RC24553 (W0805-054) May 9, 2008
Other

IBM Research Report

A Framework for Model-Based Continuous Improvement of Global IT Service Delivery Operations

Abhijit Bose[1], Aliza Heching[2], Sambit Sahu[1]
IBM Research Division
Thomas J. Watson Research Center
[1]P.O. Box 704
[2]P.O. Box 218
Yorktown Heights, NY 10598

Research Division
Almaden - Austin - Beijing - Cambridge - Haifa - India - T. J. Watson - Tokyo - Zurich

A Framework for Model-Based Continuous Improvement of Global IT Service Delivery Operations

Abhijit Bose, Aliza Heching, Sambit Sahu
IBM T. J. Watson Research Center
19 Skyline Drive
Hawthorne, NY 10532, USA
{bosea, ahechi, sahu}@us.ibm.com

Abstract

In recent years, the ability to deliver IT infrastructure services from multiple geographically distributed locations has given rise to an entirely new IT services business model. In this model, called the "Global Delivery Model", clients outsource components of their IT infrastructure operations to multiple service providers, who in turn use a combination of onsite and offsite (including offshore) resources to manage the components on behalf of their clients. Since the components of services provided can be assembled and processed at any of the delivery centers, a framework for continuous monitoring of quality and productivity of the delivery processes is essential to pinpoint and remedy potential process inefficiencies. In this paper, we describe a framework implemented by a large global service provider that uses continuous monitoring and process behavior charts to detect any potential shifts in its global IT service delivery environment. Using this framework, the service provider has already improved several of its IT delivery processes resulting in improved quality and productivity. We discuss the major components of the framework, challenges in deploying such a system for global processes whose lifecycle spans multiple delivery centers, and present examples of process improvements that resulted from deploying the framework.

1. Introduction

In recent years, the ability to deliver IT infrastructure services from multiple locations around the globe has given rise to an entirely new IT services delivery model. In this model, clients outsource components of their IT infrastructure operations to potentially multiple service providers, who in turn use a combination of onsite and offsite (including offshore) resources to manage the components on behalf of their clients. This service delivery model, called the *Global Delivery Model* (GDM), is transforming the way businesses plan and execute IT services aligned with their business objectives. The service providers are increasingly considered as a strategic business partner to enable innovation, standardize processes and to achieve high operational efficiency; they are not simply viewed as a means by which to reduce the cost of maintaining the IT infrastructure. In GDM, an offered service such as server management and operations, or an entire business process such as loan processing, can be decomposed into a set of core service elements; the resulting workflows are dispatched to the designated *Service Delivery Centers* (SDCs) for processing. The SDCs, a combination of onsite-offsite centers, process and reassemble the completed workflows before delivering the requested service to the client.

In the early years of the GDM adoption, cost was clearly a deciding factor for establishing SDCs in low-cost locations. However, due to the increasingly complex requirements of IT service management and delivery, cost is presently only one of several factors in adopting the model. The benefits of adopting a global delivery model are many. Service providers can provide round-the-clock services because of time zone differences. Global delivery also ensures business continuity and resiliency of services in case of any disaster. The clients adopting GDM can scale their core business operations with demand without worrying about resources required to manage their IT infrastructure. Further, GDM forces clients to establish worldwide best practices and to optimize their processes and resources so that local skills and cost structure can be leveraged.

However, the GDM framework poses an enterprise with several unique challenges to achieve high levels of productivity and quality. Since the work to be performed must be decomposed and reassembled across SDCs, factors such as geographical distance, time zone differences, local regulatory environments, and social and cultural differences can all affect quality and efficiency. In recent years, the GDM service providers have faced a number of operational chal-

