CHANGEADVISOR: A Solution to Support Alignment of IT Change Design with Business Objectives/Constraints

Roben Castagna Lunardi[1], Weverton Luis da Costa Cordeiro[1], Fabrício Girardi Andreis[1], Juliano Araujo Wickboldt[1], Cristiano Bonato Both[1], Luciano Paschoal Gaspary[1], Lisandro Zambenedetti Granville[1], David Trastour[2], and Claudio Bartolini[3]

[1] Institute of Informatics, Federal University of Rio Grande do Sul, Brazil
[2] HP Laboratories Bristol, UK
[3] HP Laboratories Palo Alto, USA
{rclunardi,weverton.cordeiro,fgandreis,jwickboldt, cbboth,paschoal,granville}@inf.ufrgs.br, {david.trastour,claudio.bartolini}@hp.com

Abstract. Change planning represents a key element for the operation and management of Information Technology infrastructures and services. Its scope ranges from the high level design of a change request to the generation, either manually or automatically, of detailed plans that, if executed, will perform the requested changes (*e.g.*, modification of network device settings and deployment of new services). A fundamental problem is that, although correct, such detailed plans may not be necessarily aligned with the requirements defined in the business level (*e.g.*, minimization of the downtime of a given service). To overcome this problem, in this paper we propose a solution for the alignment of change plans with business objectives/constraints. The solution is analyzed experimentally through a prototypical implementation of a decision support system called CHANGEADVISOR, which helps operators to understand the trade-offs between alternative change designs.

Keywords: IT service management, business-driven IT management, IT change management.

1 Introduction

The increasing size and complexity of Information Technology (IT) infrastructures and services have demanded the adoption, by organizations, of best practices and processes in order to ensure their correct and efficient operation. Aiming at supporting organizations in such a challenging task, the Information Technology Infrastructure Library (ITIL) [1], an important reference in this context, recommends, among other processes, *change management*. This process helps to efficiently handle the necessary changes in hardware and software within organizations.

The scope of *change management* ranges from the change specification, by an initiator in a high level of abstraction, to the generation of detailed plans – those that comprehend low level activities such as modification of network device settings and

C. Bartolini and L.P. Gaspary (Eds.): DSOM 2009, LNCS 5841, pp. 138–151, 2009.

deployment of new services. These plans, once executed, tend to accomplish the changes requested by an initiator in the managed infrastructure.

One of the main problems considering the generation of change plans is that, from the same high level specification, different detailed plans can be obtained. Although correct, they may not be necessarily aligned with the purposes defined for the IT infrastructure in the business level. The purposes can either have a technical nature, such as imposing constraints regarding the unavailability of provided services by the managed IT infrastructure, or define objectives for optimizing resource allocation during the change deployment (*e.g.*, minimizing the number of human resources involved in an e-mail service maintenance procedure). When these purposes are not considered, the generated plans, if executed, may lead to results that violate policies predefined in the business level (*e.g.*, e-mail service maintenance downtime must not exceed 10 hours monthly).

We highlight three important reasons for aligning IT change plans with business purposes (also referred to as business objectives/constraints in this paper). First, there is the possibility of optimizing available resources that are usually limited and expensive. Second, the alignment has the potential of reducing costs such as the time spent to deploy changes. And finally, perhaps the most important reason, aligning change plans with distinct purposes allows the operator to understand the resulting trade-offs when one chooses a change plan among various possibilities. Despite the potential benefits, the alignment of change plans with business objectives/constraints has been neglected in investigations carried out in the area of IT service operations and management. The generation of executable [2] and optimized plans [3] are examples of issues that, alternatively, have been addressed recently.

To tackle the aforementioned problem, in this paper we propose a solution for the alignment of IT infrastructure and service change plans with business objectives/constraints. In contrast to previous investigations conducted in the area, this paper focuses on the automated generation of change plans guided by business objectives/constraints. The proposed solution is experimentally evaluated through CHANGEADVISOR, a prototypical implementation of a decision support system that provides means for the understanding of trade-offs among alternative change designs. During the evaluation, which was performed considering scenarios based on real cases, we analyzed correction and completeness of generated plans, as well as their alignment with business purposes.

The reminder of this paper is organized as follows. In Section 2 we review some key investigations in the area of IT change management. In Section 3 a conceptual solution for the generation of change plans aligned with business purposes is presented. We detail, in Section 4, the CHANGEADVISOR system and present, in Section 5, the results achieved using the system. Finally, we conclude the paper conveying final considerations and perspectives for future work in Section 6.

2 Related Work

The IT service management area has grabbed the attention of the scientific community in recent years. Several aspects, such as models [4], automation [5], and alignment with business purposes [6] have been explored. Specifically analyzing the subarea of

change management, it is possible to observe similar activity. For example, important investigations about scheduling [7] and rollback [8] can be cited. Nevertheless, the topic of IT change planning, which is the focus of this paper, has not been sufficiently explored, as underscored in the following two paragraphs.

