RC24550 (W0805-038) May 7, 2008
Computer Science

IBM Research Report

Decision Support for Service Transition Management: Enforce Service Transition Management and Change Scheduling by Performing Risk Impact Analysis

Thomas Setzer
Technische Universität München
85748 Garching, Germany

Kamal Bhattacharya, Heiko Ludwig
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598

Decision Support for Service Transition Management

Enforce Service Transition Management and Change Scheduling by Performing Risk Impact Analysis

Thomas Setzer
Technische Universität München
Chair of Internet-based Information Systems
85748 Garching, Germany
thomas.setzer@in.tum.de

Kamal Bhattacharya, Heiko Ludwig
IBM T.J. Watson Research Center
Business-driven IT Management
19 Skyline Dr., Hawthorne, NY-10532, USA
{kamalb, hludwig}@us.ibm.com

Abstract **In IT Service Delivery, alignment of service infrastructures to continuously changing business requirements is a primary cost driver, all the more as most severe service disruptions can be attributed to poor change impact and risk assessment. In nowadays service-oriented business environments, services are shared amongst multiple higher-level or composite services, while the highest composition level finally forms the business processes. Changing services or service definitions in such an environment includes exceptionally high risk and complexity, as various business processes might depend on a service. In this paper we propose a model for analyzing the business impact of operational risks resulting from change related service downtimes of uncertain duration, as the impact on dependent, running or expected business processes is analyzed and transferred into financial losses. The proposed solution automatically considers the dependency chain up to the decomposition mapping of affected business processes. Based on the analytical model, we derive decision models in terms of deterministic and probabilistic mathematical programming formulation allowing for scheduling single or multiple correlated changes efficiently. Preliminary experiments are described to illustrate the efficiency of the proposed models. Using these decisions models, organizations can schedule service specification changes with the lowest expected impact on the business.**

Change Management, Service Transition Management, Change Scheduling, Service-oriented Architectures, Business Impact Analysis, Business-Driven IT Management

I. INTRODUCTION

In recent years, IT service management (ITSM) has received much attention as enterprises understand that operating their IT infrastructure is a large part of their overall operating costs. Today's businesses operate in dynamic environments with the need to continuously adapt to changing customer expectations, market trends, technical enhancements or changes to legislation. These changes entail changes to IT services and business processes to drive alignment of IT with business requirements. According to current surveys uncontrolled changes including flawed risk and impact analysis cause more than 80% of business-critical service disruptions [1].

Publicly available best-practices ITSM frameworks such as the IT infrastructure Library (ITIL) define reference change management processes including several activities like change initiation, where a Request for Change (RFC) describing the required change is submitted, change filtering, priority allocation, categorization, planning, testing, fulfillment and review. Mayor changes must be analyzed and approved, from a technical as well as from a business point of view before they get scheduled [2]. We focus on the impact of changes on the business and on how to schedule changes with minimum associated risks and costs.

As modern IT service infrastructures are continuously transformed towards virtualized resource pools and service-oriented architectures, infrastructural resources are shared among basic or atomic services, which are then shared by a multitude of composite services and so on. The highest composition level finally forms the business processes. In such an environment, it gets increasingly difficult to determine where services associated with specific business process functions are executing, and to analyze the business impact associated with changes to or outages of specific services.

Changing one or several services in such an environment involves exceptionally high risk and complexity, as there is a multitude of interdependencies und uncertainties to manage, and the impact of failures is likely to be business-critical as many business processes might depend on this service. Therefore, efficient and reliable change management aiming at continuous service delivery by automatically considering the dependency chains is essential.

Consider the following example, illustrated in Fig. 1: a business process application provides a 'business process as a service'. Typically, a business process application manages several business processes. The application itself is hosted on one or more physical resources and has dependencies to other applications (or services) and infrastructure components. Estimating the impact of an application failure is – without detailed knowledge of the dependency chains - a fairly manageable problem. The left scenario shows a pictorial of a Tivoli CCMDB discovery [3], where application A is connected to application B. Downtime of Application B means an impact on Application A. Now, the right pictorial shows the same applications but Application A is hosting two processes. The actual dependency between Applications A and B is through process A but not process B. Hence, if Application B fails, process B remains unaffected.

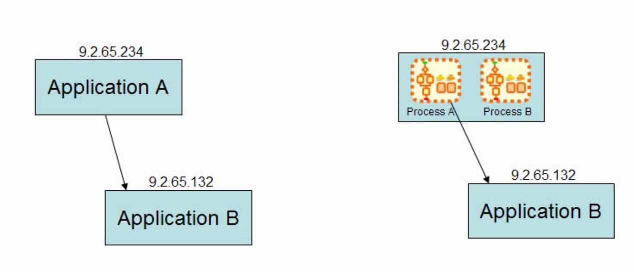

Figure 1. Business Process Application Dependencies

In many cases, a service needs to be taken offline to fulfill a change (e.g. rebooting a server). Usually, the time span required to fulfill a change (and associated service downtime) is of uncertain length. Unexpected failovers might also occur due to changes, with uncertain repair time. The longer a service is unavailable, the more business process instances will be delayed or disrupted. This results in either implicit costs as business suffers, or explicit costs due to service level agreement (SLA) violations and entailed penalties.