lenges with deployment of global processes, standard managing systems, and real-time visibility into delivery *key performance indicators* (KPIs). Quality improvement frameworks such as Lean and Six Sigma [2], Kaizen [3], Balanced Scorecards and Total Quality Management (TQM) have been applied to address quality problems at SDCs and to streamline GDM processes, with varying degrees of success. Most of these quality initiatives have been limited to Help Desk Support and Application Development and Maintenance (ADM) processes. This is due to the fact that quality improvement standards such as CMM (Capability Maturity Model) [5] and ISO 9000 have been available in the areas of software development and call center operations since the 1990's. As a result, there are well-established tools and management systems for improving these processes. On the other hand, there has been limited deployment of these frameworks for improving typical delivery processes performed at the SDCs, such as application hosting, network maintenance, system and server operations, database management and many other tasks that can not be performed by a help desk or call center. For large GDM service providers, these tasks often constitute more than 70% of their contracted services with the clients. The current GDM landscape can also be characterized by increasing pricing pressures, increases in delivery costs at many GDCs and decreasing barriers to entry driven by lower cost of managing systems. The complexities surrounding managing the GDM as well as the competitive business environment point to the need for tools and methods for monitoring and improving the quality and efficiency of the services that it provides. Therefore, development of GDM quality and process improvement techniques is an important area of service delivery research.

In this paper, we will describe an end-to-end framework for continuous improvement of global service delivery operations that can be implemented alongside of existing operations by making use of Kaizen principles [3]. This framework for continuous improvement is similar to the approach used in manufacturing environments. The methodology followed to implement the framework is as follows: As a first step, we analyze the different service activities delivered in a typical SDC or across multiple SDCs, by taking a system-oriented approach.

We then perform diagnostics to understand some of the sources of inefficiency in each of the service activities. In manufacturing, sources of inefficiency are better studied and understood; and may be attributable to e.g., job scheduling, machine downtime, or low yield. In the GDM environment, inefficiencies may be due to rework or delivering higher than agreed upon service level agreements or SLAs ("over-delivery"). Rework could be caused by service requests being incorrectly routed to the wrong service agent, multiple service tickets associated with a single problem, or lack of testing prior to moving to the production environment, for example. Inefficiency due to over-delivery occurs when the GDM targets a higher SLA level than that contractually specified, for example, resulting in greater workload than what is expected by the client and higher cost to the service provider.

The diagnostics component of the framework calculates a hierarchical set of metrics to measure the performance of the GDM. The metrics allow for identifying areas for improvement. Further, standardized metrics allow for cross-comparison across different client IT infrastructure as well as for benchmarking. Examples of metrics are number and types of service requests received on any day, standardized measures of requests service times and number of repeat incidents, etc. We measure the system according to these different metrics to gain a baseline measurement for system performance as well as to identify areas of improvement.

In the next step, these metrics are input to an analytical engine to produce a set of process behavior charts that automatically generates a series of signals whenever an anomaly is detected. The signals are then forwarded to the responsible SDC and GDC teams who analyze the data embedded in the signals, apply changes to the delivery environment as required and reset the signals. Sometimes, the signal indicate that the service delivery process has undergone a sustained shift. In this case, changes are applied to the signal-generating controls.

The steps of computing metric values, creating signals, and implementing changes based upon these signals are repeated on a daily basis. We discuss representative levers that are used to improve the delivery performance as reflected in subsequent measurements of the same metrics. The system compares baseline and post-improvement performances, and notes any changes in the process shift. This continuous monitoring framework combined with periodic diagnostics allows SDC and GDC teams to continuously improve operational efficiency and quality of the delivery environment.

The remainder of this paper is organized as follows. In Section 2, we describe the components and workflow in the Global Delivery Model. Section 3 describes steps of the continuous system monitoring framework to generate signals for continuous improvement in SDC/GDC operations; and in Section 4, we describe improvement levers that can be used to improve system performance. In Section 5, we describe the architecture of the prototype system we have developed to deliver the end-to-end framework for continuous improvement. Section 6 describes process behavior charts that are used to continually monitor system performance based upon standardized metrics. In Section 7, we show early results from a large service provider's global delivery operations, illustrating the benefits of implementing this end-to-end framework of constantly monitoring several

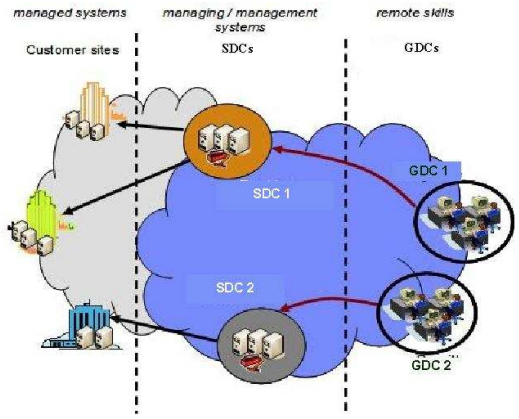

Figure 1. Components of the Global Delivery Model

SDCs and GDCs. We conclude in Section 8 with a summary of our current efforts in this area.