Keller *et al.* [3] have proposed CHAMPS, a system for the automated generation of change plans, which explores a high degree of parallelism in task execution. Although it has proved to be able to improve task scheduling considering available hardware resources, neither the activity association to human resources nor the objectives/constraints (specified by the change initiator) have been taken into account.

In a recent paper, Cordeiro *et al.* [2] have proposed a solution to *formalize*, *preserve*, and *reuse* knowledge acquired with frequent IT changes. The solution, which is based on *change templates*, allows recurrent activities (*e.g.*, service change or implementation) to be specified and (recurrently) reused. Although this solution comprises an algorithm to generate detailed plans from high level specifications, aligning generated plans with business purposes is out of its scope. In a subsequent work [9], the mentioned algorithm has been improved to consider technical constraints imposed by the managed environment (*e.g.*, availability of resources such as disk and memory space). However, business purposes have not been considered.

Other investigations, although not designed to deal with change management, have explored alignment with the business level. One of them is Keller's [10], which introduced the concept of electronic contracts. In that research, the authors have proposed four kinds of contracts: RFCs, Deployment Descriptors, Policies and Best Practices, and Service Level Agreements (SLAs). Such contracts have different purposes, formats, and level of details. They are used for objective/constraint specification and are of great importance for the change deployment process.

Summing up, although change management has been a focus of recent investigations, they have not considered alignment of changes with specific purposes in the business level. It means that the planning and subsequent change deployment may potentially be conducted in an undesired direction, *i.e.*, may not be aligned with purposes demanded by the change initiator. In order to deal with this problem, the following sections present not only a conceptual solution, but also a system developed to support change alignment with business purposes and its evaluation.

3 Conceptual Solution

Aiming to support the generation of change plans aligned with business purposes, two new key components have been introduced to the conceptual solution for change design and planning proposed in a previous work [9]. The new components – highlighted in gray – materialize the mechanism for the planning of changes guided by business purposes and fit adequately into the previously proposed solution, without significant modifications. Fig. 1 presents an overview of the extended solution, focusing on its main components, involved actors, and their interactions.

The initiator starts a change process interacting with the component *Change Designer* (flow 1 in Fig. 1) to specify a *Request for Change* (RFC). RFCs describe what changes must be done in the managed infrastructure, the *Configuration Items* (CIs) primarily affected (network devices, computers, services, applications, etc.) and the

business objectives to be met. However, the RFC does not specify details on how the change should be executed; such details have to be specified in the following step by the design of a preliminary plan, supported by the component *Change Designer* (flow 2). This plan consists in a workflow of actions that describe, in a high level of abstraction, how the requested change should be performed in the IT infrastructure.

The generation of a detailed change plan based on the preliminary specification is then performed, without human intervention, by the component *Change Refiner* (3). The automated processing, carried out by a refinement algorithm, is based on both (*i*) information about software packages, available in the *Definitive Media Library* (DML) (4), and (*ii*) information about the target IT infrastructure, found in the *Configuration Management System* (CMS) (5).

After the change refinement, the resulting plan is forwarded to the component *Change Aligner* (6), which plays a central role in the alignment of the plan with the business objectives/constraints expressed in the RFC. In order to guide this alignment, the *Change Aligner* retrieves from the CMS (7) information about costs and skills of available human resources (*e.g.*, an operator who is an expert in e-mail service and whose work hour costs 40 monetary units). It also gathers information about the capabilities of the hardware assets affected by the plan (*e.g.*, computational power in the case of workstations and servers). It is important to highlight that the process of change refinement and alignment is reiterated (8) during a period of time predetermined by the operator, aiming to generate distinct detailed plans aligned to the same business purposes determined in the RFC. This is due to the fact that the problem is NP-hard, and the algorithms employed to solve this problem are heuristic-based, *i.e.*, the techniques used take advantage of information on past experiences in order to obtain results closer to the optimal (best) solution.

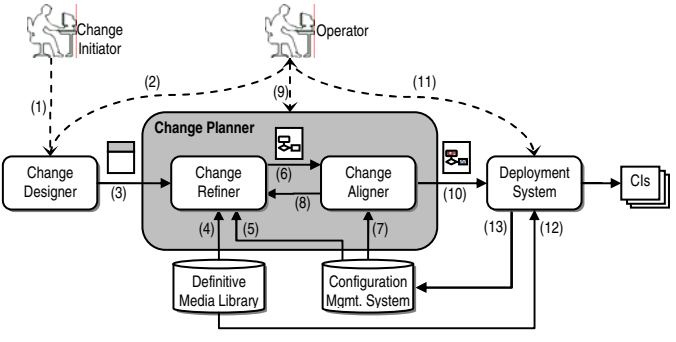

Fig. 1. Conceptual architecture of the proposed solution

In a later moment (9), the generated plans may be modified by the operator as to reflect his/her needs, and evaluated in terms of trade-offs between the various options. Among the criteria that may guide such an analysis, we cite: expected time and/or financial costs to perform the changes and human resources required. Based on this analysis, the operator will be able to select an appropriate plan to be deployed.

