Modeling IT Support Organizations Using Multiple-Priority Queues

Claudio Bartolini
HP Labs
Hewlett Packard
Palo Alto, CA, USA
claudio.bartolini@hp.com

Cesare Stefanelli and Mauro Tortonesi
Department of Engineering
University of Ferrara
Ferrara, Italy
{cesare.stefanelli,mauro.tortonesi}@unife.it

***Abstract*—As IT services grow more and more complicated, and their management becomes increasingly challenging, IT support organizations assume an essential role to ensure the delivery of Service Level Objectives. What-if scenario analysis represents a very effective tool for the performance optimization of IT support organizations, as it enables an iterative and customized performance optimization process. The problem of accurately modeling IT support organizations requires the development of sophisticated models as well as dedicated parameter inference techniques and tools. This paper presents a multiple-priority queuing model suited for the reenactment of IT support groups developed from the analysis of empirical evidence, as well as a powerful method to infer the model parameters. We applied our model to reenact the behavior of a real life IT support group with our Symian simulator. The results demonstrate that multiple-priority queuing models can reproduce real life IT support groups with a high degree of accuracy.**

***Keywords*—Decision support tools, Information Technology Infrastructure Library (ITIL), Incident management; IT support organizations.**

I. INTRODUCTION

Nowadays IT services are getting more and more complicated, and their management represents a major challenge. IT support organizations have a primary role in IT service management. Among other things, they are in charge of incident management, which ITIL defines as the process for restoring normal service operation after a disruption [1] [2]. While incident management represents a relatively simple process, enterprise-class IT support organizations usually implement very complex organizational, structural, and behavioral processes according to the strategic objectives defined at the business management level.

In order to ensure the alignment with business level objectives set by the business managers, the performance of IT support organization needs to be frequently assessed and, if needed, optimized. When re-organizations become necessary, it would be very important to evaluate alternative organization structures and incident management strategies carefully before enacting them.

What-if scenario analysis represents a very effective tool for the performance optimization of IT support organizations. What-if scenario analysis is a technique based on the construction of an accurate model of the system under evaluation and on its exploitation to reenact the system behavior with modified parameters. This 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. The performance of reenacted IT support organization configuration can be assessed through a set of predefined as well as user-specified performance metrics.

To produce sensible results, what-if scenario analysis requires accurate models of IT support organizations. In our previous work, we built a model of IT support organizations based on open queuing networks and multi-server first-come-first-served (FCFS) queues. Markovian models such as open queuing networks are particularly attractive for the reenactment of IT support organizations, as they represent a very good tradeoff between model complexity and model accuracy. In fact, our model proved capable of reenacting system-wide behavior of IT support organization very well, accurately capturing end-to-end metrics such as mean time to incident resolution and the distribution of the number of support groups visited by the incidents [3] [4].

The present work, instead, focuses on a different but related problem: the accurate modeling of a single IT support group. The modeling of single support groups represents a challenging task. Some of the support group dynamics that we have observed by analyzing transactional logs of real IT support organizations, such as the clustering of sojourn times around a few values, are difficult to accurately reproduce with FCFS queues. There is the need for a more sophisticated model that can better capture these details. Our experience with the analysis of real life IT support organization transactional logs suggests that multiple-priority queues are a promising candidate to model IT support groups.

Unfortunately, multiple-priority queuing models are significantly more complicated than those based on FCFS queues, and represent a formidable challenge from a model parameter identification perspective. Model parameter identification is further complicated by the fact that often the only data source available for the inference process is represented by transactional logs, obtained by information

inserted by operators through specific software such as HP IT Analytics. Transactional log data usually has a very limited information content, and often contains errors and inaccuracies. As a result, the adoption of an IT support organization model based on multiple-priority queues also requires the development of specific parameter identification methods.

This paper presents a multiple-priority queuing model of IT support groups developed from the analysis of empirical evidence, as well as a method to infer the model parameters. Our inference method leverages on the reenactment of support group through what-if scenario analysis techniques based on the multi-priority queuing model, using several (sets of) model parameters. The method then runs a statistical comparison of transactional logs from the real IT support organization with the outcomes of what-if scenario simulation to determine which set of parameters enables the most accurate reproduction.

We applied our model to reenact the behavior of a real life IT support group with our Symian simulator [4]. The results demonstrate that multi-priority queuing models can reproduce real life IT support organizations with a high degree of accuracy.