2. The Global Delivery Model

Figure 1 represents a typical global delivery model, comprising of three primary components: *managed systems*, *managing systems*, and *remote resources*. The managed systems represent servers, networks, applications, business processes and any other resources that a service provider manages on behalf of its clients either at its own data centers or in client premises. These are managed and serviced on a 24x7 basis by one or more *Service Delivery Centers* via managing systems. Each service component is designed to be delivered by executing a set of processes that perform specific tasks on the managed systems. The managing systems comprise systems, tools and workflow systems that are used in this context. The remote resources form a network of GDCs that receive workflows from SDCs and perform the designated tasks. The most distinguishing aspect of GDM is that these three components can be established in different geographies.

The work performed at a SDC/GDC is broadly categorized as follows: (i) *Incident and Problem Management* is work performed to restore normal service operations as quickly as possible. The root cause of incidents and outages is investigated and a new solution or work-around is created to prevent repeat events. The normal service operation is defined as service performed within the contracted Service Level Agreements (SLAs). (ii) *Change Management* is work performed to add, modify or remove changes to configurations of IT infrastructure. This work is typically scheduled in advance. (iii) *Request for Service (RFS)* is any other work presented by the client that is to be performed by the service provider. The work may be a new project under the umbrella of an existing contract or a new change request other than those approved under the change management work category.

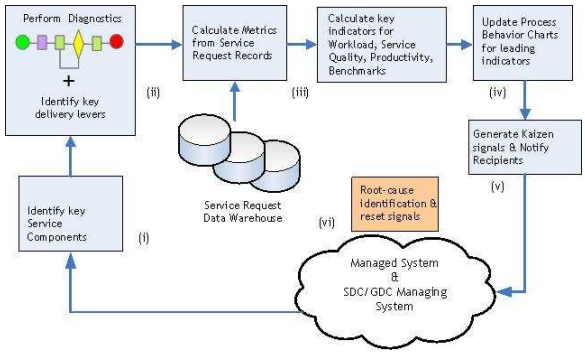

Figure 2. Continuous Monitoring and Improvement Lifecycle

These categories are part of the IT Infrastructure Library (ITIL) framework [4] for managing IT infrastructure, development and operations, and have been adopted by the majority of service providers. The complexity of managing these work categories requires a service provider to establish *hundreds* of processes depending on client requirements, local compliance and regulations, and IT infrastructure (systems, tools, applications). This inherent complexity and variability make continuous improvement at GDCs a difficult task that requires standard measurements, automated diagnostics and integration across the entire GDM value chain. In the next section, we describe how continuous improvement via Kaizen can transform a service provider's GDM by applying non-disruptive and consistent improvement levers to its operations.

3 Continuous Improvement Lifecycle

Figure 2 depicts the steps of this end-to-end process improvement framework, as follows: (i) analyze the service activities to identify the different service components, (ii) perform diagnostics to understand potential sources of inefficiency in the service components, (iii) calculate key service delivery system performance metrics, (iv) generate moving range and $\overline{X}$ charts [1] (i.e. process behavior charts), and automated signals to detect process shifts or out-of-control states, (v) automatically forward signals to respective delivery teams for root cause identification, system adjustment, and problem resolution, and (vi) reset signals after comparing pre- and post-improvement metrics. Steps (iii)-(v) are undertaken on a continuous (daily) basis, and the entire end-to-end process may be undertaken at periodic intervals to aggregate changes to the system, instead of reacting to individual improvement signals. Note that a number of industry solutions (Remedy, Peregrine, Maximo, etc.) are used to track workflows that arrive at a SDC or GDC. These requests are typically stored in data warehouses for historical analysis and reporting. We have implemented an automated system (described in Section 5) that calculates the relevant metrics from these service requests.

