SYMIAN: Analysis and Performance Improvement of the IT Incident Management Process

Claudio Bartolini, *Member, IEEE*, Cesare Stefanelli, *Member, IEEE*, and Mauro Tortonesi

***Abstract*—Incident Management is the process through which IT support organizations manage to restore normal service operation after a service disruption. The complexity of real-life enterprise-class IT support organizations makes it extremely hard to understand the impact of organizational, structural and behavioral components on the performance of the currently adopted incident management strategy and, consequently, which actions could improve it. This paper presents `SYMIAN`, a decision support tool for the performance improvement of the incident management function in IT support organizations. `SYMIAN` simulates the effect of corrective measures before their actual implementation, enabling time, effort, and cost saving. To this end, `SYMIAN` models the IT support organization as an open queuing network, thereby enabling the evaluation of both the system-wide dynamics as well as the behavior of the individual organization components and their interactions. Experimental results show the `SYMIAN` effectiveness in the performance analysis and tuning of the incident management process for real-life IT support organizations.**

***Index Terms*—Business-driven IT management (BDIM), decision support, Information Technology Infrastructure Library (ITIL), IT service management, incident management.**

I. Introduction

THE IT Infrastructure Library (ITIL [1]) is a comprehensive set of concepts and techniques for managing IT infrastructure, development, and operations. Developed by the UK Office of Government Commerce, ITIL is today the de facto best practice standard for IT service management. Among the processes that ITIL defines, *Incident Management* is the process for "...restoring normal service operation after a disruption, as quickly as possible and with minimum impact on the business".

This article tackles the problem of *optimizing the performance of an IT organization with particular regard to its help desk function and incident management process*.

IT support organizations consist of a network of support groups (each one with a team of operators). Support groups are organized into support levels, with lower level groups dealing with generic issues and higher level groups handling technical and time-consuming tasks. Real-life IT support organizations implement complex organizational, structural, and behavioral processes according to the strategic objectives defined at the business management level. In order to tune the performance of the IT support organization, it is necessary to evaluate the possible improvements brought by realignments of the current incident management strategy, or by the adoption of alternative strategies. This is a very challenging task, as it requires considering a large set of possible operations, such as re-staffing (the restructuring of the support organization by increasing or cutting staffing levels, or the transfer of operators around support groups, possibly on retraining), and the implementation of different policies for incident assignment and prioritization. In addition, the process of implementing the actual corrective measures is very expensive and time-consuming, so alternative strategies should be carefully considered before putting them in practice. This calls for *what-if scenario analysis*, a technique that enables the behavioral analysis of complex real-life systems under alternative working conditions. More specifically, what-if scenario analysis is based on the definition of an accurate model of the system under evaluation and on its exploitation to reenact of the system behavior with modified parameters.

This paper presents **`SYMIAN`** (SYMulation for Incident ANalysis), a decision support tool for the performance analysis and optimization of the incident management function in IT support organizations allowing what-if scenario analysis. `SYMIAN` enables its users to build an accurate model of real-life IT support organizations, to evaluate their performance, and to assess likely improvements brought by organizational, structural and behavioral changes.

`SYMIAN` models the IT support organization as an open queueing network [18]. This approach is particularly well suited for modeling the incident management process, as it builds on models of the dynamics of IT support organizations in terms of throughput, queue lengths, response times, and utilization, both at the system level and at the single support group level. The model is also able to make a distinction between the time spent by operators working on service restoration and the time spent waiting for operator availability, all the way down to the single incident level. The fine-grained model of the IT support organization implemented in `SYMIAN` allows users to play out what-if scenarios, such as adding technicians to a given support group and merging support groups together or splitting them apart.

`SYMIAN` exploits a discrete event simulator to reproduce in detail the behavior of IT support organizations and to evaluate their performance in managing incidents. The simulation approach is particularly appropriate given that the scale and the complexity of real-life organizations make it extremely difficult to devise an analytical model.

Manuscript received May 21, 2009; revised August 9, 2009 and May 7, 2010. The associate editor coordinating the review of this paper and approving it for publication was H.-G. Hegering.

C. Bartolini is with HP Labs, Palo Alto, CA USA (e-mail: claudio.bartolini@hp.com).

C. Stefanelli and M. Tortonesi are with the Dipartimento di Ingegneria, Università degli Studi di Ferrara, Ferrara, 44100 Italy (e-mail: {cesare.stefanelli, mauro.tortonesi}@unife.it).

Digital Object Identifier 10.1109/TNSM.2010.1009.I9P0321

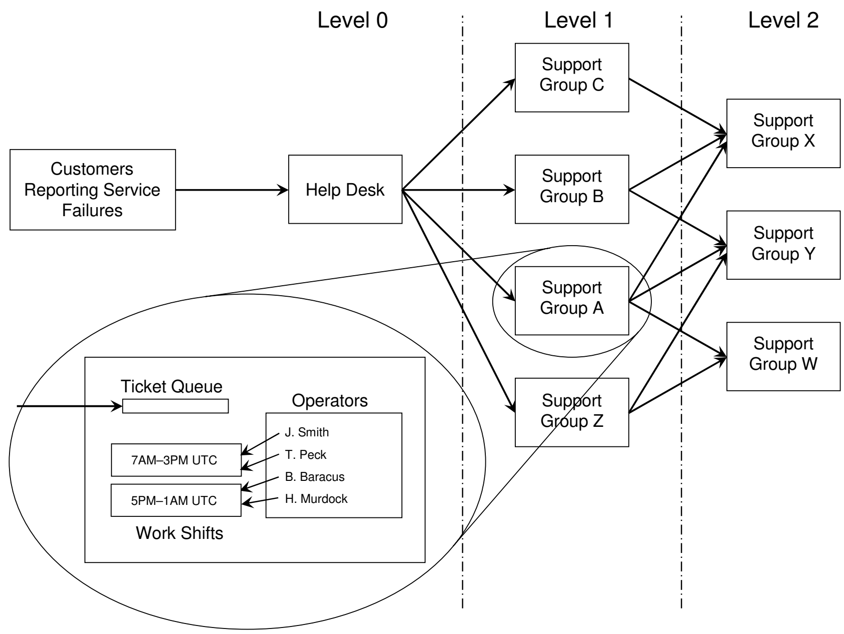


Fig. 1. Conceptual model of the IT support organization for incident management.

We applied SYMIAN to assessing and improving the performance of several real-life IT support organizations. The results demonstrated the effectiveness of the SYMIAN-based performance analysis and tuning process.

The paper is structured as follows. Section II describes the incident management process with the associated decision problem. Section III discusses the performance analysis of IT support organizations. Section IV introduces the SYMIAN tool and section V discusses its application for performance optimization. Section VI sketches both the SYMIAN architecture and implementation. Section VII presents the experimental results and section VIII reviews the related work. Finally, section IX provides conclusive remarks and future work considerations.

II. Analysis of the Incident Management Process in IT Support Organizations

ITIL divides the incident management process into several steps: incident detection and recording, classification and initial support, investigation and diagnosis, resolution and recovery, closure, and tracking.

IT support organizations dealing with incident management and are typically composed of a network of support groups, each comprising of a set of operators, with their work schedule (see Fig. 1). Support groups are divided into support levels (usually three to five), with lower level groups dealing with generic issues and higher level groups handling technical and time-consuming tasks. Support groups are further specialized by category of incidents that they deal with (network, server, etc. . .) and usually organized by geography, to ensure prompt incident response. In particular, the Help Desk represents the interface for customers reporting an IT service disruption. In response to a customer request, the Help Desk *opens* an incident, sometimes also called trouble-ticket or simply ticket. The incident is then *assigned* to a specific support group.

More specifically, when an incident arrives to a support group, it is put in an *incoming incident queue*. Incidents might also be divided in different service classes according to the support group incident prioritization policy.

As one or more operators become available to start servicing a new ticket from the incoming incident queue, the ticket is assigned to one of them according to the support group *operator selection policy*. For instance, only highly skilled and/or experienced operators can handle some particularly complicated incidents.

Incident prioritization and operator selection policies for the support group can result in the pre-emption of lower priority incidents waiting in the queue or currently being serviced.

