Workload Management in Dynamic IT Service Delivery Organizations

Yixin Diao and Aliza Heching

IBM T.J. Watson Research Center
P.O. Box 704, Yorktown Heights, NY 10598, USA
E-mail: {diao|ahechi}@us.ibm.com

Abstract. Enterprises and service providers are increasingly looking to global service delivery as a means for containing costs while improving the quality of service delivery. However, it is often difficult to effectively manage the conflicting needs associated with dynamic customer workload, strict service level constraints, and efficient service personnel organization. In this paper we propose a dynamic approach for workload and personnel management, where organization of personnel is dynamically adjusted based upon differences between observed and target service level metrics. Our approach consists of constructing a dynamic service delivery organization and developing a feedback control mechanism for dynamic workload management. We demonstrate the effectiveness of the proposed approach in an IT incident management example designed based on a large service delivery environment handling more than ten thousand service requests over a period of six months.

1 Introduction

Information Technology (IT) Services Delivery involves managing IT systems and processes to support business functions and delivering high quality IT services to meet customer needs. As defined by IT Information Library (ITIL), IT Service Management consists of a cohesive set of best practices, such as the Service Desk for contact between service providers and service users, the Incident Management process to quickly restore normal service operations in the event of failure, and the Service Level Management process to ensure that Service Level Agreements (SLAs) are met with minimum impact on service quality [1]. An IT Service Delivery provider can manage its own IT environment or an outsourced environment supporting IT infrastructure for one or many customers. Outsourcing or multisourcing was originally driven only by cost considerations and attempts to focus on core competencies. There is increasing recognition that IT service providers can also ensure greater resiliency in case of disaster, business continuity due to time zone differences, and establishment of world wide best practices.

In this paper we consider the Incident Management process, a significant IT Service Management processes that requires a large number of service personnel. ITIL defines an incident as any event that is not part of standard operations and results in a service interruption or reduction in service quality. Without lose of

generality, we use the term "service request" to refer to the three categories of incidents, namely, application (e.g., disk usage threshold exceeded), hardware (e.g., system down), and service requests (e.g., forgotten password). Generally, service requests are associated with service level agreements that are designed at the time of contract negotiation and specify the target service response time. The objective of the IT service delivery provider is to most cost effectively meet the service level agreements. However, managing the service delivery organization and meeting this objective is challenging: customer workload contains inherent uncertainty, new customer accounts are onboarded frequently, established service requests may be routed to different service delivery environments for lower cost delivery, and the so-called steady state simply does not exist.

The majority of the existing literature in the area of service request workload management is confined to *change* requests, which can be scheduled in advance. [14] introduces a system for automated change management that accounts for precedence constraints. [18] solves the change scheduling problem by using a business-driven approach that evaluates change schedules in terms of the financial loss. [21] proposes a change scheduling optimization model that can be solved using standard mathematical programming techniques. More relevant is the work presented in [5] and [6] which develop a simulation model to guide the workforce decision making in IT incident management. With respect to *workforce management*, [13] considers the problem of long term workforce planning with general nonstationary arrival and service time processes. [20] studies dynamic staffing in a call center environment where the objective is high service level attainment. [7] proposes a method for determining the optimal numbers of permanent versus temporary staff and the threshold value at which temporary staff should be called upon, considering conflicting objectives of meeting service level constraints and minimizing costs. Although these papers include aspects of managing the dynamic service delivery environment, all of them depend on accurate models (which are difficult to obtain in real world systems).

A class of workload management methods based on *control theory* provides an alternative framework for studying the behavior of dynamic systems and feedback-driven control [12]. Various control-theoretic approaches to performance management of web server, differentiated caching, and multimedia streaming have been studied. [2] proposes an admission control method using classical feedback control theory and demonstrates this method for maintaining CPU utilization of an Apache web server. In [8] both CPU and memory utilizations are managed by a multiple-input, multiple-output controller that manipulates interrelated tuning parameters such as MaxClients and KeepAlive. In [17] an adaptive resource controller is developed to adjust resource shares and meet application-level quality of service goals in a data center environment. However, none of the above feedback approaches has been applied to workload management in service delivery organizations.

In this paper we propose a feedback control approach for workload management.The approach dynamically adjusts organization of service delivery personnel based upon the observed gaps between measured and target service level

metrics, in response to the changes in the business environment. The proposed approach does not require detailed system models. Rather, it employs feedback control to build closed loop systems that sense and correct the errors in modeling and control as well as changes in workloads and resources. In addition, compared to existing feedback control approaches used in systems management, the proposed approach differentiates itself as (i) devising a means to decompose the combined service level optimization problem into a set of single input single output control problems with specific service operation targets as reference inputs, and (ii) proposing an uncertainty-based adaptive control approach that updates the control parameters without detailed system models (which are typically difficult to obtain in a service delivery environment).

