regulation applications (e.g., thermostats), steady state error is clearly undesirable. However, its impact is less clear in profit-based control in that we must quantify the revenue gained and lost as well as the cost incurred and avoided before a judgement can be made about the impact of steady state error.

A third criteria for evaluating controllers is transient response. By this, we mean how the output metrics fluctuate when the load disturbance changes (e.g., due to a change in think times and/or service times). Figure 4 illustrates the components of transient response of interest to us when there is a change in the workload. **Maximum overshoot** is the maximum deviation of the system output with respect to the reference value under the workload change. **Settling time** is the time required for the controller's response to be reduced to the point where the output metric varies by less than a prescribed amount (e.g., 3%).

1. Identify transfer functions (i.e., construct models) for response time, throughput, and rejected requests.
2. Select controller parameters to instantiate controllers with a wide range of transient responses.
3. Use the transfer functions from step (1) to compute the transient response of the controllers.
4. If the controllers are unstable or not sufficient diverse in their transient response, refine the parameters of the selected controllers and go to step 3.
5. Select a subset of controllers whose modeled profit are largest.
6. Compute profits for the selected controllers in the operational environment and choose the best settings for their reference values.
7. If the best profit obtained is acceptable, then stop. Otherwise, refine the parameters of the selected controllers and go to step 3.

Fig. 5: Methodology for Profit-Based Controller Design

4.2 Design Methodology

This section presents our methodology for designing controllers using the profit models described in Section 2. The methodology is intended to be general purpose, and is detailed in Figure 5. We discuss it in detail below using the Notes admission control example with the TI profit model, as illustrated in Figure 1. In the rest of this section, we address steps 1-5, the modeling and analysis of candidate controllers. The next section contains the detailed studies used in step 6. The **operational environment** is the live system with closed loop control. For us, the live system is the Lotus Notes simulation described in Section 3.

Step 1 is system identification. System identification uses data from the live system to develop a mathematical model that relates output metrics (denoted by $y(t)$) to tuning controls (denoted by $u(t)$). The output metrics are response time, throughput, and rejected requests (since these are inputs to the profit models). In Figure 1, $u(t)$ is *MaxUsers*. Consider the model

$$y(t) = -ay(t-1) + bu(t-1). \tag{3}$$

Here, the value of the output metric at time t is a linear function of its value at time $t-1$ and the value of the control at $t-1$ (there is a one time unit lag for the control to be in effect). We use batch least squares to estimate a and b. The data are obtained by varying $u(t)$ sufficiently to be persistently exciting (e.g., as in [AW94]); we use $u(t) = 250 - 150cos(t\pi/720)$. It turns out that Equation 3 accounts for 88% of the variability in response times. We attribute this very good fit to the fact that the system is under heavy load, and linear models (e.g., fluid models) have been shown to work well under such circumstances (e.g., [Mas97]).

Step 2 of Figure 5 selects the controller parameters to study. In our running example, we consider nine PI controllers specified by combinations of parameters chosen from $K_p \in \{0.2, 2, 20\}$ and $T_i \in \{2, 10, 200\}$.

Step 3 studies the controllers specified in step 2 using the models constructed in step 1. These studies examine the response of controllers to load disturbances. Common sources of such disturbances are changes in user think times and service times. Although Equation 3 does not explicitly address either of these, we can view load disturbances as causing changes in $u(t)$. The disturbance process we consider in the running example increases $u(t)$ by 20 at the 200th minute and decreases $u(t)$ by 20 at the 300th minute. (The first 200 minutes are used as a warm-up period.)

Figure 6 displays how modeled response times are affected. The x-axis is time, and the y-axis is response time. The dotted line is the reference value, which in our study is 1. The nine combinations of K_p, T_i are arranged in a matrix of plots with rows having the same T_i and columns having the same K_p. The controllers