The rest of the paper is as follows. Section II describes the working of IT support organizations. Section III presents an analysis based on empirical evidence suggesting that multiple-priority queuing models are well suited to represent the behavior of IT support groups in real life organizations. Section IV introduces the multiple-priority queuing model we developed for IT support groups. Section V presents our method of inferring model parameters for multiple-priority queues from historic transactional logs. Section VI presents some insights on the prototype implementation of our model parameter inference method. Section VII gives an experimental evaluation of how our inference method can effectively enable the development of highly accurate models of real life IT support groups. Section VIII overviews related work. Finally, Section IX provides concluding remarks and future work considerations.

II. IT Support Organizations

IT support organizations are typically composed of a network of support groups, each employing a set of operators, with potentially different work shift schedules. 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.

The Help Desk provides the lowest support level and 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, whose technicians either fully repair the incident or "reassign" it to a different support group (usually escalating to a higher support level).

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 the pertinent information, such as 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 into 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, the incident is "resolved" and its lifecycle ends.

The characteristics of IT support organizations suggest to model them as open queuing networks [5], as it is done for telephone call centers [6] [7]. However, notice that IT support organizations have some peculiar characteristics that distinguish them from telephone call centers. In fact, in IT support organizations, there is no need to consider call blockings, abandonments, or redials. Instead, there is the need to consider routing of incidents through the system as well as complex incident prioritization policies since IT support organizations might serve many customers with different profiles, each one with a specific SLA.

III. Modeling IT Support Groups: the Case for Multiple-Priority Queues

In previous works we developed an extensive experience with the modeling of real life IT support organizations, through the in-depth analysis of transactional logs [3]. Our observations and findings suggest that the accurate reproduction of the IT support organization behavior at the single support group level calls for multiple-priority queuing models. To demonstrate this claim, we will present the experimental analysis of transaction logs from a real life IT support organization, provided to us by the Outsourcing Services Division of Hewlett Packard. 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 not be mentioned.

More specifically, to gain insight on the behavior of IT support groups, we examined the *sojourn time* metric, defined as the amount of time (including both waiting and service times) an incident spends in one single transaction in a specific support group. We considered sojourn times as their aggregate represents a rough indicator of the support group workload and because the data we obtained is only timestamped with arrival and departure of each incident at each support groups it visited, and therefore does not allow us to discriminate between waiting and service times.

Fig. 1 shows a comparison of arrival and sojourn times for the incidents visiting the largest support group in the IT support organization dataset (group 22). The figure clearly demonstrates that incidents arriving at about the same time can have very different sojourn times. This can be explained hypothesizing that incidents are processed concurrently, instead of in a sequential fashion, and represents the first indication that support groups may be better modeled using multiple-priority queues instead of FCFS queues.

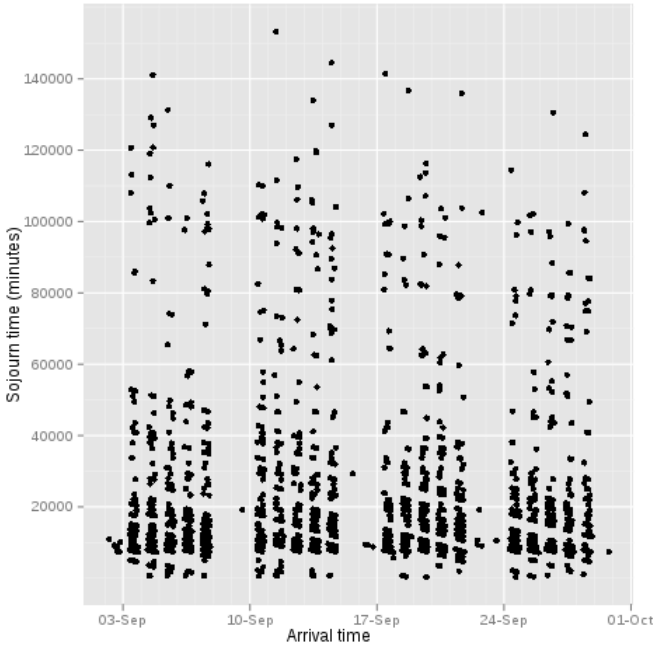

Figure 1. Comparison of arrival and sojourn times at support group 22 of the IT support organization.

We found further empirical evidence supporting the adoption of multiple-priority queues when analyzing the sojourn time distribution. Fig. 2 shows the distribution, obtained through kernel density estimation, of the incident sojourn times recorded at support group 22. The two peeks and the long tail in the distribution indicate that the large majority of incidents have a service time shorter than 2000 minutes or between 8000 and 10000 minutes.