The diagnostics step identifies improvement levers that are most strongly correlated with the measured indicators. Once the levers have been identified, the delivery teams implement the improvements in their managing systems, to achieve the desired state — this step often requires making changes to managing system components (described below). In some cases, there may be a need for automation to achieve the desired state. In that case, the delivery environment continues in the "as-is" state until the automation steps have been completed and piloted in a small portion of the managing system. The effects of any changes to the managing system are captured via a continuous monitoring system that calculates the KPIs at regular intervals, so that they can be compared against the baselines captured before the changes were introduced. The novelty of our approach is that this is done in a continuous loop as part of the standard operating procedure at a SDC or GDC for IT service delivery, where as such practices have been limited to manufacturing operations so far. Next, we describe some of the improvement levers in more detail.

4 GDM Improvement Levers

The vast majority of the service requests arriving at a SDC/GDC originate from the managed systems requiring attention to a problem, incident or change. Therefore, by performing root cause analysis of common service outages, a significant portion of the incoming service requests can be eliminated. In addition to repeat service requests, there is often a significant portion of rework done at GDCs on services performed previously. For example, Table 1 shows total numbers and percentages of changes in various categories that were not successful during implementation at a GDC over a given time period. Since each change request requires approval and testing before it can be implemented, most of the work performed to generate the system patches can be characterized as waste or "non-value added work" using the Lean terminology, and therefore, must be repeated. The present framework captures indicators that track such unsuccessful changes as well as repeat requests with the necessary drill-down information to generate appropriate process improvement signals. We have

Change Category	# Failed	% Change Category
$Software - IntelPlatforms$	40	8
$Hardware - Server$	2	7
$OS - IntelPlatforms$	11	5

Table 1. Example of non-value added work performed

implemented several additional improvement levers in the present framework, e.g. (i) Segment incoming service requests by work category, complexity and priority so that *misrouting* can be minimized, (ii) Match capacity with demand by periodically rearranging size and skills distribution of GDC/SDC teams, and (iii) Batch multiple requests of similar type to reduce setup time in the managing system. Each improvement lever is linked to one or more indicators that are tracked via process behavior charts. The signals identifying the most effective (as determined by the system) improvement lever(s) are forwarded to the relevant delivery team/organization responsible for activating it.

5 Architecture

We describe the architecture of a prototype for Continuous Monitoring and Diagnostics system that we have built for monitoring and providing insights into GDM operations. Figure 3 illustrates the component view of the designed prototype for implementing the process described earlier in Figure 2. The main components of this system are: (i) Federated Data Warehouse: provides unified data schema and storage for computed metrics, KPIs and signals, (ii) Metrics Computation Engine: provides continuous computation of desired service delivery metrics and indicators, (iii) Service Transformation Analytics Engine: identifies process improvement levers and generates signals for forwarding to the delivery teams, (iv) Service Insight Portal: multi-role and portal-based access to the system

The operational life cycle of the prototype can be described by the following phases: (i) system configuration to capture GDM service hierarchy, (ii) periodic collection of incident, problem and change requests from the data warehouses, (iii) periodic computation of metrics and indicators, and (iv) generation of signals and insight through analytics. The system configuration refers to the phase where accounts and activities that are managed by a GDC/SDC are specified in the tool. The components are described in the following.

Federated Data Warehouse: The service activities managed by a SDC/GDC typically consist of several workstreams that may use different managing systems for serving work requests. We have built a federated data warehouse that collects service data from different managing system tools and creates a single repository of metrics and KPIs. Once configured, the Metrics Computation Engine can compute and store the computed metrics data using a unified data schema. There were several challenges in the design of the unified data schema such as different time zones, unit cost metrics, and granularity of available data. We consolidated these differences using our unified data schema that is used by the Federated Data Warehouse. In addition, we address resiliency issue in our design by supporting automatic failover to a standby server in the advent of primary database failure.

Metrics Computation Engine: This component handles computation of various metrics and leading indicators. The computation is done by issuing queries against managing system databases and only computed metrics are