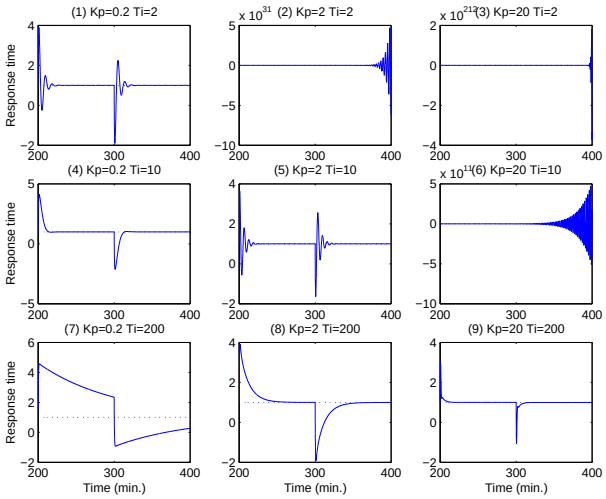

Fig. 6: Comparison of Transient Response Using First Order Models with Reference Value = 1

are numbered one through nine from left to right, and top to bottom. We have also studied the transient response of transaction completions to the same disturbance as used for response times. While the scale of the plots are different, their appearance are almost identical. Thus, in the interest of space, we do not include these plots.

In Step 4 we would like to ensure that the controllers span a sufficiently wide range of behaviors (e.g., fast, slow, and unstable). Otherwise, we repeat Step 3 with a different set of controller parameters. The plots in Figure 6 suggest that the nine controllers can be partitioned into three groups. Controllers 2, 3, and 6 are unstable, as can be seen by the large scale of the y-axis. Controllers 4, 7, and 8 have fairly long settling times (especially controller 7) compared with controllers 1, 5, and 9. We refer to the former as **slow controllers** and the latter as **fast controllers**. Observe that the maximum overshoot of the slow controllers is about the same as that of the fast controllers Only controller 7 has a significant steady-state error in the simulation time window.

How do the characteristics of the controller groups relate to T_i and K_p? In general, increasing K_p results in a faster response. However, a very large K_p creates oscillations and possibly instabilities (e.g., as in controllers 2, 3, and 6). T_i is used to eliminate steady state error, with a smaller T_i having a stronger effect. However, if T_i is too small, stability is affected. The combined effect of K_p and T_i explains the 3 classes of controllers.

Step 5 compares the profits of the controllers using the transient responses in step 4. For the running example, we use the TI profit model and the same profit model parameters as in Section 3. Figure 7 shows the profits corresponding to the nine controllers, where each plot corresponds to one set of the parameter values for c,r. No profits are included for controllers 2,3,6 since they are unstable. Observe that the faster controllers–1, 5, and 9–consistently have the largest profits. This is because longer settling times are usually associated with more violated intervals. Also, note that controller 7 fares much worse than the other slow

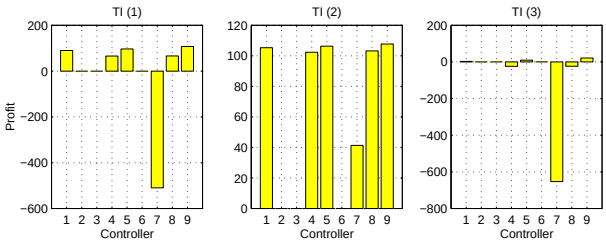

Fig. 7: Comparison of Modeled Profits of the Controllers With Reference Value = 1

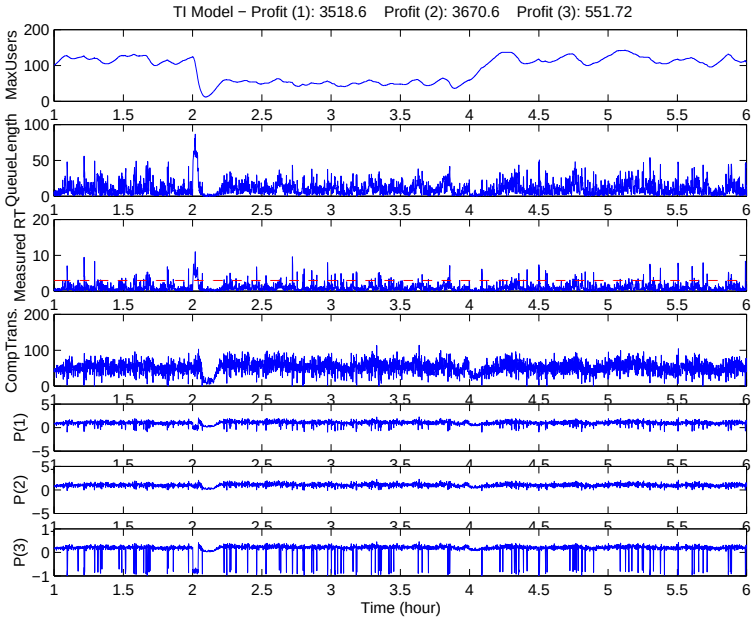

Fig. 8: Behavior of Controller 5

controller. Two characteristics of that controller contribute to this–it is the slowest of the slow controller.