To verify whether these peaks actually correspond to different service classes, we needed to rule out the possibility that they were the result of a temporarily anomalous behavior in the IT support group. We therefore divided the incidents in 3 categories: very fast incidents (incidents with sojourn time of 5000 minutes or less), fast incidents (incidents with sojourn times between 5001 and 10000 minutes), and slow incidents (incidents with sojourn time greater than 10000 minutes), and analyzed separately the incidents within each category. Fig. 3 plots, for every category, the arrival time of each incident against its sequential number (order of arrival). The figure shows that the arrivals, and therefore the servicing, of the above mentioned categories of incidents can be considered (with reasonable approximation) stationary. As a result, we can assume that the incidents in the very fast, fast, and slow categories are processed independently.

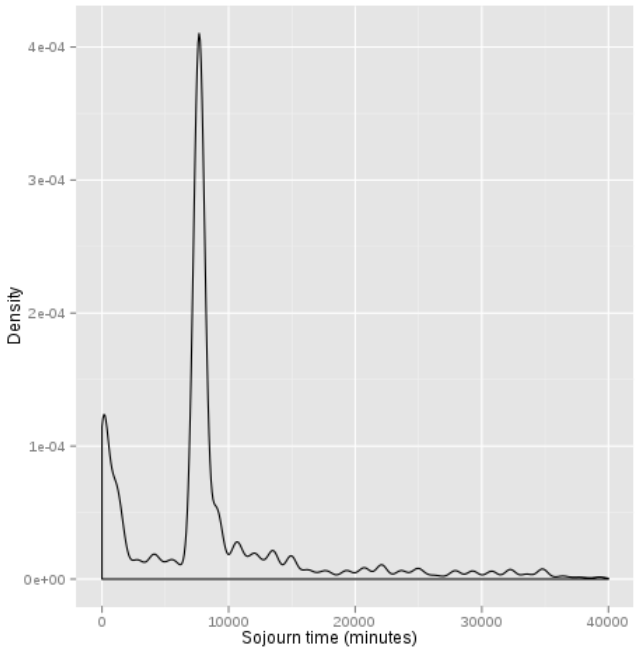

Figure 2. Distribution of sojourn times at support group 22.

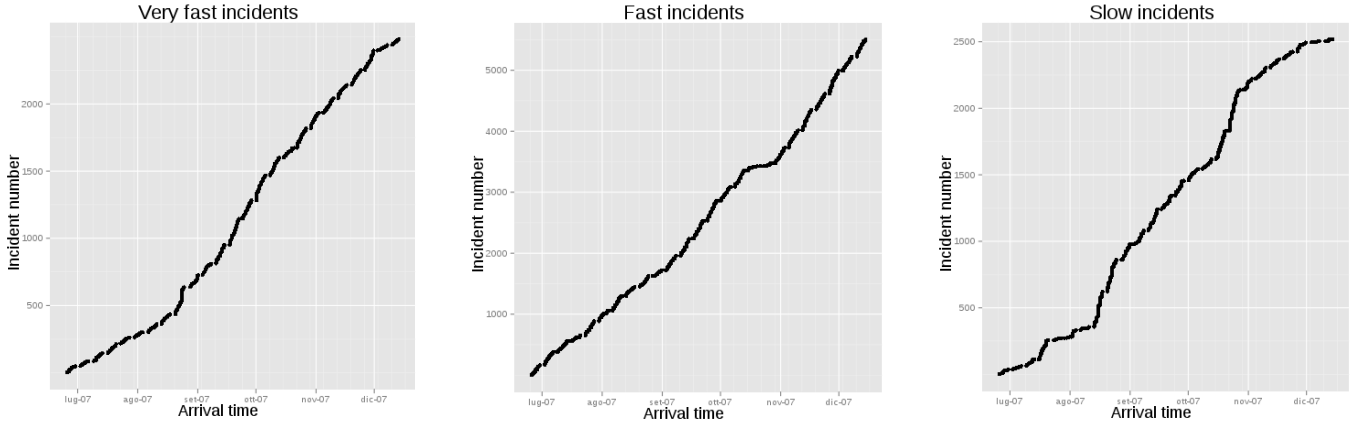

Figure 3. Quasi-stationarity of incident arrivals at support group 22.

This analysis suggests the adoption of a multiple-priority model to reenact the behavior of single support groups in IT support organizations. In particular, to accurately model support group 22 we should consider a model with 3 different priority queues: a high priority queue for very fast incidents, a normal priority queue for fast incidents, and a low priority queue for the remaining ones.