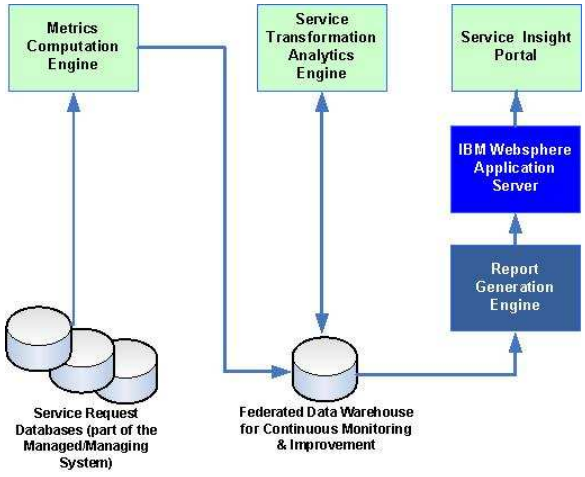

Figure 3. Continuous Monitoring and Improvement System - Architecture Diagram

transferred back to the data warehouse. Thus, there are several challenges the metric computation engine must address. It must ensure that when running the devised queries against the production databases, it does not affect the daily SDC/GDC operations. Further, it must handle several data inconsistencies before performing the actual metric computation. As there is no opportunity for data cleansing in the designed Warehouse, due to on-the-fly computing structure, the metric computation engine has to explicitly handle data inconsistencies such as different time zones, missing data fields, duplicate data etc. Once these metrics are computed, they are stored in the Federated data warehouse for the subsequent components.

Service Transformation Analytics Engine: Our framework is extensible such that new modules and analytics can be plugged in as long as it supports a set of common interfaces. We provide a key set of analytics such as (i) trending analytics based on clustering and segmentation, (ii)signal generation via process behavior charts, and (iii) predictive analysis. The process behavior charts being a key set of analytics for process improvement will be described in the next section.

Service Insight Portal: The Service Insight Portal provides a multi-role portal based interface for the users of our system and handles all the user interface related activities. This is implemented using Websphere Application Server and Web 2.0 technologies for the AJAX and client-side Java. The portal aggregates metrics, KPIs and signals based on the user role. For example, a service delivery manager can access these data for all the client accounts under her management. Users can quantify the effect of the process improvement levers by comparing the baselines (taken before improvement) with the most recent levels of the computed metrics and indicators. The supported roles can also be extended via an administrative interface in the portal.

5.1 Metric Standardization

A key challenge in measuring the GDM service delivery operations is the establishment of a standard set of metrics that represents GDM delivery status with confidence across all SDCs and GDCs. This is a complex task as different organizations and business lines use different measurement methods for monitoring their delivery operations that are often not consistent. We standardized the metric computation and evaluation process by designing a unified set of metrics based on the workload, service quality, performance and productivity criteria gathered from the different business units. The following is a representative sample of metrics computed in the present prototype of our system.

- Workload Metric: volumes of incidents, problems and change requests handled
- Service Quality Metric: percentage of service requests within the client SLA criteria, percentage of requests out of criteria, percentage of successful and unsuccessful changes performed on the managed systems
- Performance Metric: service response times at different levels of SDC/GDC hierarchy

6 Process Behavior Charts for GDM

Control charts, based on work by Shewhart in the 1920s and later popularized by Deming, have long been used in manufacturing systems as a means for measuring system variation and to ensure that variation in system output remains within acceptable limits. (See [6], [7], [1].) Control charts provide the capability to differentiate between natural variation, which cannot be removed, and special cause variation, which must be quickly identified and removed. Further, control charts allow for both retrospective as well as prospective monitoring of the process; they allow the user to review the existing state of the system as well as provide indications regarding future system performance.

We concentrate on two measurements: (i) the central tendency and (ii) the spread (variation) in the distribution of the data. Combined, these two measurements report on the accuracy and the precision with which the process is performed and, in the case of GDC/SDC, the service is delivered.

The major components of control charts are the points representing the individual measurements at different times, upper and lower control limits, and a measure of the central tendency of the measurements. (Other optional components are upper and lower warning limits.) For example, an individuals chart ("X-chart") plots the individual metric values at each point in time t. It also plots the $\overline{X}$, a measure of

central tendency over the individual metric values. Finally, it plots upper and lower natural process limits computed as:
Upper Natural Process Limit X-chart=$UNPL_X = \overline{X} + 3\theta_X$
Lower Natural Process Limit X-chart=$LNPLX = \overline{X} + 3\theta_X$

where θ_X is the standard deviation of the X values.