How many instances of a particular process are affected highly depends on the business process demand while fulfilling the change. However, the process demand is generally not known beforehand but has to be approximated by means of forecasting techniques.

The focus of this paper is to determine and minimize change related risk in service oriented business environments by introducing decisions models allowing organizations for scheduling service changes with the lowest expected financial loss, or cost. We propose models for analyzing the business impact of change related service downtimes of uncertain length, as the impact on dependent, active business processes is analyzed and transferred into financial losses. The proposed solution automatically considers the dependency chain up to the decomposition mapping of affected business processes. Based on these analytical models, we derive decision models in terms of deterministic and probabilistic mathematical programming formulations allowing for scheduling single or multiple correlated changes efficiently, i.e., with the lowest expected cost. First Experiments and sensitivity analyses are described to illustrate the efficiency of the proposed models.

This outline of this paper is structured as follows: in Section 2 we review related work in this field. In Sections 3 we discuss in detail techniques on how to estimate and quantify operational risks of service transitions. Subsequently, in Section 4 we introduce a basic deterministic decision model and probabilistic extensions to determine efficient change schedules in different business scenarios. We further provide model extensions to take into account change correlations and other sources of risks, like change deadline or change window violation risks. In Section 5, we describe the setup of our preliminary experiments conducted to make first efficiency statements of the models and present our experimental outcomes. Finally, in Section 6, we summarize and conclude.

II. RELATED WORK

In this section we review previous work and guidelines in IT change risk analysis and management related to the work described in this paper.

The definition of risk itself varies broadly according to the specific domain one looks at. The most general definition of risk is 'uncertainty of outcome' [4]. In our case, the outcome is change related cost in a sense of financial loss. To analyze risk impact, i.e., resulting costs for the business, we draw on a two-stage approach; first scanning for possible outcomes and quantifying this outcomes in terms of monetary consequences, and second, weighting these outcomes by their probabilities. This approach is generally known as probabilistic risk analysis as introduced in [5].

In IT change management, most approaches found allow for risk analyses that are not directly transferable to business or financial impact or provide general guidance to change management considering associated risk.

Publicly available best-practices ITSM frameworks and standards such as the IT infrastructure Library (ITIL) [6] or Control Objectives for Information, and related Technology (COBIT) [7] provide guidance on how to perform service management tasks and are validated across a diverse set of environments and situations. As of the importance of managing service changes or transitions efficiently, particular with respect to associated risks, this topic has recently become a mayor focus herein. For example, the Office of Governmental Commerce (OGC) dedicated in the newly published ITIL version 3.0 (May 2007) an own book on how to manage service transitions efficiently, with special regards to associated risks [4]. However, ITIL and related best-practices frameworks provide high-level guidance for performing a service management task like managing a change, but do not provide guidance in how to do the actual change management implementation, e.g., on how to determine and quantify change related risks and costs for a particular business environment.

Some commercial tools and dashboard applications are available that claim to assist in managing changes, although not enough details are available that can be used to evaluate and compare the involved methods [8, 9, 10, 11].

Several papers have presented approaches to qualitatively evaluate risk, for example [12], but do not provide quantitative risk analysis with regard to business impact.

Keller and Hellerstein present the CHAnge Management with Planning and Scheduling (CHAMPS) system to automate steps in the execution of changes. The authors propose decision models to solve different scheduling problems like maximizing the number of changes, minimizing overall downtime, or minimizing the costs associated with change related downtime. The authors assume knowledge of the cost functions for performing a change job at time *t*, while we focus on how to derive cost functions from change related downtime risks to the business processes [13].

Rebouças, Sauvé, Moura, Bartolini and Trastour address the problem of scheduling changes in a way to minimize the financial loss imposed by SLA violations when the implementation of changes is exceeds change deadlines. The authors explicitly consider uncertainty in change durations. [14].

Our work serves to filling the gap in work addressing the formal quantification of service change risk to active and depending business processes, enabling the scheduling of service changes with minimum total expected costs.

III. Service transistions and associated risk on Business processes

The goal of service transition management is to plan and control service changes and deploying changed service releases into the production environment successfully, i.e., with minimum negative impact to the business. We assume that a service is down during the change fulfillment period. As described in Section 1, service transition in service-oriented architectures is coupled with exceptionally high risk and complexity, as there are multiple interdependencies und uncertainties and many business processes might depend on a service. To estimate the risk of services changes to the business (processes), a clear picture and a formal description of the business process and service dependency structure is mandatory.