Unfortunately, we couldn't verify whether this hypothesis was sustained by experimental evidence with the set of tools at our disposal. In fact, the inference tools we adopted for the inference of FCFS based models were not applicable to the identification of multiple-priority queuing model parameters. More specifically, Kim and Park's algorithm [8], which we used to infer model parameters for FCFS queues, can only estimate the number of operators dealing with the tickets in each queue. Multiple-priority queuing models have a much larger number of parameters, and are significantly more complicated than FCFS model, thereby making the inference process much more challenging. As a result, they require more sophisticated and purposely developed parameter inference methods.

IV. A Multi-Server Multi-Priority Model for the Behavior of IT Support Groups

To accurately capture a wide range of possible behaviors, we developed a sophisticated multiple-priority queueing model for IT support groups. Our model reenacts the behavior of a support group according to a large set of model parameters, such as the *number of priority levels*, a *queue* and a *service time distribution* for each priority level, and the *set of operators*, and configurations such as a *priority assignment policy*, an *operator assignment policy*, and the operators' attributes *(workflow, skill set, etc.)*.

Fig. 4 provides a pictorial representation of how our multiple-priority model works. When an incident arrives to the support group, it is first assigned a priority level, according to the configured priority assignment policy, and forwarded to the corresponding priority queue. Priority assignment policies will typically consider incident attributes, such as their category, as well as the current support group state to make priority level decisions.

After the priority level assignment, our model attaches to the incident a service time, randomly sampled from the service time distribution associated to the selected priority level. Service time is an attribute that represents the amount of time that operators have to spend on the incident to finish the portion of work that competes to the current support group. Once assigned a service time, the incident is then inserted at the end of the corresponding priority queue.

As one or more operators become available to start servicing new incidents from the current priority queue, our model starts assigning them the tickets from the beginning of the queue. The incident-operator association procedure is performed according to the configured operator assignment policy, which specifies the set of rules to use for selecting which incident an operator should start working on when he becomes idle. The operator assignment policy is probably the most important configuration parameter of our model, as it can have a very large impact on the support group behavior. The simplest supported operator assignment policies divides operators in different, non-overlapping groups, one for each priority level, and assign incidents in a priority level to idle operators of the corresponding group in a round-robin fashion. However, it is also possible to reenact more sophisticated behaviors. For instance, one could configure an operator assignment policy that at the arrival of higher priority incidents preempts operators servicing lower priority ones, or that enforces the servicing of some particularly complicated incidents only by highly skilled and/or experienced operators.

Our model allows for very sophisticated operator management. In fact, the model provides support for operator work shift configuration, enabling an extremely realistic representation of the support group behavior. More specifically, the implementation of our model leverages on a simulated clock that reenacts the flow of simulation-time in a very similar way to what happens in real life. If an 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. In addition, as in real life IT support organizations it is not uncommon for operators to work on multiple tickets at the same time, our model also supports operators that simultaneously work on more than one incident. Finally, our model can be configured to consider specific operator skills that skew their ability to deal with incidents of a particular category. An operator with a 2.0 skill parameter in a specific incident category will take twice as less time to resolve an incident of that category than a normal operator.

When its associated service time is exhausted, the incident can leave the support group. In case further work is required to resolve the related service disruption, the incident will be escalated to a different support group. Else, the incident is considered resolved and its lifecycle ends.

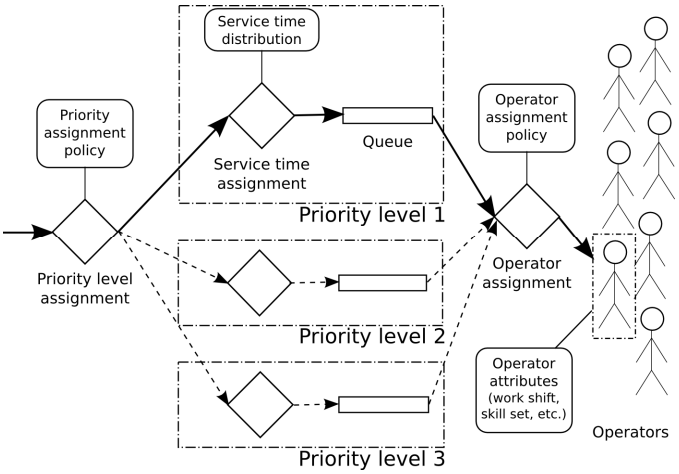

Figure 4. Multiple-priority queuing model for IT support groups.