In real-life IT support organizations it is not uncommon for operators to work on multiple tickets at the same time. In addition, if the operator's work shift ends before the incident service time is expired, incidents can either be handed over to another operator for around-the-clock servicing or simply wait until the operator's next work shift. Finally, the support group's operators either fully repair the incident or reassign it to a different support group (usually escalating to a higher support level). Fig. 2 provides a pictorial representation of the incident management process.

As a result, an incident goes through different states and is handled by different support groups throughout its lifetime. At each of these steps, the incident record is updated with pertinent information, such as its current state and related service restoration activity. If, for some reason, customers request the organization to stop working on the incident, the incident is placed in a *suspended* state to avoid incurring SLO (Service Level Objective) penalties. Once the disruption is repaired, the ticket is placed in *closed* state until the end-user confirms that the service has been fully restored. In this case,

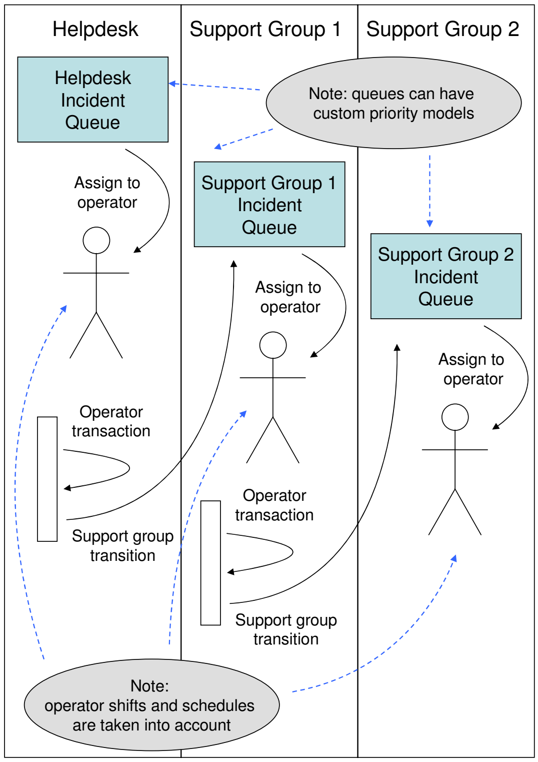


Fig. 2. The incident management process.

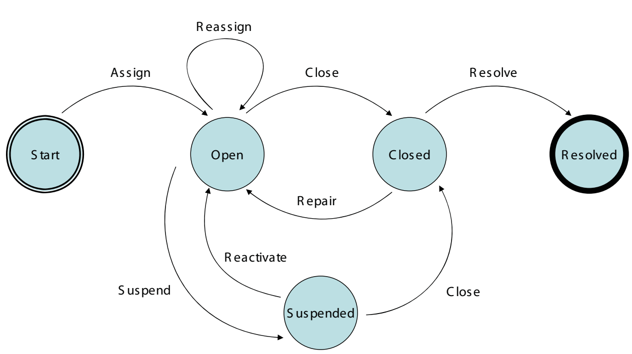


Fig. 3. Incident lifecycle.

the incident is *resolved* and its lifecycle ends (see Fig. 3).

The incident management process has objectives that are organization-specific and defined by the business, e.g., compliance with customer SLAs, minimization of economic cost in restoring service, or reducing the service disruption interval. When dealing with service disruptions, IT support organizations need to monitor incident management operations in order to verify the effectiveness of adopted incident management strategies and to evaluate possible alternative strategies when the business objectives are not met.

Frameworks such as ITIL [1] and COBIT [2] help by defining objectives for incident management, and usually linking them to simple high-level organization-wide performance metrics, such as mean time to incident resolution (MTTR). However - even though they provide an excellent starting point for driving the choices facing IT organizations - the performance indicators proposed by ITIL and COBIT present a number of limitations that make them unsuitable for capturing the complexity of large IT support organizations. On one hand, the definitions of the measures of impact, risk, etc., proposed by ITIL and COBIT are vague and not necessarily quantitative. On the other hand, more importantly, commonly used measures do not cope well with the extremely dynamic nature of the organization. For example the priority of a service incident is set at creation and only reviewed in extraordinary cases.

The performance analysis of incident management strategies adopted in IT support organizations is non-trivial and may involve a large set of complex and lower-level metrics that go beyond the simple performance indicators proposed by ITIL and COBIT. A comprehensive performance analysis of the incident management process should delve into a deeper level of detail, considering both the behavior of the single IT support organization components as well as the interactions between them and the global system behavior.

The above-mentioned observations on the common structure and behavior of IT support organizations suggest the adoption, for performance analysis purposes, of metrics that are capable of capturing system-wide aspects as well as inter- and intra-support group dynamics.

III. Performance Analysis and Bottleneck Location

The adoption of a fine-grained model for IT support organizations allows for the definition of performance metrics that can accurately capture the business impact of service disruptions. More specifically, performance metrics should consider two orthogonal dimensions: the *effectiveness* of incident routing and the *efficiency* of every single support group in dealing with the incidents. This requires taking into consideration other performance metrics that can evaluate the ability of the organization to forward incidents directly to the best equipped support groups and the optimality of staff allocation and operator work shift scheduling.

Among the predefined performance metrics meant to determine the effectiveness of routing in IT support organizations, we consider:

- number of reassignments per incident;
- number of assignment cycles;
- number of incidents seen twice or more at a given support group;
- number of cross-level reassignments;
- number of incident record updates (operator transactions) between (forward / back) reassignments;
- number of inconclusive updates (operator transactions) at a single support group before the incident is bounced back to the originating support group;
- time to closure after reassignments;
- number of incidents that had an unusually large service time at a given support group and were then escalated to another support group.

We consider sampling/averaging of these metrics on different time intervals: daily, monthly, and for the total simulation duration.

Most of these metrics intuitively represent aspects of the incident management process that can highlight ineffective routing of incidents. For instance, the number of reassignments per incident gives a clear indication of how well routing is functioning. The support groups that tend to treat more long-lived tickets than average can be pinpointed as a good starting point for troubleshooting performance issues in the IT organization.

Another example of an information-rich metric is the number of assignment cycles. The rationale behind this metric is that if an incident loops between a few support groups before resolution, then there probably is something wrong in those groups. The same consideration applies for tickets that are bounced back and forward by a pair of support groups.

The set of predefined IT performance metrics aimed at measuring the efficiency of support groups in dealing with incidents is:

- fan-in and fan-out of support groups;
- mean incident sojourn time in support group;
- number of incidents received vs. number of incidents resolved;
- number of incidents treated;
- number of operators that worked on the same ticket at each support group.

Again, most of these metrics are quite intuitive. The fan-in and fan-out metric is intended to represent the centrality of the support group in the organization. It measures the number of support groups that this group receives incidents from (fan-in) and forwards incidents to (fan-out). The rationale behind it is that IT support organizations are more sensitive to performance issues in central support group than to those in peripheral support group.

Support groups where tickets spend most of their time, support groups with a low incident resolution/escalation ratio, support groups processing a high number of incoming incidents, and finally support groups with many operators working on tickets are also potential bottlenecks.

Once the performance metrics are defined, the problem is to optimize the IT support organization performance with respect to these metrics. This might require a realignment of current organizational, structural, and behavioral processes. For instance, the operations available to IT managers for the optimization of IT support organizations include:

- increasing or cutting staffing levels;
- transferring operators around support groups;
- implementing different prioritization policies for incident queues;
- implementing different operator selection policies.

Unfortunately, while the analysis of specific metrics for performance evaluation of real IT support organization is almost straightforward, it is extremely difficult to evaluate the impact that changes in the organization may have on these metrics. As a result, the performance assessment of alternative behavioral, structural, and organizational processes calls for decision support tools enabling *what-if scenario analysis*.

IV. The SYMIAN Decision Support Tool

SYMIAN is a decision support tool designed to enable the performance analysis and optimization of the incident management process in IT support organizations. In particular, SYMIAN enables its users to play out *what-if scenarios*, allowing them to assess likely improvements in performance without having to go through the expensive and time-consuming process of implementing actual organizational, structural and behavioral changes.