The remainder of this paper is organized as follows. Section 2 discusses the workload management problem in IT service delivery organizations. Section 3 presents the architecture and algorithm of a feedback control mechanism and uncertainty-based learning and adaption. Section 4 describes the controller evaluation results in an IT incident management example built using data from a large service delivery organization. Our conclusions are contained in Section 5.

2 Service Delivery Workload Management

In this section we describe the workload management problem in IT service delivery organizations. Recognizing that the real world scenario is full of details which typically render the problem highly complex and intractable; in the interest of simplicity and applicability, we refrain from modeling every aspect of incidents and service organizations. Instead, we focus on the key properties and first order effects. Figure 1 illustrates the process and operation flow of a service delivery environment. *Customers* interact with the service delivery environment through the *service request management system*, which coordinates service request creation, queueing, assignment, and closure. Based on the nature of the required services, service requests are assigned to different *service delivery units* which are comprised of a group of *service agents* (human resources) providing a set of services to the customers.

Typically, the arrival process of service requests are independent, although a batch of requests may arrive as a result of a major failure (e.g., database or network failure) that affects multiple components within an IT infrastructure. Generally, a service request only requires service from an individual service agent and can be resolved within several hours. As such, we do not consider change requests (that require scheduling to fit within predefined change windows) or project requests (that require significant coordination between the service agents and customer); both of these will have a service response time lasting from several days to several months. We also do not consider mis-routed service requests, since handling and improving such requests is out of the scope of workload and personnel management.

A service delivery environment is generally organized as a set of *service delivery units*, each servicing a subset of customers and consisting of service agents

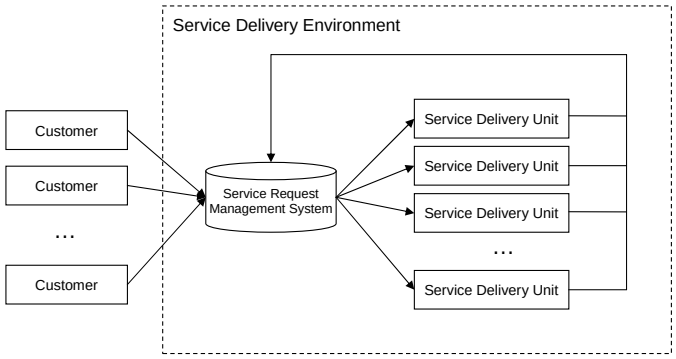


Fig. 1. Illustrative process and operation flow of a service delivery environment.

with similar skills. Assigning groups of customers to one or more service delivery units allows service agents to develop familiarity with customers' IT infrastructure, enabling improved quality of service. In addition, service agents are assigned to specific service delivery units based on skills, to achieve cost efficiencies; less complex requests can be routed to lower skilled units, while agents from the higher skilled units can focus on more complicated service requests.

One challenge faced by service delivery organizations is to determine the optimal size of a service delivery unit. An oversized service delivery unit results in low staff utilization, while an undersized service delivery unit may have difficulty achieving service level targets. Furthermore, the customer workload may vary over time. Although some variations may be predictable (for example, the banking industry may have fewer service requests at the end of the year when the financial activity slows down), other variations may be driven by random factors. The changes in customer behaviors directly impact the workload received by the service delivery units, and thus impacts their abilities to meet the service level agreements.

Next, we describe the architecture of the dynamic workload management system, where service agents are dynamically reassigned to service delivery units in response to changes in customer workload. The objective of the dynamic assignment is to balance service level attainment across service delivery units. Typically, there is a nonnegligible time penalty associated with human resource reassignment. In this paper, we assume this penalty is negligible by taking into account that the service delivery units are colocated and that reassignment only occurs between adjacent service units, where adjacency is measured by skills required to resolve the service request (customer familiarity and technical knowledge). For similar reasons, we do not consider a hiring/firing approach to workforce size adjustment, since this is costly and involves time consuming activities such as training and knowledge transer.

Figure 2 depicts the architecture. The feedback controller assigns service agents to the service delivery units, which handle customer service requests.