We will now introduce a notation that is used throughout this paper to formalize process and service dependencies. Let I be the total number of different types of business processes i (i=1, …, I), requested stochastically following a demand distribution or profile D_i. In other words, there are I different business process definitions existing, instantiated on request. A second layer service definition j (j=1, …, J) describes an aggregated or composite service on the layer below the business process layer (i.e., the first layer). Furthermore, an assignment variable u_{ij} indicates that a business process i implements service j in step u_{ij}. Steps of a business process i are enumerated by n_i (n_i=1, … N_j). We set u_{ij}=0 if a business process i definition does not embody service j. In the same manner we model the dependencies of lower-level services. We enumerate the service descriptions on the next lower aggregation level by k (k=1, …, K) and assign these third-level services by setting u_{jk} correspondingly to the step n_j (n_j=1, … N_j) in the j service flow definition. Likewise, we set u_{jk}=0 if k is not implemented by j. Fig. 2 illustrates the resulting dependency structure.

Using this dependency model, one can automatically derive which higher-lever services and business processes are affected by a specific service downtime.

However, to estimate the business impact of a change, additional information is required, like how many instances of business processes are affected, and how many service level agreements of these processes are expected to be violated.

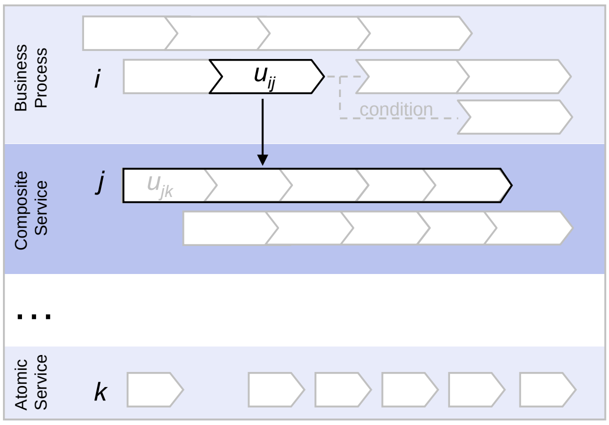

Figure 2. Three Layer Service Dependency Model

The amount of affected business process instances clearly depends on the business process demand at and before the time a change is fulfilled. Business forecasting techniques are the means of choice to estimate the demand for a certain business process during a particular period of time. With D_i as a business process i's demand distribution profile (i.e., the demand distributions profile of all considered time slots t, D_{it}, demand forecasts d_{it} are possible for a certain time slot t (for example by setting d_{it} to D_{it}'s mean value).

To keep our decision model computable, we divide time into small discrete time slots, wherein we assume demand of a fixed level. The costs of business process disruptions or delays are usually negotiated beforehand and defined in SLAs. An SLA typically includes a process' maximum response or execution time L_i and the definition of (monetary) penalties p_i to pay on SLA violations. Depending on a SLA, penalties are to paid per maximum response time violation, if the number of service level violations during a certain time span exceeds a defined threshold value, or other individual agreements. Simply multiplying the number of process instances expected during the duration of a change with the penalties would overestimate change related costs, as not all running business process instances will be disrupted or delayed. For example, business process instances which already passed the step implementing the service that is going to be changed will not be affected at all, nor is there an impact on running processes instances which will execute the changed service after the change is fulfilled and the service is available again. Furthermore, business processes and services might be queued. If the time buffer, i.e., the difference between the maximum execution time and the normal or usual execution time is large enough, there is a chance to still execute affected processes instances in a SLA compliant way.

In the following, a process is described to estimate the amount of SLA violations if queuing is not possible. Afterwards we look at infrastructures where queuing processes and services is a valid option. Firstly, we presume perfect knowledge of future aggregated business process demand per time slot and change related downtime. Later in this section, we take into account the uncertainty in demand and service downtime.

A. SLA violations without queuing

Consider a request for change (RFC) for service j, where j will be unavailable for a duration Δt_j^{down} after the start time t_j of the change. The task is to estimate $d_{ijt}^{penalty}$, the number of SLA violations of depending business process instances. Given this number for each affected business process, the estimated costs of changing j in t, c_{jt} are

$$c_{jt} = \sum_i p_i d_{ijt}^{\,p} \tag{1}$$

To predict $d_{ijt}^{\,p}$ we proceed as follows: all service instances executing j during time period $[t_j;\ t_j+\Delta t_j^{down}]$) are disrupted. From a planning perspective, we assume equal arrival rates of business process request (principle of indifference) as there is only aggregated knowledge of service demand per time slot available. This assumption is tight as long as the forecasting time periods are kept small. Of interest is the demand for a business process i not only during the change downtime Δt_j^{down} but also before t_j as running process instances, starting before t_j might reach j during $[t_j,\ t_j+\Delta t_j^{down}]$. Depending on the step in which a business process i implements service j, business process instances starting after t_j-L_i might be affected if $j$fis executed in the last process step ($u_{ij}=N_i$). If j is executed in the next to last step ($u_{ij}= N_i$-1), only process instances starting after $t_j - L_i + L_{N(i)}$ are affected, etc. On the other side, if i implements j in step N_i and the total execution duration of preceding process steps exceeds j's downtime, instances starting during $[t_j,\ t_j+\Delta t_j^{down}]$ are not impacted by the current change. To approximate the demand for a business processes i which might have j execution overlapping with $[t_j,\ t_j+\Delta t_j^{down}]$, $d_{ijt}^{\,p}$, we consider therefore business processes demand during

$$[t_j - L_i + \sum_{j'} L_j';\ t_j + \Delta t_j^{down} - \sum_{j''} L_j''] \tag{2}$$

where j' is a service executed in a process i steps preceding j's implementation step and j'' is a service executed in steps after j's implementation step.