More specifically, SYMIAN allows its users to build an accurate model of the IT support organization (in terms of the number of support groups, the support levels, the set of operators, the operator work shifts, the relationships between support groups, etc.). SYMIAN then exploits a discrete event simulator to reproduce the behavior of the IT organization considering a user-specified set of incoming incidents and metrics for the performance evaluation. Finally, SYMIAN processes the simulation outcome to spot performance bottlenecks, and proposes potential reconfigurations that are likely to bring performance improvements.

The what-if scenario analysis implemented by SYMIAN enables an iterative performance optimization process, in which users can incrementally specify the set of changes to apply to the current organization model in order to define an alternative configuration that will be tested on a set of performance metrics. SYMIAN guides users all along the optimization process, providing functions for configuration rollback and ad hoc visualization of simulation results.

A. The SYMIAN Model of IT Support Organizations

Modeling the incident management function of IT support organizations is an arduous task. In particular, the creation of a realistic model requires considering a high number of parameters in order to capture the complexity of typical IT support organizations. In addition, the effective adoption of an IT support organization model in the context of a decision support tool poses significant constraints on its computational complexity. SYMIAN's model is complex enough to accurately capture the dynamics of a real IT support organization, yet simple enough to allow for an efficient implementation and a user-friendly configuration interface.

SYMIAN models the IT support organization as an open queuing network [18]. More specifically, each support group g_i (with $i = 1 \dots N$) of the IT support organization is modeled as a G/G/s_i queue, a multi-server queue with generic arrival and service times, with s_i being the number of operators in the support group. Incidents are injected into the system by an incident generation entity, which models the aggregate behavior of customer incident reports. The simulated IT support organization behavior emerges from the interaction of its support groups and incident generation entity.

The open queuing network model has several benefits, such as capturing the dynamics of IT support organizations (in terms of waiting times, service times, throughput, queue lengths, response times and utilization) both at the system level and at the single support group level. In addition, the model parameters can be easily obtained by user input or statistical inference from incident traces (see next subsection).

Finally, the adoption of the open queuing model allows for using results of queuing theory to analytically evaluate some performance metrics such as the expected number of visits to each support group and the expected throughput [19].

To re-enact the IT support organization model, SYMIAN exploits a discrete event simulation approach. In fact, the organization models should be able to deal with incident (re)prioritizations, real incident arrival traces, and realistic, i.e., non-24x7, operator work shifts. Those features would have been extremely difficult (or even impossible) to realize with an analytic approach. In addition, a simulator can easily analyze the dynamic behavior of the system, while the complexity of analytic queuing network models makes them more suitable to investigate asymptotic steady-state behavior.

To model the relationships between support groups and, consequently, the routing of incidents through the simulated organization, SYMIAN uses a stochastic transition matrix. Each element t_{ij} of the transition matrix T represents the probability that a ticket will be forwarded from support group g_i to support group g_j. To consider the interactions of IT support organization with the outside world (arrival and departure of incidents), an extra virtual state 0 is introduced in the transition matrix. This allows to define the incident arrival vector $(a_i) = (t_{0i})$ for $i > 0$ and the incident closure vector $(c_i) = (t_{i0})$ for $i > 0$, whose elements respectively represent the probability that an incident will arrive into the organization at support group g_i and that an incident will be closed at support group g_i.

The transition matrix then becomes:

$$T = \begin{bmatrix} 0 & a_1 & a_2 & \cdots & a_N \\ c_1 & t_{11} & t_{12} & \cdots & t_{1N} \\ c_2 & t_{21} & t_{22} & \cdots & t_{2N} \\ \vdots & \vdots & \vdots & & \vdots \\ c_{N-1} & t_{N-1,1} & t_{N-1,2} & \cdots & t_{N-1,N} \\ c_N & t_{N1} & t_{N2} & \cdots & t_{N,N} \end{bmatrix}$$

In most organizations, the only entry point to the system is a single support group realizing the Help Desk function. In that case, assuming without loss of generality the Help Desk to be support group g_1, it is $a_1 = 1$ and $a_i = 0$ for $i \neq 1$.

Notice that the probability matrix has to satisfy the invariants:

$$\sum_{i=1}^{N} a_i = 1, \quad \sum_{i=1}^{N} c_i = 1 \tag{1}$$

and:

$$\forall i \in 1, 2, \ldots, N, \quad \sum_{j=1}^{N} t_{ij} = 1 \tag{2}$$

This model builds on top of the assumption of *memoryless* incident routing, i.e., that the probability of incident transition to a specific support group is independent of the history of re-assignments up to that moment. While this assumption allows for a considerable simplification of the model, extensive tests performed on real-life data with the SYMIAN tool demonstrated that the model behaves with good fidelity.

Each support group is modeled as having a specific set of operators, a queue of incoming tickets, a service time distribution, an operator assignment policy, and an incident prioritization policy. In turn, every operator has a work shift and is unavailable when off-duty, and a skill parameter that skews his effectiveness in working on incidents. In the context of a specific support group, different incident prioritization can be applied, such as first-come-first-served (FCFS), shortest-remaining-time (SRT), and fixed-priority-with-preemption (FPP). Finally, different operator selection strategies are also considered, such as assignment to the best-skilled operator, and assignment to the first-available (random) operator. It is also possible to extend the tool by defining custom incident prioritization and operator selection policies.

B. *Parameter Identification*

In order to recreate accurately the behavior of real-world organization, it is indispensable to identify carefully the configuration parameters of the IT support organization model. While some low-level parameters such as operator assignment policies and incident prioritization policies need to be provided by the user, the most important parameters can be inferred by analyzing traces of incidents obtained from the operational logs of real IT support organizations, where available.

SYMIAN is capable of inferring model parameters, such as the number of support groups, the transition matrix, and the operator set and service time distribution at each support group, using incident traces of real-world organization containing the time of arrival and departure at each visited support group for each incident. More specifically, SYMIAN computes the number of incident transitions between support groups and derives the transition matrix trough normalization with respect to the total number of incidents in the trace. To find operator number and service time distribution at each support group, SYMIAN applies the inference algorithm proposed by Park and Kim [20], assuming 24x7 operator work shifts, a first-available operator assignment policy and a first-come-first-served (FCFS) incident prioritization policy.

An accurate modeling of the incident arrival process is also of critical importance for realistic simulations. To ensure a realistic input for the simulation, one possibility is to use incident traces from real IT support organizations. However, considering only real incident traces would limit the applicability of the simulative approach to a small set of predefined input, thus preventing its use to verify how the modeled organization would behave under heavy incident load or a specific inter-arrival pattern. As a result, there is the need to consider synthetic incident generation according to configurable stochastic patterns, possibly reproducing the behavior of real customer service reports.

To this end, SYMIAN allows to infer the univariate distribution of incident inter-arrival times from transactional data of real-world organizations. A large set of continuous random probability distributions, such as CityplaceNormal, Uniform, Log-Normal, Pareto, General Pareto, etc., is available. Several discrete random probability distributions commonly found in queuing system, such as Poisson and Non-Homogeneous Poisson Process, are also supported. Incident inter-arrival time is also stochastically modeled according to a random variable distribution.

Incidents can optionally be assigned attributes, such as category and priority, which will be used later to modify their service time, prioritization, and operator selection policies as they arrive in each support groups.

C. The SYMIAN Discrete-Event Simulation Process

The SYMIAN simulation process works as follows:

1) The incident generator starts by injecting incidents into the modeled IT support organization, according to the user-specified parameters (incident inter-arrival time distribution or real-world organization incident trace source file, incident priority, etc.).
2) Each of the simulated incidents created in the previous step is forwarded to the appropriate support group (first assignment group), according to the probability values specified in the incident arrival vector.
3) When a new incident arrives at support group g_i, the simulator computes the duration of its service time according to the configured service time distribution for g_i. SYMIAN will then put the incident in the support group incident queue, according to the configured incident prioritization policy for g_i. The simulator will then try to assign the incident at the top of the queue to an operator according to the configured operator selection policy for the support group, possibly pre-empting other (lower priority) incidents currently being serviced.
4) When an operator starts working on a ticket, the simulator computes the duration of the operator transaction as the minimum between the remaining incident service time and the time to the end of the operator's work shift, and marks the operator as busy for said duration. If the operator's work shift ends before the incident service time is expired, the incident is put back in the incoming incident queue (again, according to the incident prioritization policy configured for g_i) and the operator is marked as off-duty until the beginning of his next work shift.
5) When the incident service time expires, the simulator marks the operator as available and the incident as transition-ready.
6) When a previously busy or off-duty operator becomes available, SYMIAN checks if it can assign him a ticket currently in the incoming incident queue, according to configured incident prioritization and operator selection policies. If so, the ticket is removed from the queue and assigned to the operator, and the simulation process resumes from step 4.
7) When an incident at support group g_i is marked transition-ready (step 5), the simulator closes it with probability c_i. If the incident is closed, the simulator collects the necessary information about the incident. Else, the incident is transitioned to support group g_j with probability (t_{ij}).