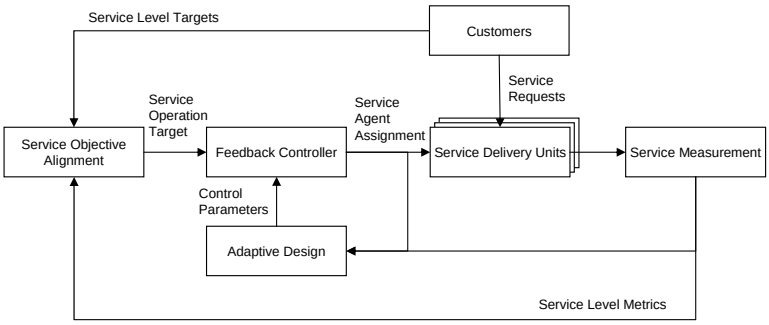


Fig. 2. Architecture of dynamic workload management.

Service requests are associated with service level targets that are customer specific. Service level metrics are periodically measured from the service delivery units, compared to the service level targets, and used by the feedback controller to adjust service agent assignment, if needed.

We now discuss in greater detail the three key modules of the dynamic workload management scheme: service objective alignment, feedback controller, and adaptive design. The goal of the *service objective alignment* module is to align the service level targets from the customers with the measured service level metrics from the service delivery environment, in order to create the service operation targets for the service delivery units. There are several distinctions here. First, the service operation target of the service delivery unit is created based on customers' business objectives regarding service level attainment (or average request resolution time, if specified by the customers), rather than internal operational metrics such as agent utilization or backlog length. Second, the service operation target is not derived based solely upon the service level targets at the service design time, but considers the actual measured service level metrics (which may vary as customer workload changes). Third, while the service requests from multiple customers are being serviced by multiple service delivery units, the purpose of aligning service level metrics with service level targets is to ensure fairness among customers. That is, we leverage the concept of load balancing. The management objective is either to meet the service level targets equally well (i.e., to keep the same safety margin between the targets and metrics in order to increase the service resiliency of unknown workload changes) or to equalize the partial derivatives of the service level metrics (which leads to a global optimal solution given convex cost functions [9]). This fairness objective can be augmented through weighting to consider differences, such as different service level infringement penalties, between customers.

The *feedback controller* operates to achieve the service operation target by dynamically adjusting the assignment of service agents to service delivery units. Indeed, to increase the scalability of dynamic workload management, we apply the separation principle to define the control objective centrally and operate the controller distributively. There is one controller for each service delivery unit,

which focuses on a single metric (the service operation target defined for that service delivery unit) and manipulates a single control (the required number of service agents for the unit). The collaboration across service delivery units occurs at a higher level: defining the service operation targets through service objective alignment to achieve fairness, and constraining the total number of service agents due to resource constraints.

The *adaptive design module* takes as input the operational data regarding service agent assignment and service level metrics, and outputs the control parameters for the feedback controller to provide appropriate control performance (i.e., quick response to workload changes while not overreacting to system noise).

We note that the above mechanism follows the monitoring, analysis, planning, and execution ("MAPE") model from autonomic computing. It supports decision making on dynamic workload and personnel management, and studies a simple feedback loop that captures the first order effect of adjusting service agents to improve service level attainment. The leverage that the feedback control mechanism provides is increased robustness of the solution in response to the stochasticity inherent in customer service request arrival patterns as well as service agent service times. In addition, as discussed in greater detail in the next section, the reactive nature of the system (i.e., the system feedback coupled with dynamic workload decision making) is designed to increase the likelihood that service level targets are met. Further, the simple yet effective structure of the control mechanism reduces the dependency on complicated system models. In contrast, optimization approaches either require deterministic input or, in cases of stochastic optimization, can result in complex models that are challenging to solve. Heuristic approaches, on the other hand, may result in solutions that oscillate and are impractical to implement.

3 Uncertainty-Based Adaptive Control

Following the *architecture* discussion presented in the previous section, in this section we describe the *implementation* of the dynamic workload management system in more detail: (i) Service objective alignment. We construct the service operation targets from service level objectives and metrics, and decompose the control strategy to enable single-input single-output controllers. (ii) Feedback control rule. We adopt a simple and yet effective control rule based on the Bang-Bang principle [19], and service agent assignment is adjusted based upon feedback from service level metrics. (iii) Adaptive design and operation. We present a set of methods that help to configure control parameters with minimal modeling effort.