An alternative way to approximate $d_{ijt}^{\,p}$, with no further knowledge of the concrete step a process i implements a service j is described in the following: assuming an equal demand distribution around t_j, the percentage of i business process instances executing j during in $[t_j,\ t_j+\Delta t_j^{down}]$ is (on average)

$$\frac{L_j}{L_i} \tag{3}$$

where L_j is the execution duration of j, and L_i is the overall process execution duration. The probability that a running process instance (executing a step preceding u_{ij}) will reach j in $[t_j,\ t_j+\Delta t_j^{down}]$ is

$$\frac{\Delta t_j^{down}}{L_i} \tag{4}$$

Therewith, the expected total costs of SLA violations caused by changing j in t_j are

$$c_{jt} = \sum_{i:u_{ij}>0} \left(\left(\frac{\Delta t_j^{down} + L_j}{L_i}\right) d_{ijt}\ \Delta t_j^{down}\right) p_i \tag{5}$$

B. SLA violations with queuing

We will now look at the estimated costs of changing j in time slot t if queuing (or buffering) is allowed. Here, not all business process instances executing j overlapping with $[t_j,\ t_j+\Delta t_j^{down}]$ are disrupted as instances can re-execute j after the change is fulfilled. If an SLA is violated depends on a process' time buffer b_i ($b_i = L_{i,max} - L_i$), where $L_{i,max}$ is the maximum execution time of a process, and L_i is the normal or usual execution time of a process. Again, the probability of a process instance currently executing j is shown in (3). If $b_i \le \Delta t_j^{down}$, all considered process i instances will exceed the maximum response time. If $b_i > \Delta t_j^{down} + L_j$, no service instance is disrupted. If $\Delta t_j^{down} < b_i < \Delta t_j^{down} + L_j$, there is a change of a rollback and re-execution without SLA violation if the time buffer exceeds the amount of time already spend executing j before t_j plus j's downtime Δt_j^{down} . This probability is shown in (5)

$$\left(\frac{L_j}{L_i}\right)\left(1 - \frac{b_i}{L_i}\right) \tag{6}$$

The probability that a running process instance (executing preceding steps) will reach j in $[t_j,\ t_j+\Delta t_j^{down}]$ is shown in (4). If $b_i > \Delta t_j^{down}$, all services are delivered successfully. *If* $b_i < \Delta t_j^{down}$, the average rate of successful delivered business process instances is:

$$\left(\frac{\Delta t_j^{down}}{L_i}\right)\left(\frac{b_i}{\Delta t_j^{down}}\right) \tag{7}$$

C. Non-Linear Business Processes and Service Flows

The estimations of expected change related penalties as introduced in the previous section assume linear business processes and service flows with a predetermined sequence of service executions. In practice, business processes might take different branches or service flow paths based on certain conditions, where one branch might include a service to be changed while others do not. Hence, business process forecasting ignoring such conditional branches overestimates the number of SLA violations and costs. A finer-grained demand forecast is required for each possible branch. This forecast can be derived by analyzing the history of the different executed branches in the same way the total demand for linear processes is derived by business forecasting methods. We model each branch as own business process as shown in Fig. 3.

Using this statistical means, one can model forked business processes. Processes including iterative sequences like loops can be demodulated in the same manner, by defining each possible flow as an own process and by assigning probabilities derived from statistical analyses of log data.

IV. CHANGE SCHEDULING DECISION MODELS

We will first introduce a basic change scheduling decision model for shared services underlying a number of restrictive assumptions like perfect knowledge of business process demand per time slot and deterministic change related downtimes of services. Afterwards, we will propose model variants considering uncertainty in business process demand and stochastic service downtime. Based on these model formulations, a couple of extensions are introduced to consider other types of operational risks and costs associated with service transitions and we will address the problem of handling correlated changes.

A. Basic Deterministic Model

We will now introduce a deterministic mathematical programming model (DMP) to solve the problem of finding the schedule for a set of uncorrelated changes J_{RFC} with minimum overall service level violation costs in environment without queuing. Business process demand per time slot t, d_{it}, the downtime of a service after the change start time, Δt_j^{down}, and execution durations of services, L_j, and whole business processes L_i, are approximated by using their mean values. A penalty is paid per SLA violation.

We introduce a binary decision variable $x_{j,t} \in \{0,1\}$ indicating whether j's change is started in t_j or not.