An mR-chart plots the absolute successive differences between the individual metric values at each point in time t (i.e., $|X_t - X_{t-1}|$, where $|\cdot|$ denotes absolute value). It also plots the $\overline{mR}$, a measure of central tendency over the range values. Finally, it plots upper and lower natural process limits computed as

Upper Natural Process Limit mR-chart=$UNPL_{mR} = \overline{mR} + 3\theta_{mR}$
Lower Natural Process Limit mR-chart=$LNPLmR = \overline{mR} + 3\theta_{mR}$

where θ_{mR} is the standard deviation of the mR values.

Shewhart chose to add (subtract) three standard deviations to (from) the central value to obtain the upper (lower) natural process limit because empirically, 99% of the data is located within three sigma units of the average. [1] Thus, these limits strike a good balance between the dangers of misclassifying common cause variation as special cause variation and vice-versa. Misclassifying common cause variation as special cause variation will lead to unnecessary cost and change to a process that is in control. Further, "false alarms" may cause users to lose faith in the capability of the process capability chart methods. On the other hand, misclassifying special cause variation as common cause variation will lead to result in missed process changes.

The method behind control charts is as follows: Assuming that the process is in control, construct control charts and determine upper and lower control limits. These limits are used to predict the system behavior. Collect new data observations and compare them to the average and upper and lower limits on the control chart. If the observations are consistent with the control chart, the process is assumed to be in control. If the new observations are inconsistent with the predictions based upon the control chart, the process is deemed out of control and action must be taken to identify the source of instability.

We have adopted the control chart methods from manufacturing and applied them to improve service delivery in the GDM. We refer to the control charts as process behavior charts, since we use these charts to monitor the behavior of the processes that drive the GDM to deliver the services that we offer, within the contractually agreed and customer expected specifications.

[1] We point out that these control charts are relatively insensitive to the distribution of the underlying data, see [6].

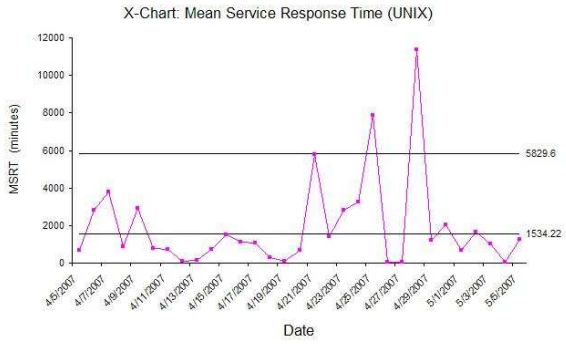

Figure 4. X-Chart; Unix work requests

We develop a hierarchy according to which we analyze performance of the GDM. All data is collected at the lowest level of the hierarchy, but metrics and process behavior charts are reported at varying views of the hierarchy. At the lowest level of the hierarchy are the different workgroups assigned to service the different tickets. The next level of the hierarchy is the specialized skills required to service the different ticket requests (e.g., UNIX, Database Management). The highest levels of the hierarchy are individual customer accounts and geography in which the delivery center providing the service resides.

Different users are interested in metrics reported at different levels of the hierarchy. For example, problems in the work queues are best addressed and studied by looking at the metrics associated with that level of the hierarchy. On the other hand, account managers will review metrics at the individual customer account level of the hierarchy.

For the metrics and indicators, at each level of the hierarchy, we provide an associated X-chart and mR-chart. We use 60 days of data to create the initial UNPL and LNPL and average lines for the mR-Chart and X-Chart. Once these limits are calculated, they are not recalculated until the process behavior charts indicate that the process has shifted and thus there is a need to calculate new process limits. On a daily basis, we update the remaining elements in the process behavior charts. By reviewing the resulting charts (two charts for each of the twenty metrics), we are able to continually monitor system performance and whether or not the system is behaving in a manner that is consistent with that predicted by the process behavior charts.

Figure 4 is an individuals chart for mean service response time ("MSRT"), at the workgroup level of the hierarchy. The limits were calculated based on two months of data, ending on 4/21/07. $UNPL_X = 5829.6$ minutes and $\overline{X}$=1534.22 minutes. [2] MSRT measures the total time (in minutes) from when a work request is entered into a system until the request is resolved (equivalent to the sojourn time

[2] We do not report $LNPL_X$ since MSRT can only take on positive values so the $LNPL_X = 0$ for this graph.