3.1 Service Objective Alignment

Feedback controllers implement goal-driven algorithms to track a reference signal and adjust the control input with the objective of minimizing the error between the measured metric and the target reference. In this section we describe how

feedback controllers can be used to achieve fairness among a set of service level targets from multiple customers. The key method is to decompose the combined targets so that they can be fulfilled by multiple single input single output controllers.

Consider a service delivery environment that services M customers from N service delivery units. For each service delivery unit i, $i = 1, \ldots, N$, we define a service operation performance function

$$f_i(SL_i(k), SL_i^*) = \sum_{m=1}^{M} w_m \left(SL_{i,m}(k) - SL_{i,m}^* \right) \tag{1}$$

where $SL_{i,m}^*$ denotes the service level target for service requests serviced by service delivery unit i and originating from customer m, $SL_{i,m}(k)$ denotes the measured service level metrics at time interval k, and w_m indicates the weighting factor for customer m. The service level target $SL_{i,m}^*$ could be the service level attainment target, e.g., percentage of service requests serviced within a predefined time interval or average service request completion time.

The controller is designed to achieve "fairness" among all service delivery units, that is, eliminating the difference between performance functions

$$f_i(SL_i(k), SL_i^*) = f_j(SL_j(k), SL_j^*) \tag{2}$$

for all pairs of service delivery units i and j. Namely, the controller attempts to achieve balanced service levels attainment across all service delivery units. While Equation (1) defines such balance in terms of the service level attainment "margin," other types of performance functions can be used to characterize SLA violation penalties as well [3]. As described in the Performance Pyramid System developed by Lynch and Cross [16], metrics are used to measure an organization's performance on market and financial objectives. Together, these two types of metrics drive an organization to achieve the overall corporate vision. Metrics that are aligned with market objectives measure external effectiveness; metrics that are aligned with financial objectives measure internal efficiencies. In this paper we focus on a market performance metric – service level attainment. As it is a measure of customer satisfaction and lies in the group of external effectiveness measures, we consider balanced service level attainment. Other metrics that consider the financial objectives of the organization would consider factors such as differing penalties for service level violations.

From the feedback control perspective, this fairness objective can be achieved by constructing a set of single input single output controllers, one for each service delivery unit, with the common reference signal defined as

$$\frac{1}{N} \sum_{j=1}^{N} f_j(SL_j(k), SL_j^*) \tag{3}$$

and the control error for controller i as

$$e_i(k) = f_i(SL_i(k), SL_i^*) - \frac{1}{N} \sum_{j=1}^{N} f_j(SL_j(k), SL_j^*) \tag{4}$$

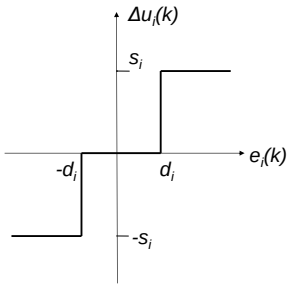


Fig. 3. An illustrative of Bang-Bang Control Law.

Thus, the objective of the feedback controller is to drive the control error $e_i(k)$ to zero using the following feedback control law

$$\Delta u_i(k) = u(k+1) - u(k) = K_i\left(e_i(k)\right) \tag{5}$$

where $u_i(k)$ denotes the control input (the number of service agents in service delivery unit i). Control law (5) looks similar to the integral control law that is used in proportional-integral-derivative (PID) controllers [12]. Although both forms aim to eliminate steady-state errors, the difference is that $K_i(\cdot)$ represents an integral function instead of an integral gain, as in the PID controller. This increases the design flexibility, as we will discuss in the following sections, so that we can design the controller directly based on system behaviors and uncertainties instead of building difference equation models through system identification.

3.2 Feedback Control Rule

Although a rich set of controllers has been studied in control literature, the majority are model-based and require extensive modeling and model-based adaptation [4, 11]. In the previous section we decomposed the multiple customer service level attainment problem into a single input single output control problem. In this section we leverage a simple yet effective control rule based on the Bang-Bang logic. A Bang-Bang (on-off) controller is frequently used in optimal control where the control input is restricted between an upper bound and a lower bound, and an optimal solution is to switch between the control bounds [10].