The objective functions to minimize the total sum of penalties resulting from changes in service infrastructures without queuing is

$$\min \sum_{j \in J_{RFC}} \sum_{i:u_{ij}>0} \sum_{t} \left(\left(\frac{\Delta t_j^{down}+L_j}{L_i}\right) d_{ijt}\, \Delta t_j^{down}\right) p_i x_{j,t} \qquad (8)$$

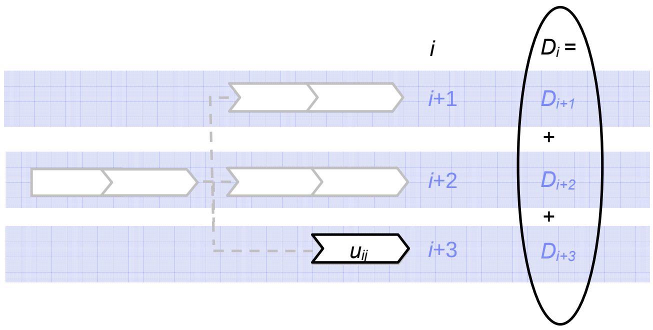

Figure 3. Non-Linear Business Processes and Service Flows

We set the beginning of our change planning period to t=0 and assume to obtain J_{RFC} before t=0 (Note that in practice, changes will be requested on a continuous time base rather than bundled. The usual way to proceed is to re-calculate the optimization problem each time a new RFC is submitted. More advanced methods might forecast aggregated RFC 'demand' if changes are submitted in regular sequences). As we divided time into discrete time slots, time related parameters are positive integer variables (t_j, Δt_j^{down}, b_i, L_i, $L_j \in Z_0^+$) and penalties and demand parameter are positive real values (d_{it}, $p_i \in R_0^+$).

As further constraints we introduce change related deadlines t_j^d. Depending on the severity of a change, there is generally a priority associated with a change, defining a deadline when a change needs to be implemented. This constraint can be formulated as

$$\sum_{t_j+\Delta t_j^{down}<t_j^d} x_{j,t} = 1,\ \forall j \in J_{RFC} \qquad (9)$$

Note that a change deadline is originally defined as a period Δt_j^d after t_j^{RFC}, the time the RFC for j arrives. As we define $t_{j,}^{RFC} = 0$, setting the deadline to t_j^d instead of $t_j^{RFC} + \Delta t_j^d$ suffices in our case.

B. Stochastic Change Scheduling Model

So far, we used deterministic approximations for expected demand, service downtime and service execution durations.

One should expect that ignoring the probabilistic nature of demand, downtime and execution time has a negative impact on the decision making. Suppose a service j change, and a depending business process i with extremely high penalties to pay on service level violations. The average change related downtime of j is 10 but varies broadly, and the decision is either to start the change in t=0 or in t=50. The demand for i is expected to be slightly lower during t=0 – 9 than during t=50 – 59 but increases rapidly from t=10 on, while demand is expected to be of constant level after t=59. The deterministic model would certainly select t=0 while a stochastic model explicitly taking into account uncertainty of downtime would select t=50, which would clearly be the better decision.

However, putting too much stochastic information into a decision model makes it – at least for medium and large problem sizes – intractable due to the large number of resulting decision variables and limits therefore its practical applicability.

Therefore, we draw on a stochastic programming formulation with simple recourse as introduced by Birge and Louveaux to consider the stochastic nature of the variables while keeping the model computable [15, 16].

This is illustrated using a change related downtime probability distribution as shown in Fig. XZ (grey line). We separate the distribution into N sequential discrete sections n (n =1, …, N). The cumulated probability (integral) of a section is then interpreted as the downtime probability of a time slot in the section, while we suppose the downtime can only take these discrete downtime values: $\Delta t_j^{down} \in \{\Delta t_{j,1}^{down}\ \Delta t_{j,2}^{down},\ \ldots,\ \Delta t_{j,N}^{down}\}$. The resulting objective function can be formulated as

$$\min \sum_{j \in J_{RFC}} \sum_{i: u_{ij}>0} \sum_{t} \sum_{n=1}^{N} P(\Delta t_{j,n}^{down}) \left(\frac{\Delta t_{j,n}^{down} + L_j}{L_i}\right) d_{ijt}\, \Delta t_{j,n}^{down}\, p_i x_{j,t} \tag{10}$$

The right part of the objective function computes the costs that would be resulted if the downtime would have been exactly $\Delta t_{j,n}^{down}$; the term on the left is a correction for the uncertainty in downtime (a weight).

Likewise, we model the other stochastic variables like business process demand during a time slot or the execution time of a service. Note that usually the parameters or even the type of distributions will depend on which time slot you consider.

C. Change Fulfillment Deadlines and Waiting Costs

As already mentioned, a change needs to be fulfilled in a maximum change fulfillment time Δt_j^d, after a change request is submitted. As discussed previously in this paper, the urgency depends on the priority of a change. In the basic deterministic model formulation we assumed that this deadline is mandatory.