Because of the stochastic nature of the process being simulated, SYMIAN performs multiple simulation runs and returns an average of the IT performance metrics collected.

V. Using SYMIAN for Optimizing Performance

SYMIAN analyzes simulation outcomes to locate potential performance bottlenecks in IT support organizations. To this end, SYMIAN uses a set of predefined IT performance metrics that were designed to consider both routing effectiveness and support group efficiency. In addition, SYMIAN also allows users to define custom performance metrics to consider in the bottleneck location process.

SYMIAN evaluates the selected performance metric and assigns each support group a bottleneck score: a value that represents the impact of that support group performance on the whole system. Support groups with the highest bottleneck score are a good candidate for improvement.

SYMIAN gives several options to optimize the performance of IT support organizations. Some of the operations available to IT managers, such as support group removal, support group creation, merging of two support groups, and splitting of a support group, have a major impact on the IT support organization and require the redefinition of the transition matrix. Merging support groups is an important instrument to avoid unnecessary bouncing back and forth of tickets between different support groups, thereby wasting unnecessary queuing time. On the other hand, splitting groups is useful – possibly along with re-training – when groups grow too large and too diverse, and begin to attract tickets of many different categories, thereby losing the advantage of specializing IT operators for certain ticket categories. Other less invasive operations, such as support group re-staffing, work shift re-definition, incident prioritization and/or operator assignment policy modification, are also available.

We describe the optimization options supported by SYMIAN in the following subsections.

1) *Removing support groups*

When removing a support group g, the arrival and closure vectors and the transition matrix will be updated to reflect the support group deletion, and subsequently renormalized to satisfy invariants (1) and (2).

Supposing - without loss of generality - that g be the $N-th$ group, the new closure vector is then given by $(c'_i) = (c_i)/(1-c_N)$ (excluding the trivial case where $c_N = 1$). A similar transformation is applied to the incident arrival vector. The new transition matrix is then $(t'_{ij}) = (t_{ij})/(1 - t_{Nj})$ (again excluding trivial cases).

2) *Creating support groups*

When creating a new support group (without loss of generality indexed $N + 1$, the user will be required to provide the scalars a_{N+1} and c_{N+1}, representing the incident arrival and closure probability at the group; a vector (to_j) representing the transition probability from the group to each of the other groups; and a vector $(from_i)$, representing the transition probability from each of the other groups. The incident arrival and closure vectors get extended with a_{N+1} and c_{N+1} and re-normalized as above. The rows of the transition matrix are first updated according to:

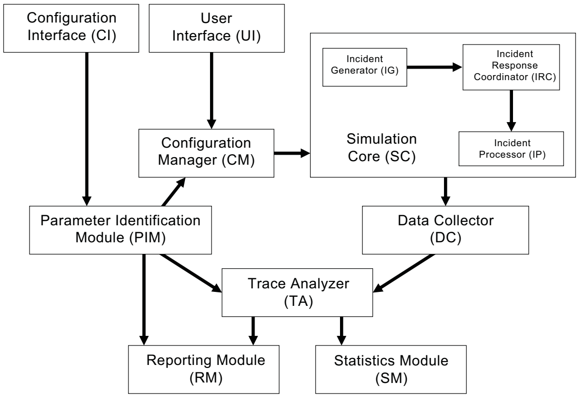


Fig. 4. Architecture of the SYMIAN tool.

$$(t'_{ij}) = (t_{ij}) * (1 - from_i) \quad \forall i, j \in 1 \dots N$$

so that:

$$\sum_{j=1}^{N} t'_{ij} = \ 1 - from_i \quad \forall i \in 1 \dots N.$$

At this point the matrix gets extended with the row $(t'N + 1, j) = (to_j)$ and the column $(t'i, N + 1) = (fromi)$. The reader can verify that the invariants are now satisfied for$(t'ij)$ so extended.

3) *Merging support groups*

When merging two support groups g_1 and g_2, SYMIAN requires information on the volume of incidents processed at each group. It is expected that SYMIAN will have that information available because of historical computation or previous simulations. If so, it will suggest those values to the user, letting him override them. Else the user will be required to input estimated values for the incident volumes.

The merging operation is equivalent to the removal of each group, followed by the creation of a new group that will have arrival, closure and transition probabilities calculated as follows. If $r = v_{g1}/v_{g2}$ is the ratio between the volume of incidents processed at each group, the arrival and closure probabilities of the addendum group will be $r \cdot a_{g1} + (1-r)a_{g2}$ and $r \cdot c_{g1} + (1 - r)c_{g2}$. The (to_j) and $(from_i)$ vectors for the addendum group will be respectively $(to_j) = r \cdot (t_{g1,j}) + (1 - r)(t_{g2,j})$ and $(from_i) = r \cdot (t_{i,g_1}) + (1 - r)(t_{i,g_2})$.

4) *Splitting support groups*

When splitting an existing support group g in two smaller support groups, SYMIAN will require the user to state the ratio r of the incident volume that each new group is expected to have. SYMIAN will suggest setting this ratio at $\frac{1}{2}$ by default. The splitting operation is equivalent to the removal of the old group, followed by the addition of two new groups that will have the arrival probabilities $a_g \cdot r/(1+r)$ and $a_g \cdot 1/(1+r)$; the same closure probability as the original group c_g; (to_j) vectors that are identical to the original group's transition matrix column (t_{gj}); and $(from_i)$ vectors that are given by $r \cdot (t_{ig})$ and $(1 - r)(t_{ig})$, respectively.

5) *Changing staffing levels, work shifts, and incident management policies*

SYMIAN allows the user to change staffing level in the IT organization support group. To this end, the tool will require the user to state, for each support group to consider, the new absolute value of staffing level or a multiplying constant that incrementally defines the new staffing level with respect to the previous one.

In addition, IT managers can also change operator work shift, at both the support group level or at the single operator granularity. Among the available options, there are both 24x7 work shifts (where operators are always on duty) and 8-hour-per-day work shifts that model service times more realistically, also considering the operator's time zone of residence.

Finally, SYMIAN allows IT managers to change the incident management policies at each support group.

VI. SYMIAN: Architecture and Implementation

The architecture of SYMIAN is depicted in Fig. 4, showing the main components of the tool: the Configuration Interface (CI), the User Interface (UI), the Configuration Manager (CM), the Parameter Identification Module (PIM), the Simulator Core (SC), the Data Collector (DC), the Trace Analyzer (TA), the Statistics Module (SM), and the Reporting Module (RM).

The Configuration Interface component allows users to configure the IT support organization to simulate. CI allows users to explicitly input model parameters such as the set of support groups, operator number and specialization for each support group, transition matrix, etc. Alternatively, CI permits to estimate the model parameter from real incident traces, by applying the statistical inference functions provided by PIM on transactional data. CI interfaces with CM to save current configuration of the IT support organization.

The User Interface component allows users to load simulation parameters from a file, to change current simulation parameters, to save current simulation parameters to file, and to start simulations. UI provides both an interactive textual and a non-interactive command-line interface.

The Configuration Manager takes care of the simulator configuration, enforcing the user-specified behaviors, e.g., with regards to verbosity of tracing information, and simulator parameters, e.g., the characterization of incident generation, the number and size of support groups, and the relationships between support groups, in the domain specific model recreated by the Simulator Core component.