V. A METHOD TO INFER MULTI-SERVER MULTI-PRIORITY QUEUING MODEL PARAMETERS FOR SUPPORT GROUPS

We developed a method to infer the parameters for a multi-server multi-priority queuing model of the support groups of a real life IT support organization from the transactional logs of incidents, as provided by incident tracking software suites such as HP Analytics.

Transactional logs are the most likely information source when building a model of a real life IT support group. Unfortunately, they typically contain a very small amount of information. In fact, they usually do not include any information about the root cause of incidents, that would allow to identify duplicated incidents and remove their records from the dataset. In addition, valuable incident attributes such as their description, classification, or prioritization are often missing from transactional logs. The dataset we analyzed in Section 3 does not even report whether incidents were actually closed on their last transaction. (We assumed that incidents whose last escalation from a support group happened some time before the end of the period covered by the dataset were closed, thus discarding a non-negligible portion of the data.)

Our method does not leverage on any knowledge or assumption about the inner working of IT support groups, but considers support group models as black boxes. This makes the method well suited to work with real life transactional logs, which essentially are a list of support groups' inputs and outputs. In addition, not relying on any assumption on the support group internals makes the method of rather generic applicability, enabling its adoption for the parameter identification of different queuing models as well (for instance, our method could be easily applicable to FCFS queues).

Our inference method considers support group models as stochastic objects that behave approximately as mathematical functions. The method assumes that two support group models are replaceable (or equal) if, when fed with the same identical input, they provide *stochastically similar* output. Building on this assumption, we realize the inference of model parameters through the statistical comparison of transactional logs from the real and the modeled support groups.

More specifically, the method aims at finding the set of parameters that minimize the stochastic distance between the set of historic sojourn times, i.e., those observed in the real life support group, and the set of sojourn times collected from the model-based reenactment of the support group.

To reenact the IT support group behavior through what-if scenario analysis techniques, our method leverages on the multiple-priority queuing model described in the previous Section. Given a model configurations, our method allows to identify which model parameters (such as the number of operators, the service time distribution for each priority queue, and even the number of priority queues to consider) that enable the most accurate reenactment of a real life support group.

In order to explore the possible model parameters space, the method conducts several reproductions of the support group behavior, each one with a different (sets of) model parameters. The method then determines which set of parameters enables the most accurate reproduction by running a statistical comparison of the transactional logs from the real life IT support organization with the outcomes of what-if scenario simulation.

Our method does not enable to identify configuration policies, but requires them to be specified up front. Theoretically, it would be possible to define a set of predefined policies and identify which of them allows to minimize the distance between sojourn times collected from historic data and simulation outcome, in the same way of model parameters. However, as policies can have a significant impact on the support group behavior, we believe it is best to let the user define them.

The first step of the model parameter inference process realized by our method consists in defining model configurations and constraints on the space of model parameters to consider. (For instance, in some cases it might be desirable to place limit boundaries on some parameters, such as the number of operators.)

The second step consists of extracting the incident arrival and sojourn times from the transactional logs of the real life organization. The historic incident arrival trace is then used as an input to the support group reenacted through what-if scenario analysis. The historic sojourn time trace is used for the stochastic similarity comparison.

As a measure of the stochastic distance between sojourn times collected from historic data and simulation outcome, we consider a purposely-designed non-parametric statistical metric: the *Wilcoxon distance*. We define the Wilcoxon distance between two sojourn time traces as the result of the application of a slightly modified version of the Wilcoxon sum rank test on the traces.

The Wilcoxon sum rank test is a non-parametric statistical test that was designed to verify whether two sample sets belong to the same population when no assumption on the distribution of the data could be made [9]. Despite model parameter identification represents an unusual application field for the Wilcoxon sum rank test, we have chosen to build our stochastic distance metric upon it because the rank-based statistical calculations at the heart of the test lend themselves very well to measure the degree of similarity between two distributions. In addition, the Wilcoxon sum rank test is a rather simple and well known statistical measure, that is widely available in most statistical software and easy to implement in programming environments that do not provide support for it already.