Considering the uncertainty in the time needed to perform the service change (we assume the service to be down during change activities) it can no longer be guaranteed to fulfill a change before the agreed change deadline; only a probability can be assigned to fulfilling the change in time. Therefore, the restriction that a change needs to be fulfilled before t_j^d of the change deadline needs to be relaxed to

$$\sum_{t} x_{j,t} = 1,\ \forall j \in J_{RFC} \tag{11}$$

Exceeding a change deadline might entail a predefined penalty and extra payments for each additional time slot needed to fulfill the change. The later a change is started, the higher the expected costs of a deadline violation will be, since the probability of completing change implementation before the deadline will decrease continuously.

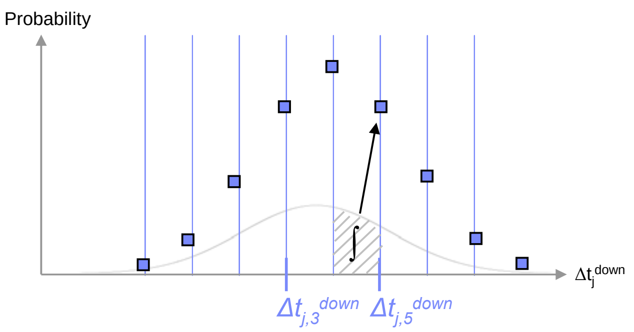

Figure 4. Probilistic Modelling with Simple Recourse

Let the fixed penalty on change deadline violation be α, and the additional costs per time slot a deadline is exceeding be β. Herewith, the expected overall deadline violation cost function which need to be added to the objective function as formulated in our decision model is

$$\min \sum_{t} \left(\alpha(t_j + \Delta t_j^{down} - t_j^{d}) > 0 \right) + \beta(\max(0, t_j + \Delta t_j^{down} - t_j^{d}))x_{jt} \tag{12}$$

Note that for reasons of brevity we provide formulas with only the service downtime modeled stochastically while other random variables are approximated by their mean values.

Furthermore, the moment an RFC is submitted, there may already be a need felt for the change to be implemented as the business may suffer until the change has been successfully fulfilled; for example, this may be due to a service being unavailable as would happen if the change request was initiated as a result of an incident, or there may be other negative impact causes, like lost opportunities such as would occur for a change meant to bring up a new required service. With γ as the implicit costs of waiting one more timeslot for a change to be fulfilled, the total waiting costs can be formulated as

$$\sum_{t} \gamma(t_j + \Delta t_j^{down}) x_{j,t} \tag{13}$$

D. Allowed Change Windows

Furthermore, the fulfillment time of a change might be restricted to a number of allowed change window time slots, e.g. at weekends or during nights. Violating a change window restriction might have serious impact on the business, as that would mean a service is down in times this service is frequently required. Therefore, penalties might result from exceeding a change window l (l=1, …, L). Let T_{cj} ($T_{cj} = \{t_{cj1}^{start}, \ldots, t_{cj1}^{end}\}, \ldots \{t_{cjL}^{start}, \ldots, t_{cjL}^{end}\}$ be the set of allowed change windows. As change related downtime might be of uncertain length, there is an increasing risk of violating the change window constraints the later a change is started. With δ as the costs per time slot a change window is exceeded, and the restriction that a change has to start (at least) inside a change window ($t_j \in T_{cj}$), the part that has to be added to the objective function as formulated in our decision model is

$$\min \sum_{t} \max(0, \delta(t_j + \Delta t_j^{down} - \min(t_{jl}^{end} : t_{jl}^{end} > t_j)))\ x_{j,t} \tag{14}$$

E. Correlated Changes

The basic model formulation handles multiple independent changes. To schedule changes in a mandatory order, a constraint for each dependency has to be added to the decision model formulation. Firstly, changes might need to be started in a certain sequence ($t_j < t_{j+1} < t_{j+2} < \ldots$) or a change must be fulfilled before the next change may get scheduled ($t_j + \Delta t_j^{down} < t_{j+1} + \Delta t_{j+1}^{down} < \ldots$). The constraints in our mathematical model formulation are therefore $x_{it} < x_{(j+1)t} < t_{(j+2)t}$, or $x_{jt} + \Delta t_j^{down} < x_{(j+1)t} + \Delta t_{j+1}^{down} < t_{j+2}$, respectively.

Besides mandatory change scheduling orders, changes might be correlated for example in terms of a reduction of aggregated downtime when executing changes together (imagine two changes to a server operating system, both requiring a reboot. The overall change duration might be reduced by applying these changes together, but this may result in higher risk in terms of higher downtime variance (incompatibilities, etc.).