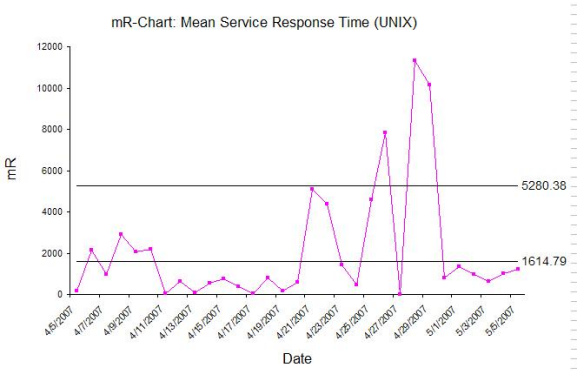

Figure 5. mR-Chart; Unix work requests

in a queue). [3]

Figure 5 is the mR chart for the same data set. The moving range chart measures the variability surrounding the service delivery. Within a given workgroup, highly variable time to resolve tickets could be indicative of, e.g., nonstandardized solution approaches or different worker skill levels.

6.1 Detecting Shifts or Out of Control

We implemented rules for detecting process shifts or out of control periods in the delivery process. By shifts in the delivery process we mean, for example, that it typically takes 100 minutes to close a ticket and the process shifts and it takes on average 80 minutes to close a ticket. Such a shift could be attributable to initiatives that improve the agents' abilities to service tickets such as introduction of new technology or recent agent training.

If the control chart indicates a process shift or out of control period, we send a signal (via email, visual indication on the graph, and tabular report) of the date and the signal detected. A designated individual must then investigate the cause for the shift or out of control period. A process shift is also a signal that process limits must be recalculated. Specifying detection rules requires a delicate balance between not sending a signal when the system has not changed but, on the other hand, not missing a signal when the delivery system has shifted or when the delivery system is out of control. We implement the following rules: (1) Any value that lies above $UNPL_X$ or below $LNPL_X$ indicates that the system is out of control and requires investigation. (2) Any value that lies above $UNPL_{mR}$ indicates that the system is out of control and requires investigation. (3) Eight consecutive values above or below $\overline{X}$ indicate a process shift.

Consider now the X-Chart in Figure 4. The dashed circle indicates eleven consecutive points that fall below $\overline{X}$. This indicates a shift in the delivery process; average time to resolve tickets has dropped significantly for an extended period of time. This process shift requires a recalculation of the process limits. In addition, the responsible individual must investigate to understand why the system performance was able to improve over this time period. For example, if the improvement is due to a short term assignment of additional agents then this improvement will not represent a sustained improvement and it does not represent a long-term shift in process behavior. The chart also indicates two points above $UNPL_X$ circled in a solid line. These points indicate unusually elevated MSRT and require investigation to understand what led to these high values.

[3] Although we present here aggregated data, we typically analyze this data at a finer level of granularity such as classified according to the different SLA agreements under which they are bound.

6.2 Other Considerations

Percentage Metrics In some cases we found that process behavior charts based strictly on the metric, was not the most appropriate measure. For example, one of the twenty metrics that we monitor is $Change_S$=Number of Successful Change Tickets. In this case, $Change_S = 0$ may mean that all change tickets were failures or that there were no change tickets. Thus, for monitoring process behavior, we normalize this metric by the total number of change tickets.

Rare Events In many cases we find that there are not observations for a given metric for every time period t. This happens most frequently for the tickets that are classified by the system as most severe. For example, the incidence of missing an SLA on a severe ticket is not common, especially when we study data at the most granular levels of the hierarchy. When the metric of interest is a pure "count" metric (e.g., number of tickets opened), standard methods can be applied to transform the count data to rate data (e.g., rate at which tickets are opened per year). However, not all of the metrics that we consider fall under the category of "count" metrics. For these metrics other methods must be used and other control charts must be considered, some of which are sensitive to assumptions regarding the distribution of the underlying data.

Non-Random Behavior The use of control charts allowed us to observe situations of non-random behavior. Such non-random behavior requires investigation as it is a certain indication of a need for process adjustment. For example, we noted that every Friday a certain account consistently missed its SLAs. The financial impact of this system behavior is obvious. Such behavior may take a long time to notice without graphical representation of the data through control charts.

