IT Support Conversation Manager: A Conversation-Centered Approach and Tool for IT Incident Management

Hamid R. Motahari-Nezhad, Claudio Bartolini, Sven Graupner, Sharad Singhal, Susan Spence

HP Laboratories
HPL-2010-46

Keyword(s):
Ad-hoc Business Processes, Best Practice Process Frameworks, IT Processes, Collaboration Applications

Abstract:

There is a push in the enterprise towards facilitating processes from best practice frameworks (such as the IT Infrastructure Library (ITIL)) to make them more repeatable, efficient and cost-effective. Best practice processes provide descriptive, high level guidelines rather than prescriptive, precise process model definitions. They are meant to be followed by people and may be adapted and enacted differently in various realizations. Currently, ITIL processes are supported by tools that hard code an interpretation of the process logic, or by people who use productivity tools. This is inefficient due to the rigidity of process logic encoded in existing tools which does not allow collaborative and flexible realizations of processes. Moreover, there is the issue of information loss when people are only using rigid productivity tools that force them to collaborate outside of tools. In this paper, we present a light-weight conversation-centered approach and a tool for dynamic and flexible definition and enactment of best practice processes in a collaborative and interactive manner. It addresses the issue of information loss by using the concept of a conversation as a container of information about the interaction among people in the context of a process. It offers a process template definition language and a semi-structured process model that supports flexible and adaptive processes. We showcase the approach using an illustrative use case based on best practice processes from ITIL, namely incident and problem management.

External Posting Date: April 6, 2010 [Fulltext]	Approved for External Publication
Internal Posting Date: April 6, 2010 [Fulltext]

IT Support Conversation Manager:

A Conversation-Centered Approach and Tool for IT Incident Management

Hamid R. Motahari-Nezhad, Claudio Bartolini, Sven Graupner, Sharad Singhal, Susan Spence
Service Automation and Integration Lab
Hewlett Packard Laboratories
Palo Alto, CA, USA
{hamid.motahari,claudio.bartolini,sven.graupner,sharad.singhal,susan.spence}@hp.com

***Abstract*—There is a push in the enterprise towards facilitating processes from best practice frameworks (such as the IT Infrastructure Library (ITIL)) to make them more repeatable, efficient and cost-effective. Best practice processes provide descriptive, high level guidelines rather than prescriptive, precise process model definitions. They are meant to be followed by people and may be adapted and enacted differently in various realizations. Currently, ITIL processes are supported by tools that hard code an interpretation of the process logic, or by people who use productivity tools. This is inefficient due to *the rigidity of process logic* encoded in existing tools which does not allow collaborative and flexible realizations of processes. Moreover, there is the issue of *information loss* when people are only using rigid productivity tools that force them to collaborate outside of tools. In this paper, we present a light-weight conversation-centered approach and a tool for dynamic and flexible definition and enactment of best practice processes in a collaborative and interactive manner. It addresses the issue of information loss by using the concept of a conversation as a container of information about the interaction among people in the context of a process. It offers a process template definition language and a semi-structured process model that supports flexible and adaptive processes. We showcase the approach using an illustrative use case based on best practice processes from ITIL, namely incident and problem management.**

Keywords- Ad-hoc Business Processes, Best Practice Process Frameworks, IT Processes, Collaboration Applications

I. INTRODUCTION

Best practice process frameworks such as the IT Infrastructure Library (ITIL) [1] and eTOM [19] provide a high level description and guidance for various processes rather than offering a precise definition of process models. These processes identify what should be done (in terms of steps that are meant to be followed by people) rather than the specificities of how they are done. People often interpret and follow these processes according to a particular work and project context. Recently, there has been a push in the enterprise towards facilitating and streamlining the realization of best practice processes (from ITIL) to make them more flexible, repeatable, traceable and more cost efficient.

Currently, there are two main problems in achieving this goal: (i) *Information loss*: there is valuable information about the realization of best practice processes in enterprise that is lost while carrying out activities and during hand-offs between different teams or individuals. There are many scenarios in which information loss is a major concern: a) IT staff may only use productivity tools to carry out some of the best practice processes. Therefore, it is hard to provide visibility on how the work was done, how it is tracked, as well as how to learn from previous realizations of a process. b) When tools exist, there are interactions among people related to the process that happen outside the tools, because these tools do not support the collaborations between people for the definition and enactment of the process. Such collaboration may occur between a group of people that has been formed in an ad-hoc manner to handle a specific case (e.g., a difficult incident). c) When work spans more than one ITIL process (e.g., across incident management and problem management processes for dealing with more substantial cases requiring root cause analysis). In these cases, the hand-offs between teams and supporting tools can lead to information loss.