We apply the Bang-Bang controller due to the coarse granularity of service agents (i.e., they can only be moved in increments of full individuals) and the restricted size of the service delivery units. This makes an on-off controller more applicable, compared to other continuous or discrete control laws. We also enhance the on-off controller with the integral control rule, as in Equation (5), and a dead zone to ensure system stability. Figure 3 depicts the operation of the Bang-Bang control law for controller i. The x-axis denotes the control error $e_i(k)$ and the y-axis denotes the change in control input $\Delta u_i(k)$. A dead zone is defined so that no control input adjustment is given if the control error is

between $-d_i$ and d_i. If the control error exceeds the dead zone, the control input is adjusted by step size s_i. In the case of a positive error (that is, service delivery unit i performs worse than average and performs worse than all service delivery units) the Bang-Bang control law will increase the control input (i.e., add service agents) to reduce the deviation. When control input dependency exists (e.g., the number of service agents assigned to each delivery unit must be positive and total number of service agents assigned to all delivery units must equal the total number of service agents available) a projection algorithm can be applied.

Based on the control interval, the Bang-Bang controller operates periodically to drive the control error within the dead zone. Once in the dead zone, the controller monitors the control errors but no adjustments are made to the control input until changes in workload (e.g., increase in service request volume) drive the control error out of the dead zone. Overall, the performance and effectiveness of the Bang-Bang controller is determined by careful selection of the three control parameters: control interval (T), deadzone size (d_i), and step size (s_i).

3.3 Adaptive Design and Operation

While Bang-Bang control is widely used in optimal control with rigorous analysis [10], in this section we propose a simple and yet effective design approach that does not rely on detailed modeling and can react to system uncertainty arising from multiple sources. In a services delivery environment, sources of system uncertainty include: (i) System Randomness. Service requests are generated at random times and with different levels of complexity. Further, much of the measured data in a service delivery environment depends upon accurate recordings from service agents. Experience has shown that this data does not always reflect real events with perfect precision, due to the discrete (and unpredictable) nature of human beings. (ii) Transient Dynamics. There are often lags between the time when events occur (e.g., the service agent moves to a different service delivery unit) and the time when the metrics are measured (e.g., when the service level metric improvement is actually realized and reported after the agent settles down), introducing uncertainty as to the true state of the system during the transient phase. (iii) Workload Variation. Both system configurations (e.g., assignment of customer accounts to the service delivery units) and workload behaviors (e.g., service request arrival rate per customer, rate at which service agents handle service requests) can vary over time. Thus, robustness and adaptability are desired for a feedback controller that operates in a real environment. By using the Bang-Bang control logic and the three control parameters defined above, we design the adaptive feedback controller to accommodate these sources of system uncertainty.

Control Interval The control interval determines how frequently to adjust the control input, that is, the service agent assignment. Frequent adjustments may cause instability; infrequent adjustments implies that the controller may not be sufficiently reactive to workload changes.

We determine the appropriate control interval from the perspective of understanding and managing system randomness. Intuitively, if data variability is high, a larger control (sample) interval is required to ensure meaningful service measurement and control. According to the Central Limit Theorem, the distribution of the sample average of random variables approaches the normal distribution with a mean equal to that of the parent distribution and variance equal to that of the parent divided by the sample size (N), irrespective of the parent distribution.

For an initial control interval T_0, we measure the service level metrics $SL_i(k)$ and calculate the mean μ_{SL_i} and the standard deviation σ_{SL_i}. Given the desired noise ratio $r = \frac{\sigma_{SL_i}}{\mu_{SL_i}}$ from the control designer, we can compute the control interval (T) as follows.

$$\sigma^*_{SL_i} = r\mu_{SL_i} = \frac{\sigma_{SL_i}}{\sqrt{N_i}} \tag{6}$$

$$T = T_0 \max N_i \tag{7}$$

Based on experience, we set $r = 0.1$ to balance the feedback controller between sensitivity to system randomness and ability to adapt to workload changes.

Dead Zone Size The dead zone is used to increase controller robustness to system randomness and workload variation. Since the impact of system randomness cannot be entirely eliminated using control interval selection, the dead zone is introduced to avoid control oscillation, especially around the optimal steady state when the control error appears small. A dead zone is also valuable when the control input has a coarse granularity. For example, service agents can only be reassigned in increments of full individuals, even if the theoretical optimal value indicates a fractional adjustment. We design the dead zone size as follows

$$d_i = l\mu_{SL_i} \tag{8}$$

where l is the threshold limit that makes the dead zone size proportional to the average of the service level metric. Typically, we choose $l = 2r$, with the objective that no control action should be reacting to system randomness. Generally, a larger threshold limit can reduce oscillation but may also lead to larger steady state error.

Step Size The step size is related to the speed of controller convergence. A larger step size results in faster controller response regarding workload variation, but may cause the controller to oscillate around the optimal point with control error bouncing around the dead zone. Conversely, a smaller step size leads to longer convergence time. From our experience, we choose an initial step size s equal to 5% of the control range (the number of service agents in the service delivery unit).