Finally, in the last step of the change process, the operator invokes the deployment of the selected plan (10), interacting with the component *Deployment System* (11).

This component may consume software packages available in the *Definitive Media Library* (12) in order to be able to execute some of the activities described in the change plan. After deploying this plan, the *Deployment System* updates the CMS, adding information about modifications in each of the affected CIs (13).

It is important to mention that the conceptual solution depicted in Fig. 1 is targeted at the *design*, *planning*, and *deployment* of changes. Therefore, other phases that comprise the traditional change management process – such as the *evaluation* of the requested changes by the *Change Advisory Board* (CAB) and the *schedule* of changes into maintenance windows – are envisaged as directions and trends for long-term, future investigation in this area.

Having presented an overview of our solution, in the following subsections we describe (*i*) the generation of detailed plans based on preliminary specifications and (*ii*) the process of aligning plans with purposes determined in the business level.

3.1 Change Refiner

The component *Change Refiner* is fundamental in the process of generating alternative detailed change plans based on a preliminary specification. The core of this component is a refinement algorithm, inspired in a previous work [9]. As previously mentioned, the generation of detailed change plans requires not only information about the CIs (*e.g.*, hardware, software, and people) that compose the IT infrastructure and their relations, but also about the software packages available for the change deployment. Aiming to represent this information in the CMS and in the DML, respectively, a subgroup of the *Common Information Model* (CIM) is used in this paper [11]. Fig. 2 presents a partial view of the model. Relationships such as associations, compositions, and aggregations, most of them omitted in the figure for better intelligibility, express dependencies among the elements of the infrastructure.

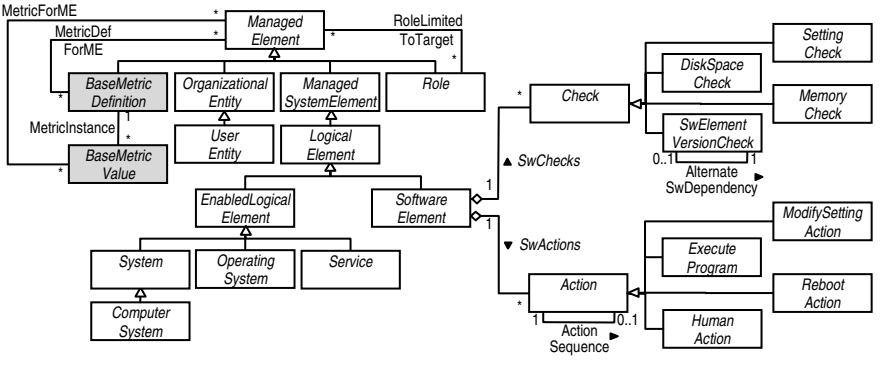

Fig. 2. Information model used to represent the IT Infrastructure and the DML

The model also incorporates the classes *Check* and *Action*, which denote necessary information for computing dependencies refining the activities of a preliminary plan. An instance of the class *Check* defines a condition to be met or a characteristic required for the associated software to evolve into a new state (*deployable*, *installable*, *executable* or *running*).

Fig. 3(a) illustrates instances of *Check* associated to the software *Squirrelmail*. The first two indicate that the software packages *Exim4* and *Apache2* must be installed before the *Squirrelmail* installation. The last two express that *Squirrelmail* requires 1MB of memory for installation and 3MB of disk space for execution. Each instance of the class *Action*, in turn, represents a mandatory activity to change the state of the associated software (*e.g.*, from *installable* to *executable*, *i.e.*, an installation process). Using once more the example of *Squirrelmail*, two activities associated with its installation process are illustrated in 3(a): *Install Squirrelmail*, responsible for the copies of *Squirrelmail* binary files; and *Configure Squirrelmail*, which performs the configurations necessary for the webmail to operate properly.