The Parameter Identification Module provides statistical inference functions that can determine whether the samples in a given data set are distributed according to a known random variable distribution. If so, PIM can determine the probability distribution function parameters using maximum likelihood estimation. PIM can also infer the number of servers in an unobservable queue from the analysis of transactional data, exploiting Kim and Park's algorithm [20].

The Simulator Core component implements the domain specific model. SC has three sub-components: Incident Generator (IG), Incident Response Coordinator (IRC) and Incident Processor (IP). The Incident Generator generates incidents according to a random distribution pattern which follows user-specified parameters, and injects them into the system. The Incident Response Coordinator receives incidents and dispatches them to the processing domain entities (support groups), which are in turn implemented by the Incident Processor.

The Data Collector component collects data from the simulation that can be post-processed to assess the performance of incident management in the modeled organization. In particular, DC performs an accurate monitoring of support group status, in terms of incoming incident queue size and operator activity, and a careful tracking of incidents status. DC saves its simulation results data in a file that users can then analyze with the Trace Analyzer component.

Finally, the Statistics Module and the Reporting Module respectively provide basic statistics and reporting functions for the higher layer components.

SYMIAN is implemented in the Ruby (*http://www.ruby-lang.org/*) programming language. We chose Ruby for its remarkable extensibility and its support for meta-programming. Ruby's capability to easily redefine the behavior of time-handling classes in the standard library made possible the implementation of a simulated clock that models the flow of simulation-time in a very similar way to what happens in real life. In addition, Ruby's meta-programming enabled the definition of domain-specific languages and their use in the realization of several simulator components. These have proved to be particularly effective development techniques.

The availability of a wide range of high-quality scientific libraries was also a major reason behind the adoption of Ruby. In particular, SYMIAN exploits the GNU Scientific Library (*http://www.gnu.org/software/gsl/*), via the Ruby/GSL bindings, for high-quality random number generation, and it integrates with the R statistics framework (*http://www.r-project.org/*) to perform complex statistical analysis of simulation data and to produce high quality plots of the simulation results. Finally, SYMIAN exploits Ruby facilities to import configuration parameters and export simulation results in the XML, YAML, and CSV formats, in order to ease integration with external software for the automation of multiple simulation runs and with scientific tools for post processing of simulation results.

VII. Experimental Results

This section presents an experimental evaluation of the SYMIAN effectiveness in the performance analysis and optimization of a real-life IT support organization. For this experiment, we used data provided to us by the Outsourcing Services Division of HP. HP Outsourcing manages, among other IT services, the Help Desk function on behalf of various enterprise customers. The data used for this experiment comes from the subset of the organization serving a single enterprise customer from the financial services industry, whose name will be disguised as *BailUsOut* in the remainder of the paper.

Having a global 24/7 presence, HP Outsourcing faces the daily challenge of supporting multiple environments for multiple customers in disparate geographies. Hundreds of support groups employing thousands of engineers provide support to clients all around the world. In particular, 34 support groups are dedicated to *BailUsOut*, and 38 more groups have shared responsibilities across multiple enterprise customers and deal with tickets generated by *BailUsOut*. These support groups are geographically distributed and each work at their own local time-zone.

A. Model Inference and Validation

We were able to obtain database logs of incidents for a 6-month period, consisting of data for more than 23,000 incidents. For each incident, the data carried transactional information about the arrival and departure times at each visited support group. Since the dataset did not include any information about incident classification or prioritization, we considered every incident as belonging to the same category and priority.

Using SYMIAN statistical analysis and inference functions on transactional data, we were able to construct a reasonably accurate model of the *BailUsOut* IT support organization. First, we constructed the escalation matrix and derived the stochastic transition matrix by normalization. Then, we modeled each support group as a G/M/s first-come-first-served (FCFS) queue. Since we did not have any information on the number of operators in each support group, we had to infer that parameter using Kim and Park's algorithm [20]. In addition, transactional data did not contain information about actual service times, but only on the aggregate waiting plus service times. Thence, for each support group we had to estimate the mean service time parameter. More specifically, for support groups dealing with a large number of incidents we made the assumption – later confirmed by the analysis of transactional data – that the number of incidents assigned to the support group is always significantly higher than the number of operators. From there we estimated service rate as the inverse of the average incident inter-departure time. For support groups with a small number of incidents, instead, we simply estimated the mean service time as the average sojourn time. We then introduced the estimated service rate parameters in SYMIAN and we used the tool's advanced configuration functions to apply corrections to the estimated parameters so that the mean total service time observed in a simulation run matches the one estimated from transactional data. Finally, to model incident inter-arrival times we used a random exponential probability distribution with a rate parameter estimated from transactional data.

At the end of the configuration process, we ran a first simulation to evaluate the accuracy of the model we built for the *BailUsOut* IT support organization. More specifically, we ran 40 different simulation rounds and considered the mean of each performance metric we were interested in. To obtain the metric distributions, we considered the sample values collected across all the simulation rounds. The simulation covered 6 months of simulated time, plus a warm-up time of 15 days, introduced in order to prime the simulation environment in order not to take unrealistic measurements on a cold start. Events such as incident arrivals, closures, and escalations occurred during the warm-up time were discarded and not taken into account for the evaluation of the organization performance metrics.

We then compared the outcome of the simulation with the transactional data and verified that SYMIAN could reproduce the behavior of the *BailUsOut* IT support organization

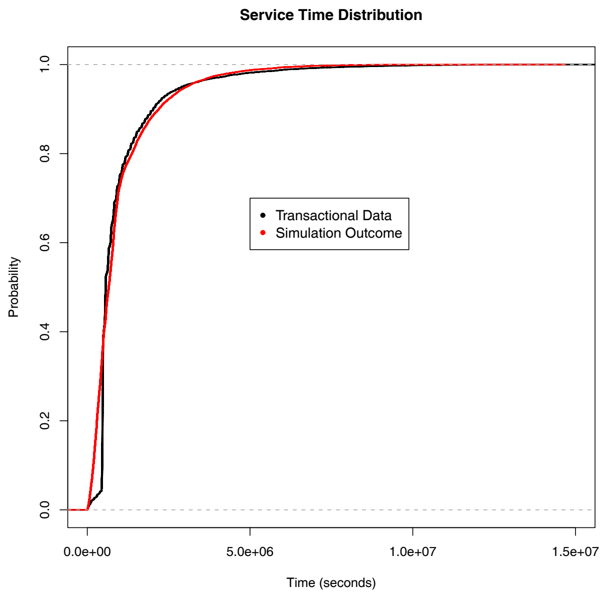


Fig. 5. Comparison of service time distribution for transactional data and simulation outcome.

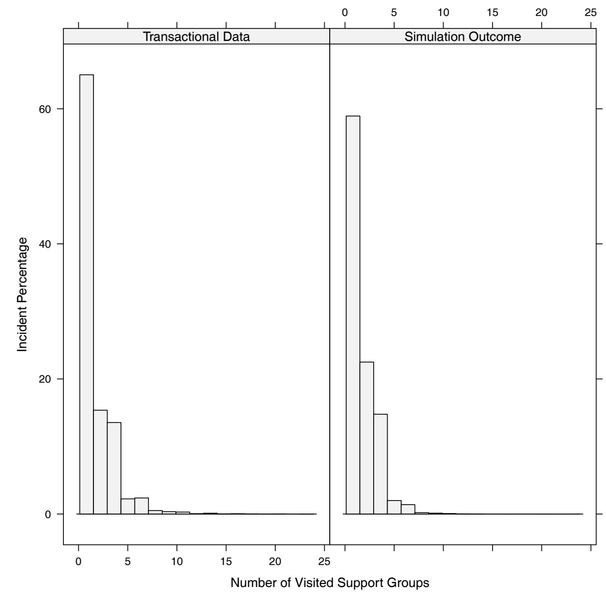


Fig. 6. Comparison of visited support group numbers per incident for transactional data and simulation outcome.

with reasonably good fidelity. Fig. 5 shows the comparison of the (empirical) cumulative distribution functions of total incident service times; Fig. 6 that of the visited support group numbers per incident; and Fig. 7 that of the number of received incidents at each support group for transactional data and simulation outcome.