If the step size is too large and causes oscillation around the dead zone, we introduce an oscillation-induced adaptation algorithm to resize the dead zone, as

Table 1. Workload parameters from an IT incident management example.

	Customer 1	Customer 2	Customer 3	Customer 4
Inter-arrival time (min)	14.2	45.4	177.1	20.1
Service Delivery Unit 1 (%)	92.5	83.8	64.4	82.1
Service Delivery Unit 2 (%)	7.5	16.2	35.6	17.9

follows: (i) Observe the control input history and record the sign of control input change. (ii) If an oscillation pattern is detected (e.g., the number of increases is equal to the number of decreases, e.g., 1, -1, 1, -1), increase the dead zone size by 20%. (iii) If a chasing pattern is detected (e.g., 1, 1, 1, 1, or -1, -1, -1, -1), decrease the dead zone size by 20%.

In addition to resizing the dead zone, this adaptation algorithm can be used to adjust the step size. The difference occurs when the granularity of the step size is too coarse. For example, adding or removing one service agent in a small service delivery unit may cause large control errors. In this case, we cannot further reduce the step size, but must increase the dead zone size to avoid oscillation.

4 Evaluation

In this section we illustrate how the proposed feedback controller can be used for workload management in IT service delivery organizations. Our evaluation is based on incident management data collected from a large service delivery environment over a period of six months, including more than ten thousand service requests. Each incident service request includes details about the incident arrival time, service time, and completion time. A separate data source provides the service level agreement information with target service response time and target service level attainment.

We analyze the service requests in order to characterize the workload. (For the purposes of this evaluation, some workload parameters have been modified to ensure sensitive business data are not revealed.) Our objective is to build a service testbed (simulator) based on a set of queueing models (specifically, M/M/m models [15]), that are calibrated using the collected service request data. We use the model to represent the dynamic behavior of the service delivery environment and examine how the proposed controller can be used for dynamic service agent assignment.

Table 1 lists the workload parameters for our evaluation, where the workload arrives from four customers. The inter-arrival times of the service requests follow exponential distributions, as approximated from the collected data. (We evaluated goodness-of-fit using the Kolmogorov-Smirnov test where the test statistic, the least upper bound for the cumulative distribution function, is 0.17 for the exponential distribution, compared to, for example, 0.28 for the normal distribution, and 0.70 for the lognormal distribution.) Table 1 also summarizes the

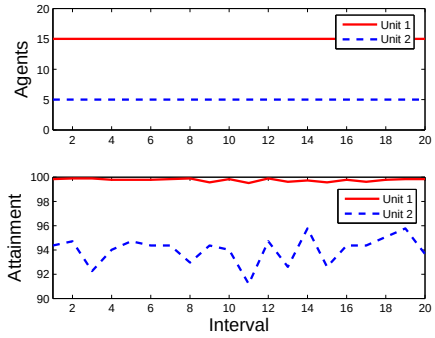
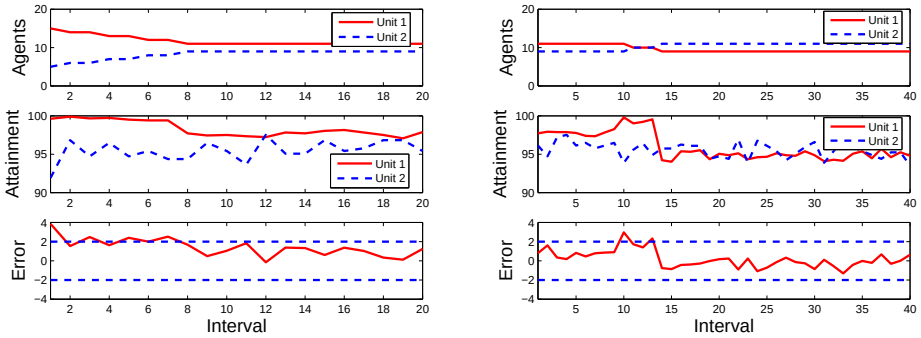


Fig. 4. Effect of static service agent allocation based on a heuristic rule.

workload distribution across the two service delivery units. In addition, the average service time per service request is 62.2 minutes in service delivery unit 1 and 123.9 minutes in service delivery unit 2. These service times follow exponential distributions as well. The target service response time is 8 hours for all service requests, and the service level attainment target is 95% for all customers in each of the service delivery units. (This is a simplification used in this queueing model; in a separate effort, we are building a discrete event simulation model for better representation). There is a total of 20 service agents in the two service delivery units. The control decision to be made in each control interval is the allocation of service agents to each of the service delivery units, with the objective of fairly meeting the service level attainment targets.