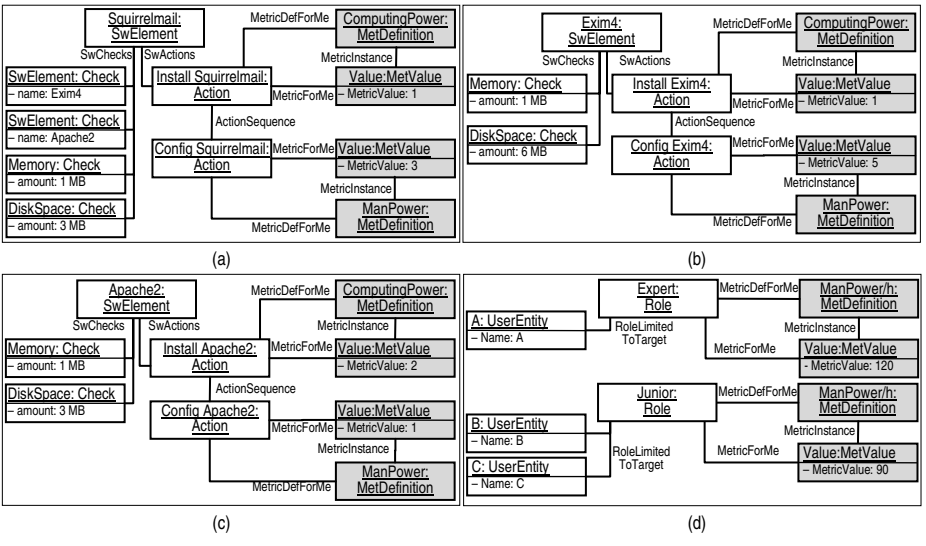

Fig. 3. Software Elements present in DML (a,b,c) and roles in the IT infrastructure (d)

The algorithm for change refinement, also known as *change_refinement*, receives as input the change specification to be refined, along with data about the current state of the managed infrastructure (from CMS) and the software packages from the DML. For each activity composing the preliminary plan, *change_refinement* computes the set of activities that should be executed before, so as to satisfy the requirements (*e.g.*, installation of software dependencies) for the activity to be executable.

To illustrate the refinement process, consider the RFC *Migrate E-mail Server*, presented in Fig. 4 (background in grey). It consists in the migration of the e-mail service from server *X* to server *Y*, and aims at minimizing the deployment time estimated in, at most, 120 minutes (in a way not to compromise the normal operations of the organization). The preliminary plan associated (background in white) is composed of six high level activities that range from the installation and configuration of a new server to the migration of data from the "old" (*X*) to the "new" server (*Y*).

Focusing on the activity *Install Squirrelmail* (illustrated in the preliminary plan), *change_refinement* identifies – through the *Checks* presented in DML (Fig. 3(a)) – that the execution of this activity depends on the previous installation of software

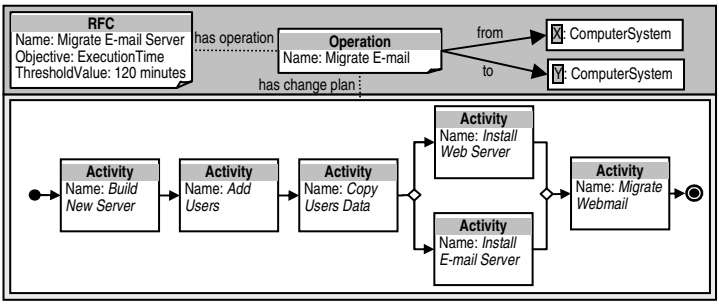

Fig. 4. Example of RFC and preliminary change plan for the migration of an e-mail server

packages *Apache2* (Web server) and *Exim4* (e-mail server). Consequently, two new activities are generated to satisfy the demands of *Install Squirrelmail*: *Install Apache2* (a) and *Install Exim4* (b). This process goes on recursively until all the requirements of the activities of the preliminary change plan are met. At the end of this process, when a detailed plan is obtained (Fig. 6), *change_refinement* stores the "current state" (refinement paths traversed to generate this plan). This step is important because, after the alignment (seen in detail in the Subsection 3.2), it will be possible to generate distinct, detailed plans, from the same initial specification, using alternative refinement "paths".

3.2 Change Aligner

The component *Change Aligner*, a key element for the alignment of detailed plans with business purposes, is grounded in three building blocks: (*i*) formalization of business objectives/constraints, (*ii*) information models to represent cost/capacity metrics, and (*iii*) algorithm for change alignment. These are described next.

3.2.1 Business Objectives/Constraints and Cost/Capability Metrics

To make it possible for change plans generated by the component *Change Refiner* to be aligned with business purposes, these must be specified in the RFC. Two business purpose categories are defined in this first iteration to solve the problem: *Execution Time* and *Execution Cost* to perform the change. These objectives were indicated as the most relevant ones by a group of network/system managers (working in Brazilian IT companies) interviewed. The specification of an objective/constraint needs to be accompanied by a *Threshold Value* representing the maximum time or cost tolerated. As shown in Fig. 4, the RFC *Migrate E-mail Server* specifies that the change, when planned, should consider at most 120 minutes for deployment.

Cost and capability metrics are also essential for the alignment to be performed. Whereas the former are used to describe the effort demanded in the execution of activities related to IT service operations and management, the latter allow expressing expectation on the performance of humans and computers to execute them. The metrics employed in this paper, described next, have already been adopted and consolidated in areas such as Distributed Systems [12], Software Engineering [13], and Economy [14]. Concerning information modeling, the metrics are specified, and their values instantiated, in the IT infrastructure model (Fig. 2) by means of the classes *Base Metric Definition* and *Base Metric Value*.