From a theoretical perspective, the model parameter inference procedure becomes a mixed-integer nonlinear programming problem, of the following kind:

$$\min W(x, y)$$
$$\textit{subject to } c(x, y) \leq 0$$
$$x \in X \subset R$$
$$y \in Y \subset Z$$

where the variable x represents the set of continuous parameter of the support group model, e.g., the service rate for each priority queue; the variable y represents the set of continuous

parameter of the support group model, e.g., the number of operators; the function *W* represents the Wilcoxon distance between the simulated sojourn time trace and the sojourn time trace collected from the transactional logs of a real life IT support organization; and the function *c* captures the set of constraints on the parameters, e.g., the number of operators in a support group should be positive and reasonably small.

VI. Implementation Insights

We realized a prototype implementation of the model parameter inference procedure described in the previous Section. The prototype architecture is depicted in Fig. 5. The main software components of the prototype are the What-if Scenario and the Optimizer modules. These components communicate with each other through a Web interface.

We chose to realize our prototype as a distributed Web application as this allows to decouple the software components, therefore enabling to implement each one of them in the most suited development platform. In addition, this architecture permits to speed up the inference process by using multiple What-if Scenario module, each one hosted in a dedicated server or in the cloud.

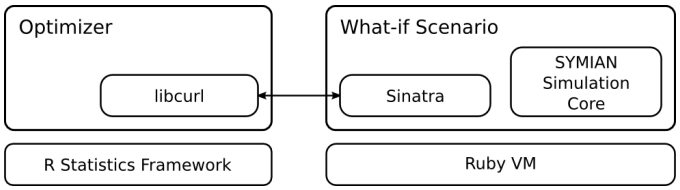

Figure 5. Architecture of the model inference prototype.

The What-if Scenario component enables to run what-if scenarios with the given model parameters, obtaining the sequence of incident departures as a result. In order to reenact the behavior of the support group, What-if Scenario leverages on a purposely-modified version of the Symian Simulation Core that enables priority queuing modeling and is optimized to reproduce only a single support group.

Both the What-if Scenario and the (modified) Symian Simulation Core components are implemented in the Ruby (*http://www.ruby-lang.org/*) programming language. Ruby's capability to easily redefine the behavior of time-handling classes in the standard library enabled 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, the Ruby support for metaprogramming allowed the definition of domain-specific languages and their use in the realization of several simulator components.

The availability of high performance, top-notch libraries for the agile development of Web applications was also a major reason behind the adoption of Ruby. More specifically, the Web interface provided by the What-if Scenario component leverages on the *Sinatra* library (*http://www.sinatrarb.com/*). While less known than the very popular Ruby on Rails framework, Sinatra represents the ideal solution for the easy integration of Ruby software components in a distributed Web application.

The Optimizer component is in charge of the whole model inference process. Optimizer adopts a heuristic approach to solve the mixed-integer nonlinear programming problem. Since the integer subspace Y of the model parameter space is of limited size, Optimizer performs an extensive search over Y, launching a nonlinear optimization procedure over the continuous subspace X for each of the possible values in Y. This effectively transforms the mixed-integer nonlinear programming problem in a series of nonlinear programming problems, that are significantly easier to deal with.

The Optimizer component is realized as an R application. R is a very advanced Open Source statistics framework (*http://www.r-project.org/*) that provides a large set of statistical functions. R enables the development of programs through a dedicated language designed for statistical computations, offers a plethora of extensions and libraries for both generic and niche statistical applications, and represents the de facto standard in modern statistical computing. Optimizer leverages on the R bindings to the *libcurl* library to interface with the What-if Scenario component and on the functions provided by the R framework to perform nonlinear optimization and to realize nonparametric statistical analyses (such as the calculation of the Wilcoxon sum rank test) of the outcomes provided by the What-if Scenario module.

VII. Experimental Evaluation

To evaluate how our inference method performs, we built a multiple-priority model of support group 22, the one analyzed in Section III, from the dataset provided to us by HP Outsourcing. We used the model to reenact the support group behavior in the Symian simulator and compared the simulation outcomes with both historical data and the simulation outcomes from the FCFS queuing model presented in [4].

First, we made a few assumptions to keep the model relatively simple. More specifically, we assumed a Gaussian distribution for the service times in each priority level, and we considered a weighted random sampling with replacement procedure as the priority assignment policy. (Notice that this effectively turns the service time assignment procedure into a Gaussian mixture model – which is relatively easy to work with.) We also considered 3 different priority levels, and assumed to have 3 non-overlapping operator groups, one for each priority level.

For the optimization procedure we adopted L-BFGS-B, a limited memory version of the Broyden–Fletcher–Goldfarb–Shanno algorithm for bound-constrained optimization [10]. L-BFGS-B represents a good choice for this particular application, because it was designed for large scale optimization and it allows to easily define boundaries on the model parameter subspace to explore.