In order to verify the accuracy of the model, we performed a statistical null hypothesis analysis of the results. In particular, we ran a Wilcoxon sum rank test [30] to find whether the service times from the historical and simulation outcome belong to the same distribution. As the result of the test is negative, with a p-value smaller than 2.2E-16, we have to reject the null hypothesis and conclude that the samples analyzed belong to different distributions. We then analyzed the densities of historical and simulated service time trough kernel density estimation, and verified that the match is very good in the distribution tails (see Fig. 8), although not as good for low service times. This is also confirmed by the mismatch for low service times in Fig. 5. Similar results have been obtained for received incidents and hops.

By applying kernel density estimation to analyze sojourn time at the various support groups, we have discovered that support groups in the *BailUsOut* IT support organization have different (usually three) service priorities. As a result, the approximation of support group as G/M/s queues tends to overestimate total service times. While the G/M/s model represents a rough approximation for the BailUsOut IT support organization, we believe that it is valid in the context of what-if scenario analysis. In fact, G/M/s queues have been extensively studied in literature, and they are well known and easy to work with. More accurate models, e.g., based on multiple priority queues, would be significantly more difficult to reconstruct from transactional logs and, most important, to work with. In fact, the inference of the number of operators in each support group and their allocation to work on incidents of different queues would be very challenging. In addition, the reallocation of workforce at the single support group level would require to consider a much larger number of parameters, therefore significantly complicating the performance tuning task for SYMIAN users.

B. Evaluation of Configuration Changes

Following the model construction and validation process, we now present an experimental evaluation of the effectiveness of SYMIAN in the performance analysis and improvement of a real IT support organization. To this end, we have applied SYMIAN to minimizing the service disruption time in the context of the *BailUsOut* organization, with the constraint of preserving the current number of operators. As a result, the objectives of the performance improvement process are the maximization of the mean incidents closed daily (MICD) metric, as well as the minimization of the mean time to resolution (MTTR) metric.

To locate the performance bottleneck, we ran a performance analysis of the incident management process in the modeled *BailUsOut* IT support organization. More specifically, we configured SYMIAN to calculate the bottleneck score for each support group SG_i as:

$$BS_i = (FI_i + FO_i) \times RI_i \times \overline{WT_i}(3),$$

where FI_i and FO_i respectively represent the fan in and fan out, RI_i the number of received incidents, and $\overline{WT_i}$ the mean waiting time at support group SG_i.

Formula (3) assigns a higher score to groups with high incident volumes and a large number of connections. While this is appropriate for the *BailUsOut* IT support organization, other organizations or different performance improvement objectives might require considering different bottleneck score calculations, e.g, taking into account queue size instead of queuing time or ignoring the fan in and fan out metrics.

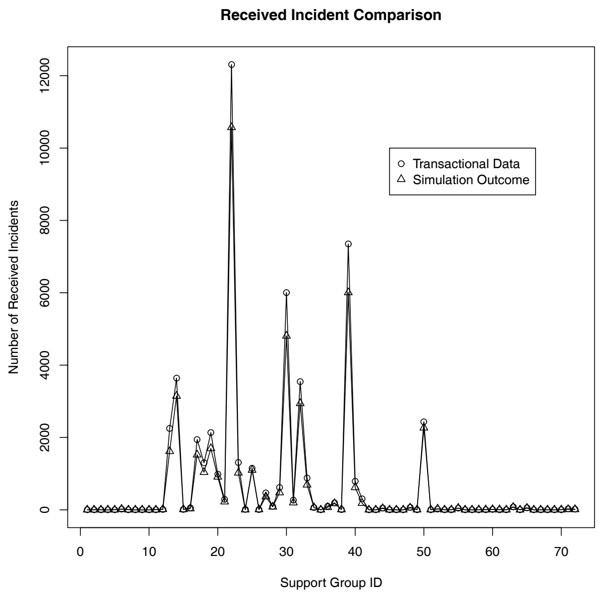


Fig. 7. Comparison of received incidents at each support group for transactional data and simulation outcome.

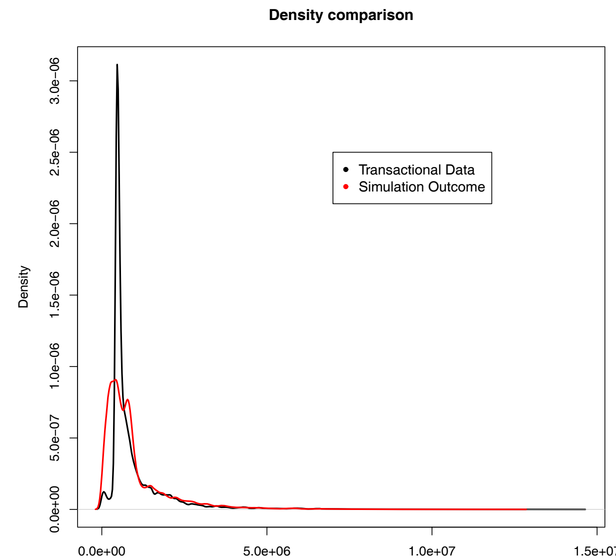


Fig. 8. Comparison of service times density for transactional data and simulation outcome.

Fig. 9 provides the bottleneck score calculated for each support group using formula (3). From the visual analysis of the plotted scores, it is easy to realize that support groups *SG22* and *SG39* are the major performance bottlenecks of the organization. The highly uneven bottleneck score distribution suggests that the local optimization at support groups *SG22* and *SG39* might be a very effective method to improve the performance of the `BailUsOut` IT support organization.

In order to improve the organization performance, we tried increasing the operator efficiency at support groups *SG22* and *SG39* in incremental steps, emulating an improvement in operator performance that could be obtained in real life by re-training technicians. For every change in the IT support organization, we launched a new 40-round simulation to assess the impact of the change on the organization performance. Table I provides a comparison of the MICD and MTTR performance metrics measured in the simulations, showing the average of those metrics over the 40 simulation rounds and the relative 95% confidence intervals. The simulation outcome demonstrates that the bottleneck location and removal is a very effective method to improve the whole system performance. In particular, the `BailUsOut` IT support organization exhibited improvements of the MICD metric up to 2.05% and a decrease of the MTTR metric up to 17.18%.

We also considered the effect of support group merging and splitting optimization options. The only viable support groups candidates for merging are SG22 and SG50. In fact, the analysis of transactional logs shows that SG50 processes 6.98% of the incidents, with 82.95% of them also going through SG22. We have therefore changed the configuration of the IT support organization by merging the two groups and ran a new 40-round simulation. The results show that the merging operation is very promising from the performance improvement perspective, leading to a 20.4% decrease of the MTTR metric (796772.2 ± 18697.83) as well as a 2.46% increase of the MICD metric (143.70 ± 0.31). However, we note that such a significant improvement represents an ideal case and calls for further investigations – that the limited information in the `BailUsOut` dataset prevent us to conduct. In fact, merging SG50 with SG22 might be infeasible in practice – SG50 is a support group shared with other IT support organizations while SG22 is a dedicated support group – or even counterproductive and unadvisable. Indeed, SG22 is already the support group with the highest workload, and merging SG50 into it would make the workloads in `BailUsOut` even more unbalanced.

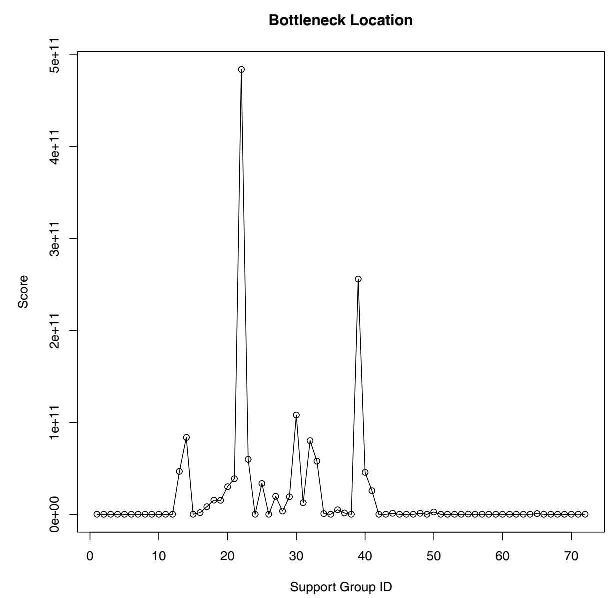


Fig. 9. Bottleneck score for each support group.