(ii) *Rigidity of the process definitions and tools*: there are tools (such as HP Service Manager [2]) that support the automation of some ITIL processes by encoding a specific interpretation of them into program logic. However, processes described in frameworks are meant to be adaptable and updatable in each realization during the execution (e.g., for managing incidents differently based on the scope of the incident, required skills and effort). Rigidly defining these processes, hard-coded into tools, does not allow for the flexibility and dynamic adaptation needed. While straight-forward cases might properly fit the built-in logic, more complex cases may not. These processes are usually refined and followed in a collaborative and flexible manner. Thus, there is a need for tools that support people in flexible realizations of the best practice processes rather than forcing people into hard coded processes in tools.

In this paper, we present a *conversation*-centered approach to address the above problems and support the flexibility and dynamicity of descriptive processes in process frameworks (in particular ITIL). The contributions of this paper lie in leveraging the advances in business processes and human interaction paradigms to offer a novel approach and a lightweight tool that bridge business processes, collaboration tools and enterprise applications. In particular, we make the following contributions:

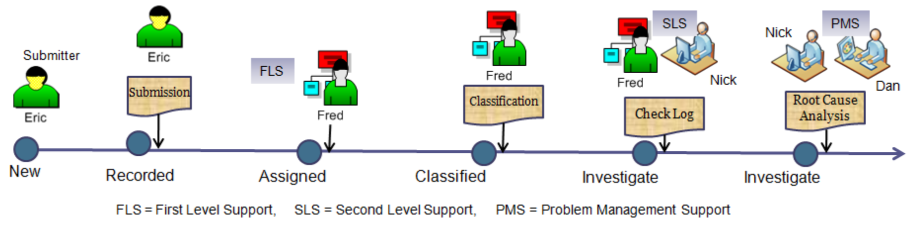


Figure 1. The example IT incident management scenario

(i) We introduce the notion of a *conversation* as a logical container for capturing the interactions of people around the definition, refinement and enactment of a best practice process or a set of related processes. A conversation includes the informal thread of interactions between participants (chat, email threads, attachments, etc), and a more structured flow of work activities consisting of a set of tasklets and their dependency model. Tasklets can be simple or composed of other tasklets. The tasklets dependency model shows the precedence of tasklets in terms of their order of execution. We choose to represent the dependency model using a dependency graph. This dependency model is used by a supporting monitoring engine that maintains the tasklets dependency model, updates it as the result of changes made by conversation participants, and supports the execution of the tasklets by participants via tracking tasklets, enabling automatic work allocation and progress monitoring. The involvement of participants in fulfillment of each tasklet is specified by a RACI (Responsible, Accountable, Consulted, Informed) model [3]. This model and the supporting engine enable the flexible definition and dynamic update of ad-hoc processes (ad-hoc processes are not pre-defined and emerge as the work progresses) by participants in a gradual manner.

(ii) We use *conversation templates* as a way to represent ITIL processes in a machine-interpretable manner, and introduce a method for framing ITIL process templates. Conversation templates can be used to initiate a specific conversation between people. They can be refined and updated. The use of templates enables us to take the process logic outside the programmed logic making it adaptive rather than rigid.

(iii) We present a system called *IT Support Conversation Manager* which supports the *guided* interaction of people in fulfilling an IT function in a *flexible*, *adaptive* and *collaborative* manner. It includes an interaction manager that captures informal interactions of people around the process activities. It enables the collaborative and gradual definition and dynamic update of activities (and corresponding tasklet model) as the result of the actions of the conversation participants. In addition, it provides automated assistances for monitoring and tracking the execution of tasklets and sending notifications related to the progression status of tasklets to participants. It also proactively suggests next steps in the conversation to participants based on the conversation templates and past similar conversations. The similarity of conversations is identified based on conversation information including the incident classification and the similarity of conversations' tasklet models.

(iv) We present an implementation of the conversation manager and illustrate the approach with an ITIL use case in the domain of IT incident and problem management to demonstrate the proposed IT support conversation manager.