The cost metrics are subdivided in *Man Power* (MP) and *Computing Power* (CP). MP is used to quantify human effort required for executing an activity. CP, in turn, denotes computational effort required for executing an automated activity, *i.e.*, with no (or minimal) human intervention. Value instances for these metrics are used to express the costs of activities associated with software elements registered in the DML. In Fig. 3, it is possible to observe, as an illustration, that activities like *Install Squirrelmail* (a), *Install Exim4* (b), and *Install Apache2* (c) have an associated execution cost, in CP, of 1, 1, and 2 units, respectively. Additionally, the activities for the configuration of these services, namely *Config Squirrelmail* (a), *Config Exim4* (b), and *Config Apache 2* (c), have an associated cost, in MP, of 3, 5, and 1 units.

Similarly to what happens to cost metrics, the capability metrics are also subdivided in two: *Man Power/hour* (MP/H) and *Computing Power/hour* (CP/H). They represent the amount of work required from humans and computers to perform the execution of activities. In the case of MP/H, instance values are associated with roles played by the board of human resources available in the IT department of the organization. In the example illustrated in Fig. 3 (d), two roles are modelled, *Expert* and *Junior*, with MP/H equal to 120 and 90 units, respectively. Human resources are then associated to these roles, according to the functions they are able to perform in the department. MP/H instance values, conversely, are attributed to each of the computers present in the IT infrastructure, reflecting its expected performance.

We highlight that the investigation about ways of valuing costs and capabilities, although fundamental for the proposed solution, are not the scope of this paper. To fill in this gap, we assume the employment of one of the approaches proposed in the literature (some of them have been compiled by Jorgensen [15]). More specifically, in this paper, we adopted an empirical method, which consists of valuing simpler activities – not only automated but also of human-based execution – assigning to them a value of 1 CP and 1 MP, respectively. Having this as a basis, costs of other activities are estimated. The estimation of computing resource capabilities (CP/H) and human ones (MP/H), in turn, is performed by analysing historic data.

3.2.2 Alignment of Change Plans with Business Objectives/Constraints

The building blocks previously presented form the basis upon which the algorithm for aligning change plans with business purposes is built. The algorithm, shown in Fig. 5, is denoted by *change_alignment* (*C, DML, IT, H*), where *C* is the refined plan generated by *change_refinement*; *DML* is the repository of that contains information about software packages available; *IT* is the repository of information about the managed infrastructure; and *H* is the set of human resources available for conducting changes.

As a first step of the alignment process, an 'empty' change plan is created (line 1 in Fig. 5). After that, the activities contained in *C* are copied to the set *D* (line 2), and the function *recursive_alignment* is invoked (4).

The alignment process, as described in *recursive_alignment*, starts by verifying if both *D* (the set of activities which have not been processed) is empty and plan *C'* received satisfies the business objectives determined in the *RFC* (line 7). If so, plan *C'* is returned as a solution. Otherwise, the following steps are executed. First, an activity a_i is extracted from set *D* for processing (line 9). Second, the algorithm verifies if a_i requires a human to be executed (line 10). In case it does, the algorithm tries to allocate the most capable human *h* (available in *H*, the group of human resources), who has not

been allocated in any other parallel activity to a_i (lines 11-18). Human resources are allocated in a descending order in relation to the role they are associated with. Analyzing the roles *Junior* and *Expert* illustrated in Fig. 3(d), the most capable humans belong to *Expert's*.

It is important to mention that, for the alignment computation, the allocation of humans is conducted in a "non-nominal" way, *i.e.*, there is the specification that a human playing role *x* should execute a set of activities. The effective (or nominal) allocation of human resources to the activities is postponed to the change scheduling stage, which is out of the scope of this paper and has been investigated by several related work [3, 7].

```
01 change_alignment (C, DML, IT, H)
02   C' ← empty change plan
03   D ← set of activities from C
04   recursive_alignment (C, C', D, DML, IT, H)
05
06 recursive_alignment (C, C', D, DML, IT, H)
07   if D is empty, cost of deploying C' comply with objective threshold, given DML, IT, then
08     return C' as solution
09   extract activity ai from D
10   if ai is a human activity then
11     for all h in H do
12       for all aj in C do
13         allocable ← true
14         if ai is parallel to aj in C and h is allocated to aj then
15           allocable  ← false
16       if allocable is true then
17         associate human h to ai
18       add activity ai to change plan C'
19       if cost of deploying C' comply with objective threshold, given DML, IT, then
20         recursive_alignment (C, C', D, DML, IT, H)
21   else
22     add activity ai to change plan C'
23     if cost of deploying C' comply with objective threshold, given DML, IT, then
24       recursive_alignment (C, C', D, DML, IT, H)
25   return failure
```

Fig. 5. Algorithm for the alignment of change plans with business objectives/constraints

Once a human is allocated to a_i, the algorithm verifies the cost of executing plan *C'* on the managed infrastructure, considering such allocation (line 19). If the cost does not exceed *Threshold Value* (specified in the RFC), the alignment process continues recursively (line 20), in order to process the remaining activities in *D*. The semantics of *cost* (and *threshold*) depends on the RFC objective/constraint. For example, if *Execution Time* is chosen, then *cost* and *threshold* are processed considering time units.