On the other hand, support groups with the highest workloads such as SG22 and SG39 represent the best candidates for

TABLE I
COMPARISON OF PERFORMANCE METRICS MEASURED IN THE SIMULATIONS OF CONFIGURATION CHANGES TO THE BAILUSOUT IT SUPPORT ORGANIZATION. THE RESULTS PROVIDED FOR THE MTTR AND MICD METRICS ARE MEAN VALUES AND 95% CONFIDENCE INTERVALS CALCULATED OVER 40 SIMULATION RUNS. THE STATISTICAL SIGNIFICANCE OF THE IMPROVEMENTS FOR ALL THE PRESENTED MTTR AND THE MICD METRICS WAS VERIFIED USING PAIRED SAMPLES T-TEST WITH A 95% CONFIDENCE INTERVAL.

Change in workforce specialization	MTTR (seconds)	MICD	MTTR Improvement	MICD Improvement
None	1001027 ± 18315	140.25 ± 0.34	NA	NA
3% improvement at SG22	936148 ± 20286	141.62 ± 0.29	6.48%	0.98%
2% improvement at SG22, 1% improvement at SG39	950424 ± 19242	141.40 ± 0.30	5.06%	0.82%
5% improvement at SG22	878793 ± 19703	142.61 ± 0.29	12.21%	1.68%
3% improvement at SG22, 2% improvement at SG39	898788 ± 24892	142.10 ± 0.27	10.21%	1.32%
7% improvement at SG22	829021 ± 24892	143.12 ± 0.33	17.18%	2.05%
4% improvement at SG22, 3% improvement at SG39	858662 ± 18375	142.73 ± 0.28	14.22%	1.77%

splitting. Unfortunately, the limited data in the *BailUsOut* dataset does not provide us the necessary contextual information to make realistic and sensible hypotheses on how to divide incidents between the new support groups, e.g., forwarding the incidents of a given category to the first support group and all the other incidents to the second support group. With the limited information we have, every effort to split a support group would result in an arbitrary operation and produce unrealistic, and therefore uninteresting, outcomes.

During the performance improvement process, we did not consider costs related to the actual implementation of the proposed changes to the organization. That would require an in-depth business impact analysis on the organization behavior, and is outside the scope of this paper. The reader interested in the application of SYMIAN for the business-impact driven performance optimization of IT support organizations is referred to [31].

The application of SYMIAN to a large-scale real-life IT organization gives an indication of the effectiveness of the SYMIAN tool for the performance optimization of the incident management function in IT support organizations.

Finally, in our experiments with the simulated version of the *BailUsOut* IT support organization in SYMIAN, we found that small changes to the configuration of support groups with a large fan-in and fan out can have a relatively large impact on the whole system behavior. As a result, we expect that local optimization might not be a very effective practice in IT support organizations with strongly interconnected support groups and an even distribution of workload. In this case, there is the need to adopt different optimization strategies that also consider the relationships between the different support groups.

VIII. RELATED WORK

The present work contributes to the up and coming research domain of Business-driven IT management (BDIM), which builds on the tradition of the research in network, system and service management. BDIM has been defined as "*the application of a set of models, practices, techniques and tools to map and to quantitatively evaluate interdependencies between business performance and IT solutions – and using the quantified evaluation – to improve the IT solutions' quality of service and related business results*". For a thorough review of BDIM, see [5]. Some notable early works in BDIM include applications to change management [6][7], capacity management and placeSLA design [8][9][10], network security [11], and network configuration management [12]. All these research efforts (with the exception of [12], which has a broader scope), limit their scope to the *technology* dimension of IT management, thereby focusing on the fine tuning of systems configuration and on the introduction of automation as means to improve the IT management function.

Within the BDIM domain, the present work belongs to a recently emerged research area that focuses on the other two fundamental dimensions of IT management than technology: *people* and *processes*.

The literature on business process management and related disciplines, such as business operation analysis provides an interesting background to compare the novelty of our approach against. At a first level of approximation, our work could be assimilated to approaches to business operation analysis that aim at improving business processes through collection of metrics and making inferences over them (see [13] for a notable example) or using simulation methods [14]. However, the main difference is that the techniques applied in business operation analysis are tailored to process descriptions consisting of many quite well defined steps with a limited array for alternative paths. Instead, the incident management process that we model in section II is characterized by a few simple steps (analyze the incident, operate on it, either escalate it or close it) and with a huge fan-out of alternative possibilities at each step. In the case study that we present in this paper, at the end of the incident processing step, escalation might happen towards one of the other 37 support groups or result in the incident closure. The relative simplicity of process being modeled, together with the complexity of the organization being modeled (that makes it so that each incident could be redirected to any of hundreds of support group) makes common business process management techniques either overkill or unapplicable to the case at hand.

As a representative example of improving IT processes, we cite Diao *et al.*'s recent studies on the estimation of labor cost and business value of IT services from the analysis of process complexity [15][16]. The main difference between our approach and theirs is that our focus is in achieving significant improvements in the performance of the organization through decision support and simulation techniques.

In this context, in previous works one of the authors has extensively studied the business impact of incident management strategies [3], using a methodology that moved from the definition of business-level objectives such as those commonly used in balanced scorecards [17]. With respect to those works, this paper follows a novel approach that for the first time proposes and implements a detailed model of IT support organizations to enable what-if scenario analysis.

The analysis of the incident management process and the IT support organization model that we present in this paper share is founded on a previous work presented in [4]. However, in this paper we push the modeling effort far beyond the definition of metrics for the performance assessment of IT support organizations that is conducted in [4], all the way to the design and implementation of the SYMIAN decision support tool.

In the context of modeling IT support organizations, Shao *et al.* have recently undertaken a research effort along the same lines as the work presented here ([21] [22]). Their EasyTicket system aims at optimizing the ticket routing in IT support organizations through machine learning techniques, and is based on a queuing network-based model similar to SYMIAN's. The EasyTicket rationale is that each type of incident has a specific resolver support group, and that therefore the IT support organization performance could be improved by routing incidents to their resolver support group as quickly as possible. In our experience, this assumption does not always hold in practice, as usually in real-life IT support organizations several support group need to cooperate in order to restore major service disruptions. SYMIAN, instead, enables a comprehensive analysis of the organization performance including both incident routing effectiveness and efficiency at the individual support group level.

While what-if analysis certainly represents an interesting approach to estimate the outcome of complex network management operations before putting them in practice, the only other research proposal that we know of in this context is WISE [23]. However, WISE focuses on the deployment of large Content Delivery Networks and adopts a machine learning-based approach to identify the subset of observable state variables that govern the system. Therefore, SYMIAN and WISE differ considerably for both the specific application domain and the model used to capture the essential behavior of the system to simulate.

Queuing networks allow to model complex systems, whose components can be characterized as a set of servers with a similar behavior. Queuing network-based models have been applied in a broad spectrum of research area such as computing [24], communications [25], transportation systems [26], health care [27], manufacturing systems [28], and supply chain systems [29].

Queuing network models are usually applied to evaluate predefined system parameters, such as throughput or delay, or to solve dimensioning or optimization problems. These models are also often based on restrictive assumptions about the service policy at the single queue level, to keep the model complexity low and analytically tractable. SYMIAN, instead, was designed to carefully reproduce the behavior of real-life IT support organization and to enable their performance analysis and improvement by applying user-defined metrics. In order to accurately model IT support organizations, SYMIAN integrates advanced support for the statistical analysis of transactional data and the inference of model parameters, and allows to consider complex service policies.

IX. Conclusions and Future Work

The performance optimization of large-scale IT support organizations can be extremely complex and lends itself to being tackled with decision support tools. This paper presented the SYMIAN tool for the performance optimization of incident management in IT support organizations. The application of SYMIAN in real-life IT support organizations demonstrates the tool effectiveness in the performance analysis and improvement process.

As part of the SYMIAN evaluation process, we demonstrated that open queuing network models could reproduce the behavior of real-life IT support organizations with a very high degree of accuracy. These promising results call for further study, that could bring to a deeper understanding of the performance of the incident management function in IT support organizations. Finally, the significant potential demonstrated by the SYMIAN decision support tool could be exploited by commercial applications.