The paper is structured as follows. Section II presents a motivating scenario and characterizes best practice frameworks. Section III presents the concepts and definitions of conversation and the theoretical foundation of the IT support conversation manager. Section IV presents the architecture, design and functionality of the IT support conversation manager. In Section V, we present the implementation and the use case study. We discuss related work in Section VI. Finally, we conclude and present areas of future work in Section VII.

II. Motivating Scenario and Problem Statement

A. *Motivating Scenario: IT Incident and Problem Management*

In the following, we describe an example scenario from the ITIL incident management domain and show how best practice processes are currently supported by tools. We use this scenario to highlight the problem investigated in this paper. We then revisit it in Section V.B to demonstrate how it is supported using the proposed approach and the tool.

In the example, we assume that Eric, a user, encounters a data loss problem while working with software ABC. He decides to file an incident through the incident management application in the IT department. The IT department runs an incident management application that encodes incident management procedures from ITIL (see Figure 1 for an illustration of the example scenario). Eric creates a new incident case through the (Web-based) interface of the incident management application and enters the details of the problem. The incident management application has a fixed process built-in. The application creates a new incident case and inserts it into the queue for first level support (FLS), where it is assigned to a representative (Fred). Fred reviews the incident description. He notices that it is not an immediately resolvable issue that can be handled by FLS. He classifies the incident as a software operation issue and sends it to the queue for second level support (SLS) with a recommendation that the application log should be checked.

In second level support, Nick is assigned to this case. He checks ABC's log and notices that a series of errors occurred around the same time the problem was experienced. Nick sends an email to the operation team of ABC and learns that there has been no interruption of ABC during the reported timeframe that could have led to the data loss, and therefore concludes the problem might have originated in a software bug that occurs when ABC is used in specific situations. However, handling such issues is outside of the scope of incident management. He refers the case to another group in the IT department (ABC's problem management) by forwarding Eric's problem description to them. Nick also sends a message to Eric, advising him that this case has been referred to the problem management group for further investigation.

In the problem management group, Dan is assigned to this case. He reviews the problem description and wants to know what has been done on this case so far. However, he does not have access to the incident management application to review the details of taken actions, and starts the investigation of the problem by asking Eric for further information.

B. Characteristics of Best Practice Processes and Problem Statement

Best practice processes are descriptive rather than prescriptive. We use the term "best practice processes" and "descriptive process" interchangeably in the following. We characterize descriptive processes such as those presented in the ITIL (IT Infrastructure Library [1]) framework, and then consider how they are typically used by people and explain the problem tackled in this paper by looking at the example scenario.

Descriptive processes are intended to be followed by people. They can be characterized as follows: they are (i) *non-prescriptive*, i.e., no precise or formal definition of the process (model) is given but a description of goals, milestones and overall steps with no precise execution order is provided, (ii) *adaptive*, i.e., the identified steps may be updated for each specific realization of the process and at execution time, i.e., some may be skipped and new ones may be added.

Best practice processes are often used by people in a collaborative manner to structure and manage complex work domains. From this perspective, we can add the following characteristics to the above list for descriptive processes: (iii) they are defined, refined and executed *collaboratively* and *interactively* by people, before and during the time they take place. Furthermore, some IT operations and different stages of the lifecycles of a project may involve several descriptive processes. Therefore, another characteristic of descriptive processes related to their use can be stated as: (iv) there are usage scenarios that span several processes (e.g., both incident and problem management). Therefore, tools should enable *cross-process realizations and collaborations*.

The problem that we investigate in this paper is how to provide an approach and a tool that supports descriptive processes. In particular, we consider addressing two issues: (a) the rigid realization of descriptive processes in tools that does not allow for flexibility and adaptation, and sometimes forces users to work outside tools (as demonstrated in the example scenario, where Fred had to contact the operations team outside of the tool). We aim to offer a flexible and adaptive system that supports dynamic definition, refinement and enactment of descriptive processes, and (b) that prevents information loss that happens during hand-offs between teams and lifecycle stages (e.g., the hand-offs between incident and problem management teams).

III. CONVERSATIONS

A. Basic Concepts and Definitions

We introduce the notion of *conversation* as a conceptual container for all interactions among a number of participants to collaboratively define, refine and carry out a descriptive process. A conversation includes a number of participants, information related to an informal thread of interactions among people about the process as well as a structured definition of the process activities. More formally, we define a process-oriented conversation as follows:

Definition 1 (Conversation). A conversation c is a triple $c =< P, E, W >$ in which P is the set of participants, E is the time-ordered set of events (an event sequence) related to the informal thread of interactions among participants, and T is the tasklet model representing the formal definition of activities (tasklets) carried out in the process.