Alternatively, the activity a_i extracted from set *D* may not require human intervention (line 21). In this case, it is directly added to the plan that will be returned as a solution (line 22). The processing continues recursively (line 24), in case the execution cost of *C'* is aligned with the defined objectives (line 23).

If none of the humans available can be allocated to a_i or the cost for executing *C'* (added to a_i) exceeds *Threshold Value*, the alignment process fails (line 25). Consequently, the allocation of human resources done up to this moment are recursively rolled back, and new alternatives for the alignment are explored. In the worst case, *recursive_alignment* is not able to produce a plan aligned with the business purpose specified in the RFC, in which case a detailed feedback is returned to the operator.

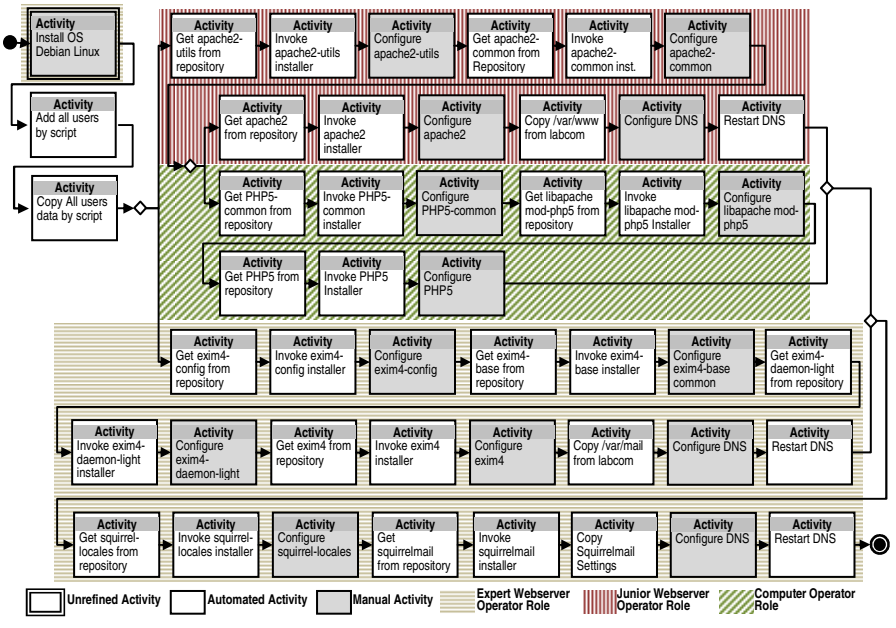

Fig. 6. Detailed change plan for the RFC Migrate E-mail Server

To illustrate the alignment process, consider the change plan presented in Fig. 6, generated for the RFC *Migrate E-mail Server* (Fig. 4). The different human roles allocated for the plan execution are represented using hachured areas around the activities, which have a grey background when human intervention is required. In this example, there is only one human associated with each role defined in *H*, being *Expert Webserver Operator*, *Junior Webserver Operator,* and *Computer Operator* the roles from where humans have been chosen. The presented plan meets the *Execution Time* objective, according to what was specified in the RFC. Note that the activities related to the installation of *Exim4* (e-mail system) and that require human intervention have a higher cost to be executed. Therefore, they are associated with the most capable available human (belonging to role *Expert Webserver Operator*). In the same way, the allocation of humans for other activities in the plan is done considering the existent parallelisms, as well as the associated costs.

4 Implementation

The solution to support alignment of change plans with business purposes presented in Section 3 has been implemented as a prototype of a decision support system called CHANGEADVISOR. This system is a key part of CHANGELEDGE [2], a framework for change management focused on the reuse of knowledge and automation. In this section, we present some of the main aspects of the implemented system, as well as its interfaces with the CHANGELEDGE components.

CHANGEADVISOR, implemented using the Java programming language, performs the functionalities of the components *Change Refiner* and *Change Aligner* (grey area of Fig. 1). The change specifications consumed by the system are coded using XML and processed through the *Streaming API for XML* (StAX). From the received specifications, the system generates detailed change plans (executable workflows), which are also coded in XML and respect the *Business Execution Language* (BPEL) [16]. The choice for BPEL is due to the widespread use of the standard and to its suitability for coordinating distributed activities in IT infrastructures. In regard to data handling and persistence (*e.g.*, about the managed infrastructure and the software packages available), both the Hibernate object-relational mapper and the MySQL database management system have been employed.