Typically, there is a mismatch between the desires of the service designers and the levers they are provided. In a service delivery environment, service designers wish to meet service level attainment target (with a safety margin) on all the service delivery units. However, this can only be achieved indirectly by adjusting service agent allocation (or service request routing). A heuristic method for assigning service agents to service delivery units is based upon the ratio of arrival rates and service rates. As indicated in Table 1, the arrival rate for delivery unit 1 is 0.128 requests/minute, which is 6.4 times the arrival rate for delivery unit 2 (0.02 requests/minute). Further, the average service time per service request in delivery unit 1 is approximately half that of delivery unit 2. Given these rates, one may expect the workload in delivery unit 1 to be approximately 3.2 that of Unit 2. If delivery unit 1 is allocated approximately 3.2 times the number of service agents than delivery unit 2 (i.e., 15 service agents to delivery unit 1 and 5 to unit 2), one may expect that the workload will be balanced and the service level attainment target can be equally met in both delivery units. Although simple, this algebraic rule does not capture the nonlinearity inherent in the queueing system. To illustrate, we applied this heuristic rule and the results are displayed in Figure 4. The x axis shows the control (sample) interval, and each interval is 10 days for service level metrics measurement. The upper graph plots the number



(a) Unknown workload (b) Workload change

Fig. 5. Control performance of the feedback controller.

of service agents assigned to the two service delivery units, and the lower graph plots their service level attainment over time. One can observe the gap between the service level attainment between the two service delivery units. Moreover, the attainment metric for service delivery unit 2 is hovering about the 95% target, causing SLA violations. Alternatively, the service designer can conduct experiments to determine the desired agent allocation. These experiments are time-consuming and must be repeated with changes in workload.

Figure 5(a) displays the performance of the proposed feedback control mechanism. The controller begins in the same static setting as that in Figure 4 and converges to a balanced service level attainment based on sensing the difference between service level attainment in the two service delivery units. The bottom figure displays the control error (the solid line) and the dead zone (the two dashed lines). As the control error enters the dead zone, no more service agent adjustment is performed.

Figure 5(b) displays the controller performance when the workload changes at interval 10. Specifically, the workload distribution between service delivery units 1 and 2 changes to 62.5% and 37.5%, respectively, for customer 1. This could occur, for example, if customer service requests follow seasonal patterns, or if the service delivery provider changes the assignment of customers (or types of customer service requests) to service delivery units. The controller begins with the optimal setting from Figure 5(a). When the workload changes, service attainment increases for service delivery unit 1 since fewer service requests arrive, prompting the controller to shift service agents to service delivery unit 2. After measuring the the service level metrics, the controller modifies the service agent allocation to nine service agents in service delivery unit 1 and eleven service agents in service delivery unit 2.

Overall, the proposed control mechanism successfully balances service level attainment across service delivery units, both when the workload pattern is stationary over time and when the workload dynamically changes. Further, ap-

propriate choice of step size combined with dead zone definition prevents the controller both from oscillating and from responding to system noise.

5 Conclusions and Future Work

Enterprises and service providers are increasingly looking to global service delivery as a means for containing costs while improving the quality of service delivery. In this paper we proposed a feedback control approach to dynamically adjust personnel based on dynamically changing workload. We also demonstrated the effectiveness of the proposed approach with an IT incident management example, designed using data from a large service delivery environment with more than ten thousand service requests over a period of six months.

While the initial results are encouraging, there are several areas for further research. First, the work presented in this paper focuses on dynamic service agent allocation. The underlying assumption is that there is no productivity impact (or "switching costs") associated with moving service agents between service delivery units. We assume that the service delivery units are colocated so that productivity impact would not come in the form of physical considerations such as required travel time or office space setup. However, other considerations include factors such as potential impacts on collaboration, team productivity, and communication between agents. Social network analysis may be useful to understand how service agents interact with other service agents within their service delivery unit and with service agents in other service delivery units as a measure of the impact of service agent switching. Second, while preserving the simplicity and applicability of the current approach, we can also extend the modeling considerations to include the service agent skills and the difference between service delivery units. Third, it is our intention to further validate our approach using the aforementioned discrete event simulation and study the impact of various workload mix and changes, and the effectiveness of control decisions (regarding step size, deadzone size, and control interval) in the presence of multiple (three or more) service delivery units. Finally, besides dynamic service agent allocation, we would like to study optimal service request routing as well as optimal mappings of customers to service delivery units.