While arbitrary statistical values can be chosen, in our example we focus on mean (M) and variance (V) deviation. Therefore, we consider two changes to j and $j+1$ are correlated if either

$M(\Delta t_j^{down}(t)+\Delta t_{j+1}^{down}(t)) \neq M(\Delta t_j^{down}(t)+\Delta t_{j+1}^{down}(t+\Delta t))$ and/or

$V(\Delta t_j^{down}(t)+\Delta t_{j+1}^{down}(t)) \neq V(\Delta t_j^{down}(t)+\Delta t_{j+1}^{down}(t+\Delta t))$

We treat each change item combination with significant deviant aggregated statistical mean and/or variance values as one single change. The decision to make is to either schedule all included single changes separately or to schedule the novel 'aggregated' change. This XOR constraint can be formulated as follows (if the question is to either change j and $j+1$ separately, or, alternatively the aggregated change $(j, j+1)$)

$$\sum_t x_{j,t} + x_{(j+1),t} + 2x_{(j,j+1),t} = 2 \qquad (15)$$

Furthermore, the change deadline for $(j, j+1)$ is set to min $(t_{j,RFC} + \Delta t_j^d, t_{j+1,RFC} + \Delta t_{j+1}^d)$.

F. Change Re-Scheduling

The decision model selects the time slot with the lowest expected overall costs based on business process demand forecasting. However, when approaching to t_j, further knowledge is available of process demand and process states (progress). This knowledge can be used to reschedule the change start time t_j. For example, if in (t_j-1) more business process instances are running than expected, or a higher percentage of running instances is currently executing service j, there is a decision to make on whether to retain t_j or to wait several timeslots where. However, increasing delay costs and a higher probability to violate change window restrictions have to be taken into account when making such a decision. Note that demand forecasting for processes might be adapted by using short term prognoses if current demand differs significantly from demand expected beforehand. Furthermore, business process request arrivals might be modeled as Poison Process to consider the uncertainty regarding the exact arrival rates and arrival times, with $P_{\lambda(i)}(r=k)$ as the probability of k incoming service i requests in t. As we did with downtime uncertainty, we model the impact of different possible arrival rates weighted by their probabilities.

V. Experimental Analysis

In this section, we analyze and discuss the efficiency of the scheduling models proposed in this paper. In our preliminary experimental evaluations we compared variants of our models to the optimal solutions (be scanning the total solution space), with total change related costs under different service infrastructures, demand scenarios, and downtime distributions used as a benchmark. Firstly, the experimental set-up that we used for our experiments is described. Secondly, we report the results of our experiments and discuss their outcome.

A. Experimental Set-Up

We analyzed 12 different service infrastructure scenarios under different business process demand profiles. The durations of each experiment was set to 300 time slots t (t= 0, …, 299). The change deadline was set to t_j^d=275 with fixed costs of $20 for violating this restriction and additional $2 per exceeded times slot. In our first evaluations, change windows, and waiting costs were not considered. To allow for sensitivity analysis how variations in the output of our models can be apportioned to variations of j's downtime distribution, we repeated each experiment until our results were significant (experimental item, average all outcomes) for each downtime distribution. We analyzed 8 different downtime distributions with increasing variance. To configure and automate our experiments and to analyze our experimental outcomes a simulation tool has been developed (see Fig. 5). The figure shows a visualization of an example service infrastructure scenario used in our experiments with two business processes, a linear and a forked process.

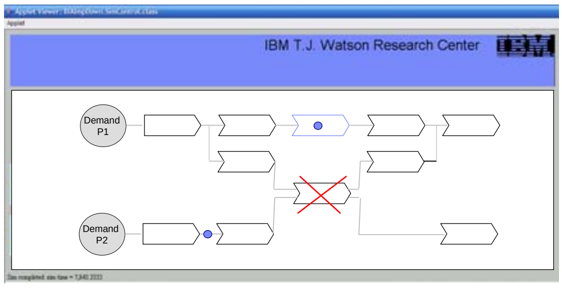

Figure 5. Example Business Process Demand Scenario

An example business process demand scenario is shown in Fig. 6. The graph shows the mean demand level M per time slot. We adapted the demand level after each time slot to generate a demand profile following these curves. During a time slot, we generated demand following a (M, 0.20M) normal probability distribution (uniformly distributed).

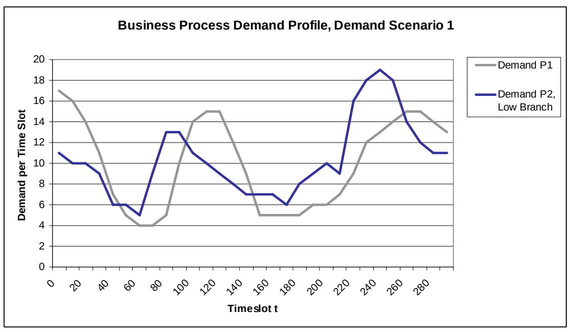

Figure 6. Analyzed Scenario

B. Experimental Results

Experimental results show that the probabilistic decision model with a simple resource of the service downtime distributions (applying the objective function as shown in equation (10)) found the optimal solution for all experimental items. In experiments with low service downtime variance (less than 15% of the mean downtime duration), the deterministic model selected the change start time slot with minimum costs. Except one demand scenario with almost flat process demand levels, the deterministic never found the optimal solution in scenarios with one of the two highest downtime variances. Fig. 7 presents the cost savings by using either the deterministic or the probabilistic scheduling model. The bars show the change related costs when using one of the two decision model variants relative to the average costs over all scenarios (with a certain downtime variance level) when the change start time was selected randomly.