We further elaborate each of the conversation elements in the following.

Participants. The participants are described as $P = \{p \mid p \in R \vee p \in L\}$ where R is the set of roles and L is the set of people in the enterprise. A participant can take one of four roles in a tasklet: "Responsible", "Accountable", "Consulted" and "Informed" (we have adopted the RACI model [3] for assigning roles and responsibilities). The participant that is responsible should perform the tasklet, while an accountable participant usually has an authoritative managing role. Participants with consulted roles are those who can be approached for advice, and finally, participants in informed roles have an interest to be informed about the progress and the result of performing the tasklet. Note that not all these roles have to be assigned for a given tasklet, but any tasklet should have a participant assigned in a responsible and accountable role.

Events. The event sequence $E =< e_1, e_2, \ldots, e_n >$ is a time-ordered set of events in which $e_i \in E, 1 \leq i \leq n$ is the record of an action taken by a participant p_i in the conversation. Events refer to two types of actions by conversation participants: a speech act [21], i.e. a message (e.g., consisting of one or more consecutive sentences of text) sent by the same participant in a chat, or an email (note we may need adapters to intercept events from communication channels), and actions related to defining, updating and carrying out the process tasklets. The event sequence E allows us to provide features such as playback that enable the understanding of the informal interactions and progression of the process.

Tasklets. A conversation includes a set of tasklets. A tasklet represents either a predefined activity or an ad-hoc activity that is defined on-the-fly by participants during the conversation. Tasklets can be atomic or can be composed from other tasklets. An atomic tasklet can be carried out by participants, while a composite tasklet is abstract and its

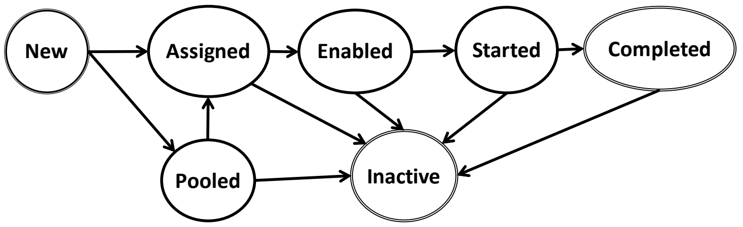


Figure 2. The lifecycle of an atomic tasklet

completion requires the completion of its subtasklets. A tasklet may have one of the following states: “new”, “assigned”, “pooled” (can be picked up by one of the participants), “enabled” (ready to be performed), “started”, “completed” and “in-active” (not part of the conversation anymore). Figure 2 shows the tasklet state transitions.

Each tasklet may have a set of input (and output) documents that are used (respectively, generated) while performing the tasklet. We refer to the set of documents manipulated in a conversation *c* as D_c. Tasklets may have dependencies on one another. We define the dependency relationship between two tasklets as either control precedence or data dependency. In case of data dependency, an input document for the dependent tasklet is produced by the depending tasklet. We define the tasklet model W for a conversation as follows:

Definition 2 (Tasklet model). A tasklet model T_c for a conversation c is a hierarchical directed graph represented with the pair $T_c = < W, X >$ where W is the set of tasklets (nodes in T), and $X \subseteq W \times W$ is the set of transitions. A tasklet $t \in T_c$ is defined as the triple $t = < I, O, s >$ in which $I, O \subseteq D_c$ is the set of inputs (outputs respectively), and s ∈ {new, assigned, pooled, enabled, started, completed, in-active} is its status. A transition x is represented as triple $< t_1, t_2, q > \in X$ meaning that the performance of tasklet t_2 depends on that of t_1 with the dependency type $q \in \{start, completion\}$. If q=“start” then t_2 is not enabled unless t1 is started, and if q=“completion” then t_2 is not enabled until t_1 is completed. If a tasklet t is composite, there is a child tasklet model T' associated with it.

To support users in the execution of tasklets in a conversation, we conceptually map the hierarchical tasklet model into a hierarchical colored Petri net (HCP-net) [10] and adopt its execution semantics. Adopting this semantics allows us to provide a monitoring engine that reacts to events related to the process definition and to enactment (see Section IV.A). It enables monitoring the progression of the process and provides participants with notifications about their tasklets. The mapping of the tasklet model to HCP-nets is conceptual meaning that we do not use Petri nets directly but use HCP-net execution semantics for evaluating the tasklet dependency graph. In general, the dependency graph of a tasklet model may consist of a set of disconnected sub-graphs. Each connected subgraph in this model is related to a set of dependent tasklets. When evaluating the dependency model from the execution semantics perspective, we form a single HCP-net by creating an additional initial place and transitions through which all sub-graphs are connected to form the overall tasklet model.