The communication between CHANGELEDGE components and the CHANGEADVISOR system is done through well-defined interfaces. Among the existing interfaces, the ones that "connect" the system with the components *Change Designer* and *Deployment System* are highlighted. In the former, *Change Designer* generates XML documents, adopting as a pattern the same XML schema as CHANGEADVISOR. In the latter, the BPEL documents generated by CHANGEADVISOR are consumed by the orchestrator ActiveBPEL, which implements the *Deployment System* functionalities. Please refer to [8] for additional information concerning the operation of the *Deployment System.*

5 Experimental Evaluation

In order to evaluate the technical feasibility of IT change planning aligned with business objectives/constraints, we have carried out several experiments using CHANGEADVISOR. RFCs with different objectives/constraints have been submitted, considering the same IT infrastructure as a basis. For space limitations, the analysis is focused on two of the changes.

Table 1. Number of humans, cost, and capacity per role

Role	Number of Available Operators	Allocation Cost ($ / hour)	ManPower / Hour
Expert Webserver Operator	1	40	2
Junior Webserver Operator	1	25	1.5
Computer Operator	3	12	1

Table 2. RFCs used in the experimental evaluation

Scenario	RFC	Goal	Objective/Constraint	Threshold Value
1	1	Migrate Webmail from server A to C	Reduce Execution Time	120
2	1	Migrate Webmail from server A to C	Reduce Execution Cost	15
3	1	Migrate Webmail from server A to C	None	-
4	2	Install Web Application in Server B	Reduce Execution Time	120
5	2	Install Web Application in Server B	Reduce Execution Cost	15
6	2	Install Web Application in Server B	None	-

The IT infrastructure employed in this evaluation is composed by 3 servers – *A*, *B* and *C* – out of which only *A* has *Debian GNU/Linux* installed. There are also three roles available, from which humans resources may be chosen to assist change execution. Table 1 shows the number of humans playing each role, the capacity, and the allocation cost per hour (per role). Table 2 describes the scenarios employed in the evaluation, highlighting the goals of the RFCs, as well as their respective objectives/constraints.

A partial view of the plans generated for RFC 1, in scenarios 1 and 2, is presented in Figs. 6 and 7, respectively. Due to space restrictions, the refinement of *Install Debian GNU/Linux* (Fig. 6) and *Install Windows 2003 Server* (Fig. 7) is suppressed in both figures, as well as decision structures. The activities present in the illustrated plans are consistently connected and ordered, respecting the dependency information specified in the DML. Note, for example, that the installation of *apache2* in *Debian GNU/Linux* is correctly preceded by the installation of its basic libraries.

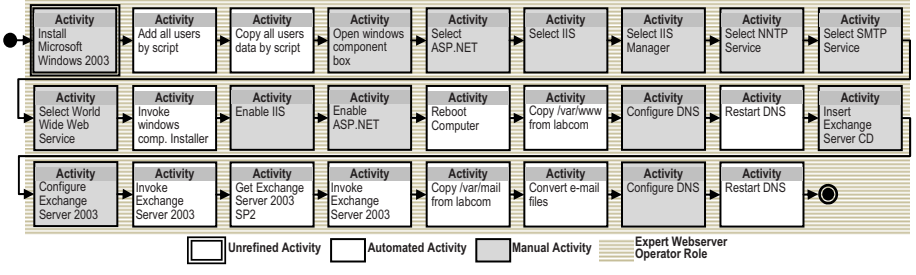

Fig. 7. Alternative change plan for the RFC Migrate E-mail Server

One can also observe in Figs. 6 and 7 that different human roles are required for the execution of different "niches" of manual activities, so as to meet the specified business objective/constraint. For example, the change plan generated for scenario 1 requires three humans, each playing a different role (one *Expert Webserver Operator*, one *Junior Webserver Operator,* and one *Computer Operator*) for the determined objective to be met. Conversely, the plan for scenario 2 requires only one operator (*Expert Webserver Operator*) to meet the determined objective.

Table 3. Estimated execution time and human allocation cost for the generated detailed plans

Scenario	RFC	Activities	Estimated Execution Time (min)	Human Allocation Cost ($)
1	1	69	118.75	34.21
2	1	38	156.12	7.75
3	1	68	157.08	45.98
4	2	52	75.5	18.73
5	2	39	211.31	10.42
6	2	49	250.75	28.43

Table 3 presents costs associated to the execution of the generated change plans for each of the evaluated scenarios. Observe that the resulting detailed plans exhibit satisfactory results in relation to the determined business purpose. For instance, RFC 1 could be accomplished by two distinct plans: one that has a shorter execution time (118 min), although having a higher implementation cost ($34.21), or another one which has a reasonable cost ($7.75), in spite of demanding a longer execution time (156 min). If analyzed individually, these cost differences may be insignificant. However, assuming that many changes may be required on a daily basis within an organization, such an alignment has the potential to reduce operational costs significantly.

Table 3 also enumerates, for the sake of comparison, change plans not aligned with objectives/constraints. Although correct, such plans have higher costs if compared to the plans that follow any alignment. For example, the plan generated for scenario 3 has an estimated cost of $45.98 with human allocation, an amount 493% higher than the cost of the aligned plan. This is explained by the different possible refinements for a same preliminary plan. If objectives/constraints are not taken into account, the exploration space gets wider, possibly leading to plans that violate business objectives/constraints.