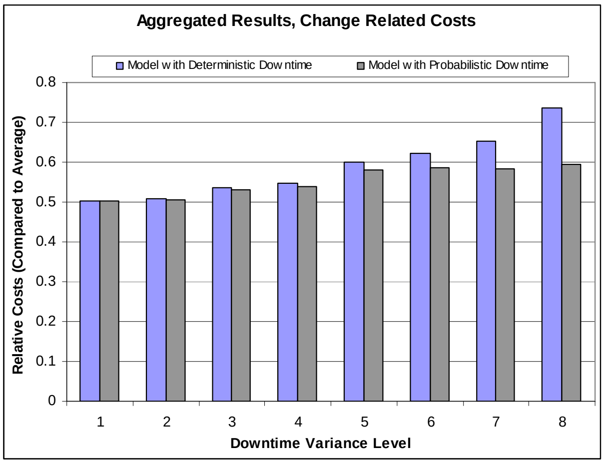

Figure 7. Aggregated Experimental Results

VI. CONCLUSION AND OUTLOOK

The contribution of this work is the introduction and first experimental evaluation of models for analyzing the business impact of service changes in service-oriented architectures. We analyzed change related operational risks on active business processes and techniques to transfer these risks into financial metrics, or costs.

As far as we know, no previous work exists that formally quantifies the risk of changing services in SOA environments to the business (processes), or that derives decision models which allow organizations to schedule service changes with minimum total expected costs.

In our experimental analyses we evaluated the efficiency of our models compared to the optimal and average solution, with total change related costs under different demand scenarios and downtime distributions used as a benchmark. We conducted preliminary numerical experiments with various business process demand scenarios and different downtime distributions and made initial efficiency statements. Experimental results show that the proposed probabilistic model derived the optimal solution in all of our experiments.

Future working plans are more exhaustive sets of experiments with different, possibly real-world, business scenarios. We intend to additionally explore the impact of rescheduling change times when approaching the planned change start time, the impact of uncertain service execution durations, and the impact of latency and change window violation costs. We also plan to test the models in the field as a decision support tool for change scheduling in selected businesses. Note that, when applying the models to certain businesses, the individual context has to be considered and adequate parameters and costs functions have to be determined.

REFERENCES

[1] CRM Today: Change Control Management, URL: http://www.crm2day.com/news/crm/118482.php

[2] Office of Government Commerce: ITIL Service Support. The Stationery Office, London, 2000

[3] IBM Tivoli Change and Configuration Management Database, URL: www.ibm.com/software/tivoli/products/ccmdb/

[4] Office of Government Commerce: Management of Risk: Guidance for Practioneers. The Stationery Office, London, 2007

[5] Kaplan, S. and Garrick, B.J., “On the Quantitative Definition of Risk” Risk Analysis, Vol. 1, 1981

[6] OGC’s Official ITIL Website, URL: http://www.best-management-practice.com/IT-Service-Management-ITIL

[7] IT Governance Institute, “Cobit 3rd Edition”, 2000, www.isaca.org/cobit.htm

[8] IT Service Management - Change and Configuration Management: Reducing Risk by understanding your infrastructure, URL: http://www-03.ibm.com/solutions/itsolutions/doc/content/bin/itsol_it_service_management_change_and_configuration_management.pdf

[9] Cisco IT Balances Innovation and Risk With Change Management Process,URL:http://www.cisco.com/web/about/ciscoitatwork/case_ \ studies/business_management_dl2.html

[10] BMC Remedy Change Management Dashboard, www.bmc.com/products/documents/25/43/62543/62543.pdf

[11] HP Mercury Change Control Management, URL: http://www.mercury.com/us/products/change-control-management/

[12] Goolsbey J., “Risk-Based IT Change Management”, URL: http://web.reed.edu/nwacc/programs/awards/excellence_award/pnnl_submissions_07/pnnl_risk-based_it_change_management.pdf

[13] Keller, A., Hellerstein, J.L., Wolf, J.L., Wu, K.-L., Krishnan, V., "The CHAMPS system: change management with planning and scheduling", in: Network Operations and Management Symposium, 2004

[14] Rebouças, R., Sauvé J., Moura A., Bartolini C., Trastour D., “A Decision Support Tool for Optimizing Scheduling of IT Changes”, 10th IFIP/IEEE Symposium on Integrated Management, 2007

[15] S. V. de Boer, R. Freling, N. Piersma, “Stochastic Programming for Multiple-Leg Network Revenue Management” Report EI-9935/A , ORTEC Consultants, Gouda, Netherlands, 1999

[16] J.R. Birge, F. Louveaux, “Introduction to Stochastic Programming,” Springer Series in Operations Research, 1997