B. Templates

Another important concept in our approach for supporting descriptive processes is the *template*. A template captures a generic pattern of compositions of tasklets with dependencies among them, encoding the activities in descriptive processes. A template provides a starting point for a new realization of a descriptive process (e.g., for a new incident management case). Following our earlier work [6], we encode and formalize IT incident management and problem management processes as RDF graph descriptions. In Section V.B, we show the templates for the example scenario in this paper.

IV. IT Support Conversation Manager

In this section, we describe our system for the interactive and collaborative definition and enactment of descriptive processes.

A. Architecture

We have designed a generic system for the definition and management of descriptive processes to address the problems discussed in Section II.A. While the architecture and the description in the following are generic, we focus on descriptive processes from IT service management. Therefore, we refer to the system as IT Support Conversation Manager (ITSCM) in the following. The system could be adapted for other descriptive processes as well.

The ITSCM is designed as a service that exposes a set of APIs and has a Web-based interface. The main components of the ITSCM are those supporting the conversation, including the event and the tasklet models. The user interface mediates the interactions of users with the service. ITSCM is designed to be extensible by including plug-ins for other components and adapters for existing tools. The system’s architecture is shown in Figure 3. It has the following components:

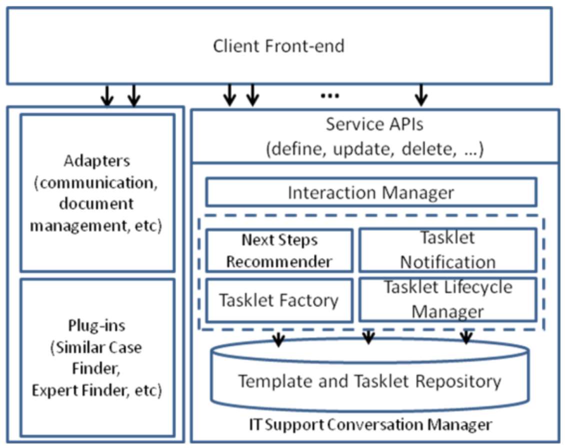

Client Front-end
...
Service APIs
(define, update, delete, ...)
Adapters
(communication, document management, etc)
Interaction Manager
Next Steps Recommender
Tasklet Notification
Tasklet Factory
Tasklet Lifecycle Manager
Template and Tasklet Repository
Plug-ins
(Similar Case Finder, Expert Finder, etc)
IT Support Conversation Manager

Figure 3. The architecture of IT Support Conversation Manager

a) The interaction manager: This component is responsible for establishing and managing the informal thread of interactions between participants. It records the events related to the interactions among people as well as between tasklet definitions, updates and enactments. This component also addresses the issue of information loss by providing a container for capturing the interactions among all stakeholders involved in handling a case (i.e., the participants of a conversation), and events showing the history of how the conversation evolves.

b) Components for tasklet definition, notification, recommendation and lifecycle manager: These components provide automated assistance for creating new tasklets, updating their status progression during their lifecycle and notifying interested parties, recommending next best steps and managing the lifecycle of tasklets, their dependencies and maintenance of the tasklet model of a conversation.

c) Templates and tasklet repository: this component stores information about templates from descriptive processes in best practices frameworks (ITIL in our case). The repository contains information about the conversations including events, the tasklet models and the execution information of each tasklet. Note that the templates enable us to address the issue of rigid processes by extracting and representing process tasklets in a form that can be further refined and updated.

d) Adapters: the adapters are components that enable existing tools and applications to be used in ITSCM (e.g., adapters to document management systems). Adapters capture related events while participants work with them in the context of a conversation. Adapters are also needed for enabling the use of various communication channels. The communication channels enable interaction between participants, e.g., chat and email.

The other components include the service APIs and the client front-end. The APIs expose the functionality of the IT support conversation manager as a Web service. The client front end is a Web-based application that implements the user interface and supports informal user interactions and their actions to define, view and update tasklets and perform work. In this architecture, plug-ins provide place holders for additional components to be integrated with ITSCM.