We close this section with two observations. First, given that we are dealing with a stochastic system, the models we use are quite simple–first order deterministic linear models. Even so, these simple models are accurate in terms of the variability explained and thus provide a means to select a subset of the controllers to examine in more detail. We attribute this accuracy to our focus on heavy loads (where need for control is highest).

Second, we may be able to gain some insight into good settings for reference values. In particular, we know the standard deviation of the response time in the operational environment for the workloads we consider. Thus, if c/r is large, we want a low probability of violating the response time constraint while maximizing transaction completions. A good rule of thumb is to set the reference response time to be two standard deviations below the response time constraint (W).

5 Detailed Studies

This section conducts detailed studies of controllers using the simulated Notes system and the time varying workload described in Section 3.

We begin with a detailed examination of controller 5 ($K_p = 2$, $T_i = 10$) with a reference value of 1. Figure 8 plots *MaxUsers*, queue length, response times, transaction completions, and profits for the three values of r, c (see Section 3). Observe that *MaxUsers* is much lower between hours 2 and 4, the period during which think times drop from 10s to 3s. Also, comparing the total profits given in this figure with those in Figure 3, the open loop system, we see that controller 5 does considerably better than the open loop system for profit parameters 1 and 3, and there is little difference for the second set of parameters. The latter is a consequence of these parameters resulting in a relatively low cost for violating the response time constraint and so the benefit of the feedback loop is diminished. Indeed, if $c = 0$, it is preferred to have an open-loop system with a large *MaxUsers*.

Figure 9 shows the response times obtained by simulating the nine controllers (reference value is 1) for the same time varying workload. Note that the unstable controllers (2, 3, 6) have many large spikes in response times. The slow controllers (4, 7, 8) have fairly elevated response times during the period when think times are reduced (hours 2-4), although response times gradually decrease as these controllers adjust. The fast controllers (1, 5, 9) evidence only a small increase in response times at the second hour.

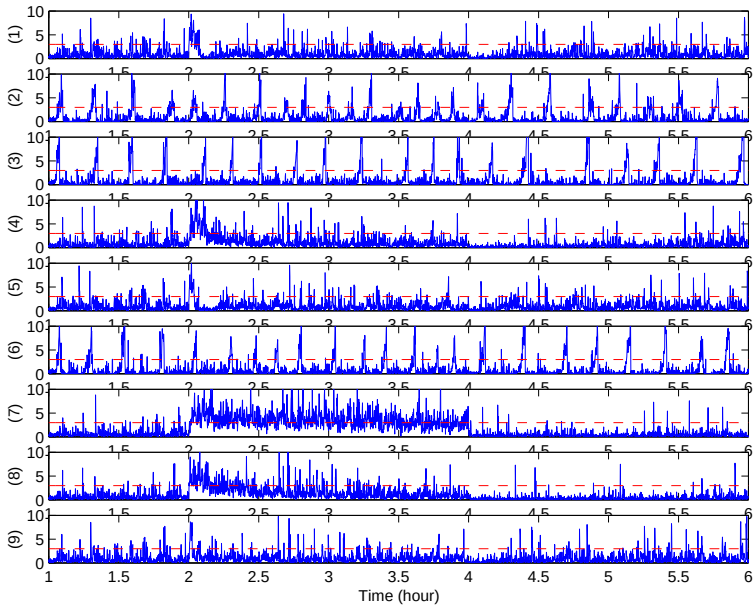

Fig. 9: Comparison of Controller Performance in the Operational Environment

The effect of these dynamics on profits is shown in Figure 10. There are two rows of three plots. The top row compares the nine controllers under the TI profit model. Note that the fast controllers are still the most profitable, followed by the slow controllers, and then the unstable controllers. The only exception to this is controller 7 when c/r is large, which is evidenced in the model results in Figure 6 as well. We attribute this to controller 7 having a very long settling time (although steady state error may play a role as well). The second row of plots in Figure 10 are for the TT profit model (i.e., both revenue and cost are transaction based). Here, we use $r_0 = 0.05$, and $c_0 = 0.05$. We see that the relative profit of the controllers remains unchanged. The foregoing suggests two insights, both of which require further study to validate. First, the simple linear models we use are sufficient to identify the relative performance of the controllers for the Lotus Notes system we study. Second, the relative performance of controllers is not strongly influenced by modest changes in the parameters of the profit models or even by the choice of profit model.