To help the convergence of the optimization procedure, we also identified a sensible initial value in the model parameter subspace from which to start the optimization process. More specifically, we analyzed the historical data for support group 22 in order to calculate how many incidents transited in the very fast, fast, and slow service categories, and used those values as an estimate of the weights. We then obtained a rough approximation of the mean and variance parameters for the

service time distribution of each priority level by applying maximum likelihood estimation on the sojourn times observed for incidents of the corresponding category. We also assumed to start with 150 operators in each priority level. Finally, we ran the model a few times with Symian and made a few corrections to the initial values.

We then ran the inference procedure, and used the resulting model parameters to reenact support group 22 with Symian. Finally, we analyzed the distribution of sojourn times from the simulation outcome. Figure 6 shows a comparison between the distribution of the historical sojourn times observed in transactional logs and the distribution of sojourn times obtained from simulation. These results demonstrate that multiple-priority queuing models can capture the behavior of support group with a very good accuracy.

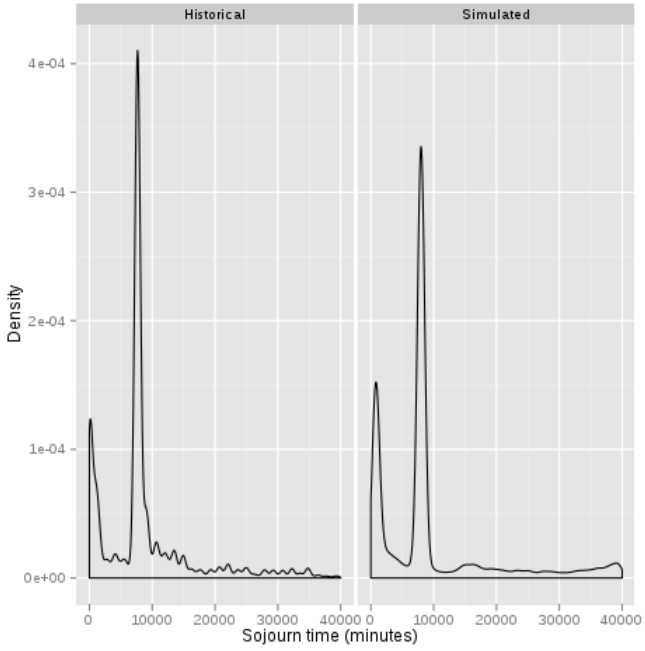

Figure 6. Comparison of distributions of historical and simulated sojourn times at support group 22.

Despite the fact that multiple-priority models seem to be much more appropriate to model IT support groups, the experimental results we collected enable us to formulate a few considerations with regards to the method adopted for model parameter inference.

In our experiments with the FCFS model, the parameter inference method based on Kim and Park's algorithm that we adopted tended to return a low number of fast operators. This effectively enables to realize models that accurately reproduce average-based metrics such as MTTR (Mean Time To incident Resolution) and MICD (Mean Incidents Closed Daily). However, the low number of operators makes incoming incident queues to grow large, thus causing an "inertia" effect that prevents those models to fully capture support group level dynamics in the support group behavior.

Our model parameter inference method for multiple-priority queues does not suffer from this problem. In fact, it returned a much larger number of operators and proved to be capable of capturing the support group dynamics with a much higher accuracy.

On the other hand, our inference procedure for multiple-priority queuing model is significantly more computationally expensive than the one we adopted for FCFS queues. In case of support group 22, each simulation run took around 30 seconds on a laptop PC with a quad-core 2.8GHz Intel Core i7 CPU and 8GB of RAM running a 64-bit version of Arch Linux. This is roughly the same amount of time required to identify the whole set of model parameters for all the 72 groups in our previous model based on FCFS queues. The full amount of time taken for the model parameter inference depends on the size of the parameter subspace to explore as well as the convergence speed of the optimization algorithm adopted. These considerations suggest that FCFS queuing models might still be preferable to multiple-priority queuing models for the system-wide analysis of IT support organization.

In addition, the performance of our inference method is relatively sensitive to the algorithm adopted for the optimization. While the L-BFGS-B algorithm worked pretty well for the model parameter inference of support group 22, it might not lead to optimal results in case of larger parameter spaces. Further experiments with other algorithms are required in order to fully evaluate the robustness of our inference method.

VIII. Related Work

There is an extensive research literature on queuing models for support organizations. For a representative example, see [6] [7].