B. A Conversation-centered Approach for the Definition and Enactment of Descriptive Processes

We take a conversation-centered approach to enable the flexible and ad-hoc definition and enactment of descriptive processes in a collaborative and interactive manner, which is supported by the ITSCM architecture described above. In the following, we explain how each of the main ITCM components works and how they work together.

Interaction Manager. The interaction manager builds on top of existing communication mechanisms and provides a set of abstractions and techniques to assist the definition and enactment of descriptive, ad-hoc processes. In particular, it supports the abstractions of *conversations* and *views*, and offers *record/replay* and *tasklet assistance* features.

A view captures part of the interactions of a conversation between a specified subset of the participants. For instance, in the IT incident management domain, we may have a "submitter view", which is the view of the submitter of the incident (Eric). Another view may be the "helpdesk view" which is shared between Fred and other members of FLS (if they join the conversation). A conversation is a container for the information of all interactions between participants, and therefore, a conversation may include multiple views.

The record feature enables the recording of events in the interactions among people and the definition and enactment of the process (e.g., a message sent by a participant, or a new tasklet added by a participant) according to the definition of *E*. The replay feature enables conversation participants to review the sequence of events of interactions and tasklet updates related to the views that they are part of. This is important since it enables participants to understand what interactions and actions have been taken at which stages of the conversation and over the process progression.

The tasklet assistant supports participants in defining tasklets, creating dependencies on other tasklets when a new tasklet is created and updating the tasklet model (e.g., deleting tasklets considering their dependencies). The tasklet assistant looks at the tasklets assigned to users and reminds them of the tasklets to perform and tasklet status updates.

Tasklet Factory. A component in the architecture that allows users to create and instantiate new tasklets by defining their characteristics, filling in roles, and expressing dependencies on other tasklets. In particular, the factory may require a user to specify information such as ***start-date***, ***due-date***, ***end-date***, ***status***, ***actorIDs***, ***inputDocuments***, ***dependsOn***, and ***parent*** need to be specified. ***ActorIDs*** provide the list of the participants involved with their role(s) in accomplishing the tasklet. The property ***dependsOn*** gives the list of other tasklets the current tasklet depends on. A tasklet may not have any dependencies. The ***parent*** property takes as its value the conversation or another composite tasklet. The factory allows users to create tasklets either from scratch (as ad-hoc tasklets) or by refining pre-defined conversation templates based on best practices by adding new tasklets to them.

Tasklet Notification Service. This is a component in the architecture that notifies actors and other tasklets that have subscribed to a particular tasklet for status updates. The tasklet notification service is modeled after the publish-subscribe pattern. In particular, the actors of a tasklet are subscribed to status updates and get notified when changes in its status occur. For example, a notification is sent when a tasklet is enabled after its dependent tasklets have completed. The actors of a tasklet also can subscribe to the status of tasklets they depend on. An example of such a notification is notifying actors of a tasklet of the removal or cancellation of a tasklet they depend on. When actors are notified, they can respond accordingly.

Tasklet Lifecycle Manager. A component in the architecture that makes sure that the constraints expressed by the task dependencies are maintained and that necessary tasklet state transitions are executed according to the execution semantics. When a tasklet is created and instantiated by the factory, the lifecycle manager sets its status to the initial "new" status. During the tasklet's lifecycle, the lifecycle manager manages its state transitions consistently according to actors' actions and notifications from tasklets it depends upon. The lifecycle manager maintains the tasklet model and respective dependencies between tasklets based on actions that users take through operations of adding, removing and updating the status of tasklets. In particular, for removing a tasklet we take a consensus-based approach. When the removal request for a tasklet is made by a participant, this component triggers an event and a notification message is sent to all participants with "accountable" and "responsible" roles for the tasklet. Recipients can object to the removal. If no objections are received within a specified timeframe, the tasklet is removed. Upon removal, the dependency lists of all dependent tasklets on the removed tasklet are updated to replace the removed tasklet with the tasklets in its dependency list. The status of the removed tasklet is set to "in-active" meaning that it is no longer in the tasklet model of the conversation. The removed tasklet remains in the repository for tracking purposes. It can be restored if needed by the participants.