Can profits be improved with a different reference value? Figure 11 displays several plots of three representative controllers with reference value as the x-axis. The top row of plots have, respectively, completed transactions and violated intervals as their y-axis. We see that controllers 5 and 8 behave quite similarly in that completed transactions increase to the saturation point as the reference value is increased. However, the revenues of controller 5 (the fast controller) grow faster and its costs grow more slowly. Controller 2, an unstable controller, has far fewer transaction completions than either 5 or 8 and modestly more violated intervals. The impact on profits of these characteristics on profits is shown in the bottom row of plots. Controller 5 (which is fast) generally has more profits than controller 8 (slow). Controller 2's oscillating behaviors cause its profits to be the lowest since it has less revenue and its costs are not that much lower. Also, note that the preferred reference value depends on the parameters of the profit model and the controller. From these plots we conclude that the selection of controller reference value has a significant impact on controller profits. For example, fast controllers with modest oscillations should use a reference value that is modestly below the response time constraint since this reduces costs. Slower controllers do better if the reference value is closer to the response time constraint (W) since this increases revenue.

6 Conclusions

This paper proposes an approach to automated enforcement of service level agreements (SLAs) by constructing IT level feedback loops that achieve business objectives, especially maximizing SLA profits. To

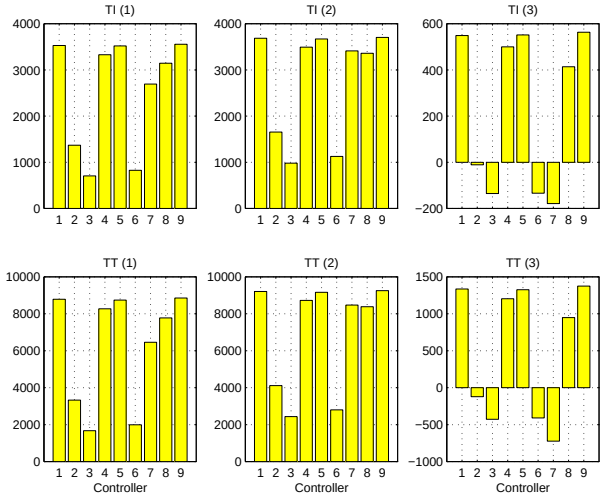

Fig. 10: Comparison of Profits in the Operational Environment

this end, we develop a model of the profits received by service providers as specified in service level agreements. Revenues accrue from services delivered (e.g., completed transactions), and costs take the form of rebates to customers if the service is unavailable or violates response time constraints. We propose a framework of eight profit models that consider whether a revenue or cost is obtained on a per transaction basis or by interval-based measurements.

Our methodology for controller design has steps for: model construction, selection of controller parameters, computation of modeled transient responses, evaluation of transient responses, controller pruning (subsetting) based on modeled profits, and detailed analysis of the selected controllers. The first several steps rely heavily on models of transient response. The last step also includes an analysis to select the controller reference value that maximizes profits (since this choice depends on the controller and the profit model parameters). One concern with this methodology is that the simple models constructed in the first step may not be sufficiently accurate for complex IT systems. Hence, controllers that are pruned based on modeled behavior may have done well in the live system. However, our studies of a Lotus Notes system with a time varying workload show that the ranking of controllers by profits that is obtained by the models is the same as that for the "live" system. Most likely this is due to our focus on heavy loads in that others have found linear models to be sufficient for modeling control dynamics. We focus on heavy loads since maximizing profits at light to moderate loads only requires maximizing transaction completions. In contrast, at heavy loads controllers must properly choose between increasing transaction completions to gain revenue versus decreasing violations of response time constraints to reduce costs.