Most of the previous research has focused on telephone call centers. Brown et al. perform an analysis of real life telephone call center transaction logs, observing non-stationary behaviors in the arrival process similar to what we find in our dataset [11]. However, they focus on the statistical analysis of parameters, such as customer arrival intensity, that are of interest for the optimization of workforce allocation in order to improve quality of service. Instead, we are interested in the development of a model for the reenactment of IT support groups to adopt in what-if scenario analysis.

Only recently have researchers started studying IT support organizations. See for example the works by Wasserkrug et al. [12], focusing on shift scheduling for IT support personnel and reporting on the inadequacy of M/M/N queues to model support groups, and by Zeltyn et al. [13], developing analytic models for multi-server multi-priority queues with applications in IT support organizations. Our work adopts a different approach, as we are interested in the analysis of transactional logs to development a simulative model of real life IT support groups.

IX. Conclusions and Future Work

The multiple-priority queuing model presented in this paper, developed from our experience with the extensive analysis of transactional logs from real life enterprise-class IT support organizations, represents a significant step towards the

realization of accurate tools for the what-if scenario analysis of IT support groups.

By leveraging on our parameter inference method we showed how to build multiple-priority queuing models that can accurately reproduce the behavior of real life IT support groups, capture distinctive traits such as the clustering of sojourn times around a few values.

In future, we are planning to investigate more sophisticated queuing models that consider processor sharing incident servicing with FCFS and/or priority queues. In fact, we observed that batch departures recur very often in the dataset available to us, suggesting that operators work on several incidents at the same time. This is consistent with our observation of how operators usually deal with tickets in real life IT support organizations. As a result, the adoption of multiple-priority processor sharing queues, in which operators work on several tickets at the same time, could help to improve the accuracy of IT support group models even further.

In addition, we are planning to investigate other model parameter inference methods that leverage on knowledge about the inner working of multiple-priority queuing model, and compare their performance with the inference method presented in this paper.

REFERENCES

[1] Office of Government Commerce, "Service Strategy Book", The Stationery Office, Norwich , 2007.

[2] Office of Government Commerce, "Service Operation Book", The Stationery Office, Norwich, 2007.

[3] C. Bartolini, C. Stefanelli, M. Tortonesi, "Modeling IT Support Organizations from Transactional Logs", in *Proceedings of the 12th IEEE/IFIP Network Operations and Management Symposium (NOMS 2010)*, pp. 256-263, 19-23 April 2010, Osaka, Japan.

[4] C. Bartolini, C.Stefanelli, M. Tortonesi, "SYMIAN: Analysis and Performance Improvement of the IT Incident Management Process", *IEEE Transactions on Network and Service Management*, IEEE Communications Society Press, Vol. 7, No. 3, pp. 132-144, September 2010.

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

[6] G. Koole, A. Mandelbaum, "Queueing Models of Call Centers: An Introduction", *Ann. of Oper. Res.*, Vol. 113, No. 1-4, pp. 41-59, July 2002.

[7] G. Koole, "Call Center Mathematics", VU University of Amsterdam, Department of Mathematics Website. http://www.math.vu.nl/~koole/ccmath/. Accessed 3 September 2011.

[8] Y. B. Kim, J. Park, "New Approaches for Inference of Unobservable Queues", in *Proceedings of the 2008 Winter Simulation Conference*.

[9] P. Kvam, B. Vidakovic, "Nonparametric Statistics with Applications to Science and Engineering", Wiley, August 2007.

[10] J. Nocedal, S. Wright, "Numerical Optimization", 2nd Edition, Springer, 2008.

[11] L. Brown, N. Gans, A. Mandelbaum, A. Sakov, H. Shen, S. Zeltyn, L. Zhao, "Statistical Analysis of a Telephone Call Center: A Queueing-Science Perspective", *Journal of the American Statistical Association*, Vol. 100, No. 469, Applications and Case Studies, March 2005.

[12] S. Wasserkrug, S. Taub, S. Zeltyn, D. Gilat, V. Lipets, Z. Feldman, A. Mandelbaum, "Creating operational shift schedules for third-level IT support: challenges, models and case study", *International Journal of Services Operations and Informatics*, Vol. 3, No. 3-4, pp. 242-257, 19 November 2008.

[13] S. Zeltyn, Z. Feldman, S. Wasserkrug, "Waiting and sojourn times in a multi-server queue with mixed priorities", Queueing Systems, Vol. 61, No. 4, pp. 305-328, April 2009.