We are currently working on a more comprehensive version of the SYMIAN tool that will link performance optimization metrics with key performance indicators or impact metrics that are meaningful at the business level, e.g., SLO violation penalties.

A. Acknowledgments

C. Bartolini would like to thank Gilad Barash and Lionel Fradin, for their invaluable help in the analysis of the incident management process, in particular with the definition of the metrics to consider for the performance optimization.

References

[1] United Kingdom Office of Government Commerce, "ITIL Service Delivery" and "ITIL Service Support," IT Infrastructure Library version 3, 2007.

[2] IT Governance Institute, COBIT 3rd edition, 2000, http://www.isaca.org/COBIT.htm

[3] C. Bartolini, M. Sallé, and D. Trastour, "IT service management driven by business objectives—an application to incident management," in *Proc. IEEE/IFIP Network Operations and Management Symposium (NOMS 2006)*, Apr. 2006.

[4] G. Barash, C. Bartolini, and L. Wu, "Measuring and improving the performance of an IT support organization in managing service incidents," in *Proc. 2nd IEEE Workshop on Business-driven IT Management (BDIM 2007)*, Munich, Germany, 2007.

[5] A. Moura, J. Sauvé, and C. Bartolini, "Research challenges of business-driven IT management," in *Proc. 2nd IEEE/IFIP International Workshop On Business-Driven IT Management (BDIM 2007)*, Munich, Germany.

[6] A. Keller, J. Hellerstein, J. L. Wolf, K. Wu, and V. Krishnan, "The CHAMPS system: change management with planning and scheduling," in *Proc. IEEE/IFIP Network Operations and Management Symposium (NOMS 2004)*, Apr. 2004

[7] J. Sauvé, R. Rebouças, A. Moura, C. Bartolini, A. Boulmakoul, and D. Trastour, "Business-driven decision support for change management: planning and scheduling of changes," in *Proc. DSOM 2006*, Dublin, Ireland

[8] S. Aiber, D. Gilat, A. Landau, N. Razinkov, A. Sela, and S. Wasserkrug, "Autonomic self–optimization according to business objectives," in *Proc. International Conference on Autonomic Computing*, 2004.

[9] D. Menascé, V. A. F. Almeida, R. Fonseca, and M. A. Mendes, "Business-oriented resource management policies for e-commerce servers," *Performance Evaluation 42*. Elsevier Science, 2000, pp. 223-239.

[10] J. Sauvé, F. Marques, A. Moura, M. Sampaio, J. Jornada, and E. Radziuk, "SLA design from a business perspective," in *Proc. DSOM 2005.*

[11] H. Wei, D. Frinke, O. Carter, *et al*., "Cost–benefit analysis for network intrusion detection systems," in *Proc. 28th Annual Computer Security Conference*, Oct. 2001.

[12] R. Boutaba, J. Xiao, and I. Aib, "CyberPlanner: a comprehensive toolkit for network service providers," in *Proc. 11th IEEE/IFIP Network Operation and Management Symposium (NOMS 2008)*, Salvador de Bahia, Brazil.

[13] F. Casati, M. Castellanos, U. Dayal, and M. C. Shan, "A metric definition, computation, and reporting model for business operation analysis," in *Proc. Advances in Database Technology - EDBT 2006, 10th International Conference on Extending Database Technology*, Munich, Germany, Mar. 2006

[14] K. Tumay, "Business process simulation," in *Proc. Simulation Conference 1995, Winter Volume*, Dec 1995 pp, 55–60.

[15] Y. Diao, A. Keller, S. Parekh, and V. Marinov, "Predicting labor cost through IT management complexity metrics," in *Proc. 10th IEEE/IFIP Symposium on Integrated Management (IM 2007),* Munich, Germany.

[16] Y. Diao and K. Bhattacharya, "Estimating business value of IT services through process complexity analysis," in *Proc. 11th IEEE/IFIP Network Operation and Management Symposium (NOMS 2008)*, Salvador de Bahia, Brazil.

[17] R. Kaplan and D. Norton, "The balanced scorecard: measures that drive performance," *Harvard Business Review*, vol. 70, no. 1, pp. 71-79, 1992.

[18] G. Bolch, S. Greiner, H. de Meer, and K. Trivedi, *Queuing Networks and Markov Chains: Modeling and Performance Evaluation with Computer Science Applications*, 2nd edition. Wiley 2006.

[19] U. N. CityplaceBath, *An Introduction to Queuing Theory*. Birkhäuser, 2008.

[20] Y. B. Kim and J. Park, "New approaches for inference of unobservable queues," in *Proc. 2008 Winter Simulation Conference*.

[21] Q. Shao, Y. Chen, S. Tao, X. Yan, and N. Anerousis, "Efficient ticket routing by resolution sequence mining," in *Proc. 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'08)*, Las Vegas, NV, USA, Aug. 2008.

[22] Q. Shao, Y. Chen, S. Tao, X. Yan, and N. Anerousis, "EasyTicket: a ticket routing recommendation engine for enterprise problem resolution," in *Proc. 34th International Conference on Very Large Data Bases (VLDB'08)*, Auckland, New Zealand, Aug. 2008.

[23] M. Bin Tariq, A. Zeitoun, V. Valancius, N. Feamster, and M. Ammar, "Answering what-if deployment and configuration questions with WISE."

[24] D. Liu, Y.-D. Cao, and C.-Q. Li, "Liana: a decentralized load-dependent scheduler for performance-cost optimization of grid service," *J. Supercomputing*, vol. 49, no. 1, pp. 127-156, July 2009.

[25] N. Bisnik and A. Abouzeid, "Queuing network models for delay analysis of multihop wireless ad hoc networks," *Ad Hoc Networks*, vol. 7, no. 1, pp. 79-97, Jan. 2009.

[26] D. Mitchell and J. MacGregor Smith, "Topological network design of pedestrian networks," *Transportation Research Part B: Methodological*, vol. 35, no. 2, Feb. 2001.

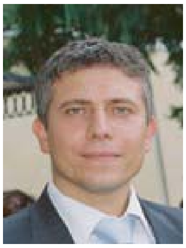
[27] N. Koizumi, E. Kuno, and T. Smith, "Modeling patient flows using a queuing network with blocking," *Health Care Management Science*, vol. 8, no. 1, Feb. 2005.

[28] V. Bhaskar and P. Lallement, "A four-input three-stage queuing network approach to model an industrial system," *Applied Mathematical Modelling*, vol. 33, no. 8, Aug. 2009.

[29] Q. Gong, K. K. Lai, and S. Wang, "Supply chain networks: closed Jackson network models and properties," *International Journal of Production Economics*, vol. 113, no. 2, June 2008.

[30] P. Kvam and B. Vidakovic, *Nonparametric Statistics with Applications to Science and Engineering*. Wiley, 2007.

[31] C. Bartolini, C. Stefanelli, and M. Tortonesi, "Business-impact analysis and simulation of critical incidents in IT service management," in *Proc. 11th IFIP/IEEE International Symposium on Integrated Network Management (IM 2009)*, June 2009, New York.



Claudio Bartolini is a principal scientist and research manager at HP Labs Palo Alto, USA. He manages a team of researchers in Palo Alto and Bristol, UK, focusing on predictive IT Analytics. He holds an M.Sc. degree in electronic engineering and from the University of Bologna, Italy and a Ph.D. in information engineering from the University of Ferrara, Italy He has published extensively in the areas of IT service management and related disciplines.

Cesare Stefanelli is a professor of computer science engineering in the Engineering Department of the University of Ferrara, Italy. His research interests include distributed and mobile computing, adaptive and distributed multimedia systems, network and systems management, and network security. He received a Laurea degree in electronic engineering and a Ph.D. in computer science engineering from the University of Bologna, Italy.

Mauro Tortonesi is a research assistant at the Engineering Department of the University of Ferrara, Italy. His research interests include distributed and mobile computing, QoS management, network and systems management, and industrial automation. He received a Laurea degree in electronic engineering and a Ph.D. in computer science engineering from the University of Ferrara, Italy.