In regard to performance, CHANGEADVISOR has required from a few hundreds of milliseconds to a few seconds to generate each aligned change plan. This is a negligible processing cost, especially if compared to the time humans would require to do the same job.

6 Conclusions and Future Work

Change planning is a fundamental element for the operation and management of Information Technology infrastructures and services. Existing automated solutions, however, do not take into consideration the generation of change plans aligned with objectives determined in the business level. Consequently, the planning and execution of changes will present little (or no) relation with the objectives/constraints determined by the change initiator. In order to tackle this problem, in this paper we have proposed CHANGEADVISOR, a solution for generation of change plans, which are consistent with objectives/constraints determined in the business level.

The results obtained, although not exhaustive, confirm the potential benefits of aligning change plans with business purposes. As previously mentioned, several detailed plans have been obtained from a same RFC, each of them satisfying specific execution time and cost requirements. One may also observe that the generated plans were sensibly different in form (comprising activities) as well as in associated costs (time and financial costs). Finally, the system has performed satisfactorily with respect to the time spent in the generation of the plans (kept in the magnitude of seconds). This time is certainly shorter than the time that would be spent by an experienced operator to manually design the plans using a *workflow* editor.

As future work, we intend to: (*i*) extend the scope of the alignment process so as to support two or more simultaneous objectives/constraints; (*ii*) investigate a richer set of objectives/constraints; and (*iii*) explore different heuristics (*e.g.*, for human resource allocation) to be used by the alignment algorithm.

References

1. Information Technology Infrastructure Library. ITIL V3. Office of Government Commerce (OGC) (2008), http://www.itil-officialsite.com
2. Cordeiro, W., Machado, G., Daitx, F., et al.: A Template-based Solution to Support Knowledge Reuse in IT Change Design. In: IFIP/IEEE Network Operations and Management Symposium (NOMS 2008), pp. 355–362 (2008)
3. Keller, A., Hellerstein, J.L., Wolf, J.L., et al.: The CHAMPS System: Change Management with Planning and Scheduling. In: IEEE/IFIP Network Operations and Management Symposium (NOMS 2004), pp. 395–408 (2004)
4. Rodosek, G.D.: A Generic Model for IT Services and Service Management. In: IFIP/IEEE International Integrated Network Management (IM 2003), pp. 171–184 (2003)
5. Brown, A.B., Keller, A.: A Best Practice Approach for Automating IT Management Processes. In: IFIP/IEEE Network Operations and Management Symposium (NOMS 2006), pp. 33–44 (2006)
6. Moura, A., Sauvé, J., Bartolini, C.: Business-driven IT Management - Upping the Ante of IT: Exploring the Linkage between IT and Business to Improve both IT and Business Results. IEEE Communications Magazine 46(10), 148–153 (2008)
7. Trastour, D., Rahmouni, M., Bartolini, C.: Activity-Based Scheduling of IT Changes. In: Bandara, A.K., Burgess, M. (eds.) AIMS 2007. LNCS, vol. 4543, pp. 73–84. Springer, Heidelberg (2007)
8. Machado, G., Daitx, F., Cordeiro, W., et al.: Enabling Rollback Support in IT Change Management Systems. In: IFIP/IEEE Network Operations and Management Sympo-sium (NOMS 2008), pp. 347–354 (2008)
9. Cordeiro, W., Machado, G., Andreis, F., et al.: A Runtime Constraint-Aware Solution for Automated Refinement of IT Change Plans. In: De Turck, F., Kellerer, W., Kormentzas, G. (eds.) DSOM 2008. LNCS, vol. 5273, pp. 69–82. Springer, Heidelberg (2008)
10. Keller, A.: Automating the Change Management Process with Electronic Contracts. In: IEEE International Conference on E-Commerce Technology Workshops (CECW 2005), pp. 99–108 (2005)
11. Distributed Management Task Force (DMTF): Common Information Model, http://www.dmtf.org/standards/cim
12. Shepherd, M.: Special Feature Distributed Computing Power: a Key to Productivity. IEEE Computer 10(11), 66–74 (1977)
13. Pillai, K., Sukumaran Nair, V.S.: A Model for Software Development Effort and Cost Estimation. IEEE Transactions on Software Engineering 23(8), 485–497 (1997)
14. Banyahia, H.: Costs and Productivity Estimation in Computer Engineering Economics. Engineering Economist 41(3), 229–241 (1996)
15. Jorgensen, M., Shepperd, M.: A Systematic Review of Software Development Cost Estimation Studies. IEEE Transactions on Software Engineering 33(1), 33–53 (2007)
16. Organization for the Advancement of Structured Information Standards. OASIS: Business Process Execution Language, version 2.0, http://docs.oasis-open.org/wsbpel/2.0