References

1. IT Infrastructure Library. ITIL Service Support, version 3. Office of Government Commerce, 2007.
2. Tarek F. Abdelzaher and Chenyang Lu. Modeling and performance control of Internet servers. In *Proceedings of the 39th IEEE Conference on Decision and Control, Sydney, Australia*, pages 2234–2239, 2000.
3. Issam Aib and Raouf Boutaba. On leveraging policy-based management for maximizing business profit. *IEEE Transactions on Network and Service Management*, 2007.
4. Karl J. Astrom and Bjorn Wittenmark. *Adaptive Control*. Addison-Wesley Publishing Company, 2rd edition, 1994.

5. C. Bartolini, C. Stefanelli, and M. Tortonesi. SYMIAN: A simulation tool for the optimization of the IT incident management process. In *Proceedings of IFIP/IEEE International Workshop on Distributed Systems: Operations and Management*, 2008.
6. C. Bartolini, C. Stefanelli, and M. Tortonesi. Business-impact analysis and simulation of critical incidents in IT service management. In *Proceedings of IFIP/IEEE International Symposium on Integrated Network Management*, 2009.
7. Atul Bhandari, Alan Scheller-Wolf, and Mor Harchol-Balter. An exact and efficient algorithm for the constrained dynamic operator staffing problem for call centers. *Management Science*, 54:339–353, 2008.
8. Yixin Diao, Neha Gandhi, Joseph L. Hellerstein, Sujay Parekh, and Dawn M. Tilbury. Using MIMO feedback control to enforce policies for interrelated metrics with application to the Apache web server. In *Proceedings of Network Operations and Management*, 2002.
9. Yixin Diao, Joseph L. Hellerstein, Adam Storm, Maheswaran Surendra, Sam Lightstone, Sujay Parekh, and Christian Garcia-Arellano. Using MIMO linear control for load balancing in computing systems. In *Proceedings of the American Control Conference, Boston, MA*, 2004.
10. Wendell H. Fleming and Raymond W. Rishel. *Deterministic and Stochastic Optimal Control*. Springer Verlag, 1996.
11. G. F. Franklin, J. D. Powell, and A. Emani-Naeini. *Feedback Control of Dynamic Systems*. Addison-Wesley, Reading, Massachusetts, third edition, 1994.
12. J. L. Hellerstein, Y. Diao, S. Parekh, and D. M. Tilbury. *Feedback Control of Computing Systems*. John Wiley and Sons, 2004.
13. Otis B. Jennnings, Avishai Mandelbaum, William A. Massey, and Ward Whitt. Server staffing to meet time-varying demand. *Management Science*, 42:1383–1394, 1996.
14. A. Keller, J. Hellerstein, J. Wolf, K. Wu, and V. Krishnan. The CHAMPS system: Change management with planning and scheduling. In *Proceedings of IFIP/IEEE Network Operations and Management Symposium, Seoul, Korea*, pages 395–408, 2004.
15. Stephen S. Lavenberg, editor. *Computer performance modeling handbook*. Academic Press, INC, Orlando, FL, 1983.
16. R.L. Lynch and K.F. Cross. *Measure Up! How to Measure Corporate Performance*. Blackwell Pub., 1995.
17. Pradeep Padala, Kang G. Shin, Xiaoyun Zhu, Mustafa Uysal, Zhikui Wang, Sharad Singhal, Arif Merchant, and Kenneth Salem. Adaptive control of virtualized resources in utility computing environments. In *Proceedings of the 2007 EuroSys Conference, Lisbon, Portugal*, 2007.
18. R. Reboucas, J. Sauve, A. Moura, C. Bartolini, and D. Trastour. A decision support tool to optimize scheduling of IT changes. In *Proceedings of IFIP/IEEE International Symposium on Integrated Network Management, Munich, Germany*, 2007.
19. L. M. Sonneborn and F. S. Van Vleck. The bang-bang principle for linear control systems. *SIAM Journal on Control and Optimization*, pages 151–159, 1964.
20. Ward Whitt. Dynamic staffing in a telephone call center aiming to immediately answer all calls. *Operations Research Letters*, 24:205–212, 1999.
21. Leila Zia, Yixin Diao, Daniela Rosu, Chris Ward, and Kamal Bhattacharya. Optimizing change request scheduling in IT service management. In *Proceedings of IEEE International Conference on Services Computing*, 2008.