Targets The natural process limits define the behavior of the process given its current configuration and its inherent variability. An additional limit that can be defined is the *Target*. The Target can represent, for example, contractually agreed upon service levels. The Target represents an external demand or request of the process, rather than an inherent capability of the process. Thus, for example, if service level

agreements are set at a level lower than $UNPL_X$, either the process must be modified or there is a good likelihood that the service level agreements will be violated. Process behavior charts have been useful in highlighting the concept of the capability of the processes and how this relates to the ability to deliver on different promised Target values.

7. Representative Results

This end-to-end framework has been implemented in stages, beginning in early 2007 and has already improved service delivery for a number of teams. A step-by-step deployment approach was taken as follows: the process improvement approach via appropriate lever adjustment was implemented iteratively for each work activity rather than in a single step for the entire GDM infrastructure. For example, in one deployment initiated in one of the service activities during the summer months of 2007, the primary lever utilized was improved routing of the problems to the appropriate service agents. We compare, for different incident and problem classes, the value of the MSRT metric before and after the improvements were implemented. Table 2 displays the results. Table 2 contains average service response

Priority Class	Date	MSRT
$Class1$	2/2007-3/2007	912.24 minutes
$Class1$	10/2007-11/2007	890.32 minutes
$Class2$	2/2007-3/2007	7654.99 minutes
$Class2$	10/2007-11/2007	7493.89 minutes

Table 2. MSRT (in minutes) by incident and problem class

time (in minutes) for different classes of incidents and problems before and after the process improvement lever was implemented. (Further details cannot be provided without revealing confidential service provider data but the scaled values in 2 still reflect true magnitude of improvement.) We analyze the different priority classes separately because these are typically governed by different service level agreements.

For Class 1 problems, average MSRT prior to applying the improvement levers as measured over a two month period is 912.24 minutes compared with 890.32 minutes, as measured over a two month period after the process improvement. This represents a relative improvement in MSRT of 2.4%. For Class 2 problems, average MSRT prior to improvement as measured over a two month period is 7654.99 minutes compared with 7493.89 minutes, as measured over a two month period after the process improvement. This represents a relative improvement in MSRT of 2.1%. While the average results seem low when aggregated at a SDC/GDC workstream, larger percentage improvements were observed for several individual teams. Also note the difference in magnitude between the response times for the different problem classes; the contractual agreements on Class 2 problems typically allow for longer response times.

8 Conclusions

Service delivery, in particular via the global delivery model, creates unique operational challenges that challenge the service provider's ability to deliver quality service. We have taken methods that have been applied in the manufacturing environment and used them to improve efficiency and quality of service delivery, in particular, in the IT service delivery setting. The end-to-end framework that was developed and implemented includes consistent metrics that are used to compare across all levels of the delivery environment hierarchy such as client accounts and service activities.

The upper and lower natural process limits in the process behavior charts can be used to help guide future client contractual agreements. Knowledge of the process behavior such as average service response times should help guide service provider commitments to clients regarding response time, and can provide insight into the likelihood of being able to meet client obligations if targets are not aligned with the natural process capability. For example, the process capability charts can highlight the likelihood of failure when the target line is set at a distance equidistant between the average and upper natural process limit. The statistical likelihood of missing this target is high, since points that lie in this range are within acceptable range of deviation from the mean and do not indicate any special cause variation. Thus, the only way to ensure compliance with this target is to use levers that shift the process so that the overall average service response time is reduced.

8.1. References

References

[1] W. E. Deming. *Elementary Principles of the Statistical Control of Quality*. Nippon Kagaku Gijutsu Renmei, Tokyo, 1952.

[2] M. George. *Lean Six Sigma for service*. McGraw-Hill New York, 2003.

[3] M. Imai. *Kaizen: The Key To Japan's Competitive Success*. McGraw-Hill/Irwin, 1986.

[4] F. Niessink and H. Vliet. Towards mature IT services. *Software Process: Improvement and Practice*, 4(2):55–71, 1998.

[5] M. Paulk, C. Weber, B. Curtis, and M. Chrissis. *The capability maturity model: guidelines for improving the software process*. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 1995.

[6] W. A. Shewhart. *Economic Control of Quality of Manufactured Product*. American Society for Quality Control, Milwaukee, Wisconsin, 1980.

[7] W. A. Shewhart. *Statistical Method from the Viewpoint of Quality Control*. Dover Publications, New York, NY, 1986.