Our study of Lotus Notes suggests that faster controllers (those with short settling times) consistently provide the best profits. This results from faster controllers having fewer cases of violating response time constraints and not sacrificing as much revenue since their undershoot of transaction completions is of short duration. Our studies also show that the selection of controller reference value significantly impacts profits. For example, fast controllers with modest oscillations should use a reference value that is modestly below the response time constraint since this reduces costs. Slower controllers do better if the reference value is closer to the response time constraint since this increases revenue.

The present study has identified several factors that potentially affect controller profits. Our next step is to study these systematically and in greater detail. This will provide the basis for follow-on work such as using non-linear models and developing optimal approaches to parameterizing controllers.

References

[AW94] Karl J. Astrom and Bjorn Wittenmark. *Adaptive Control*. Addison-Wesley Publishing Company, 2rd edition, 1994.

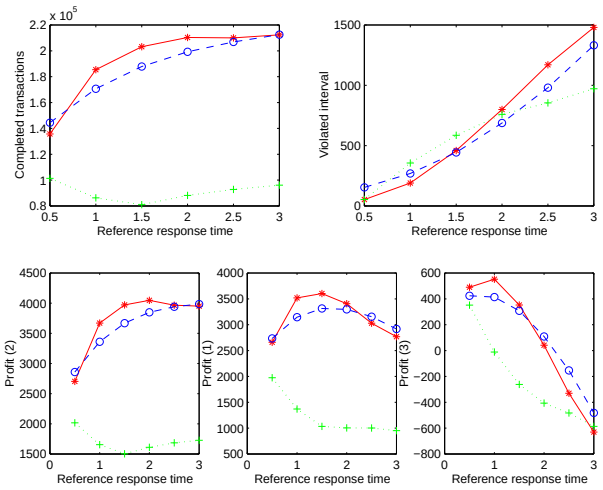

Fig. 11: Controller Comparison with respect to Different Reference Values (Fast Controller 5: *, Slow Controller 8: o, and Unstable Controller 2: +)

[FO98] P. Fishburn and A. Odlyzko. Dynamic behavoir of differential pricing and quality of service options for the internet. In *ACM First International Confrence on Information and Computation Economics*, pages 128–139, 1998.

[HL97] B. Huberman and R. Lukose. A methodology for managing electronic transactions over the internet. In *Third International Conference on Computing in Economics and Finance*, pages 1–14, June 1997.

[LSA+00] Chenyang Lu, Jack A. Stankovic, Tarek F. Abdelzaher, Gang Tao, S.H. Son, and M. Marley. Performance specifications and metrics for adaptive real time systems. In *Proceedings 21st IEEE Real Time Systems Symposium*, pages 13–24, November 2000.

[LSW01] Zhen Liu, Mark S. Squillante, and Joel L. Wolf. On maximizing service-level-agreement profits. In *Proceedings of the ACM Conference on Electronic Commerce (EC'01)*, 2001.

[MAFM00] D.A. Menasce, V.A.F. Almeida, R. Fonsece, and M.A. Mendes. Busines oriented resource management policies for e-commerce servers. *Performance Evaluation*, 42(2-3):223–239, October 2000.

[Mas97] S. Mascolo. Linear control theory for modelling, designing, and performance evaluation of ATM congestion control algorithm. In *Proceedings of 5th IFIP Workshop on Performance Modelling and Evaluation of ATM Networks Ilkley, UK*, pages 309–336, July 1997.

[MWT00] V. Misra, Bo Gong Wei, and D. Towsley. Fluid based analysis of a network of AQM routers supporting TCP flows with an application to RED. In *ACM SIGCOMM*, pages 151–160, October 2000.

[Oga97] Katsuhiko Ogata. *Modern Control Engineering*. Prentice Hall, 3rd edition, 1997.

[PGH+01] Sujay Parekh, Neha Gandhi, Joseph L. Hellerstein, Dawn M. Tilbury, and Joseph P. Bigus. Using control theory to achieve service level objectives in performance management. In *Proceedings of IEEE/IFIP Symposium on Integrated Network Management*, pages 841–854, 2001.

[SLAa] http://www.alliancedatacom.com/service-level-agreement-template.htm.

[SLAb] http://www.itcom.itd.umich.edu/mainten/lan_standard_sla.html.

[SLAc] http://www.tifb.state.tx.us/Handbooks/Service_Level_Agreement.htm.