Next Best Steps Recommender. This is a component that considers the information of the current conversation to suggest best actions to participants. This component works based on two sources of information: the predefined best practice conversation template, and the tasklet models of similar conversations in the past. Note that the tasklet model of a past conversation may be the refined version of a best practice template. We define the similarity of conversations in terms of their tasklet model similarity in combination with incident classification information (which is collected by the helpdesk application for all IT incidents). We use the tree edit distance algorithm [22] to find similar tasklet models. This algorithm returns the number of edits (insert, deletes and relabeling) needed to transform the tasklet model of the current conversation to one of a past conversation. The conversations relating to the same class of incidents are considered more similar than others. The recommendation approach generates a list of tasklets as suggestions for new tasklets to be performed in the conversation. The list has two parts: tasklet(s) from the templates, and frequently used tasklets from past conversations. The number of recommended tasklets is configurable. This feature enables reusing past experiences and the knowledge for refined best practice templates.

V. Implementation and Use Case Study

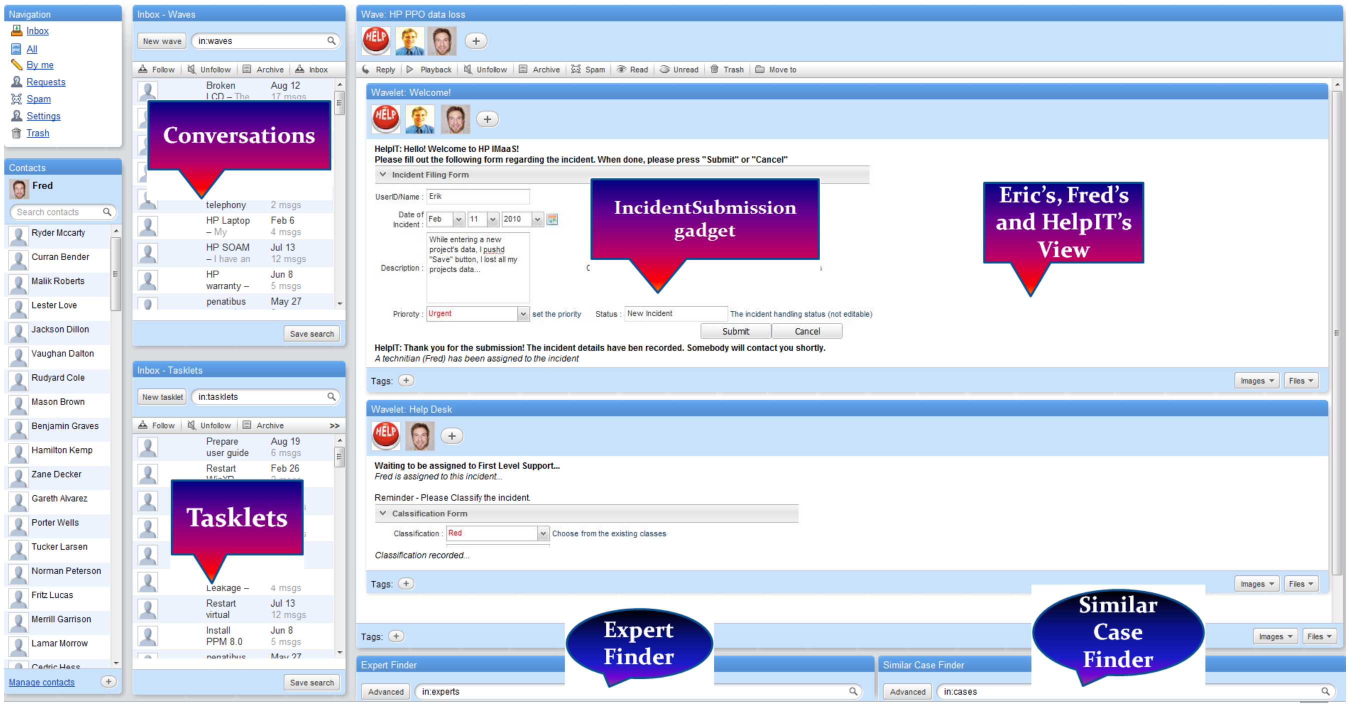
We describe the implementation of a prototype realization of IT conversation manager and a use case that shows how it facilitates the incident management scenario introduced in Section II.A.

A. Implementation

We have implemented the prototype IT Support Conversation Manager in Java including the tasklet factory, the tasklet lifecycle manager, the notification and next steps recommender. The user interface has been implemented using the Google Web Toolkit (GWT available at http://code.google.com/webtoolkit/). We have chosen to design the user interface similar to Google Wave (http://wave.google.com/). The design of the interface has been customized to accommodate the requirements of the interaction manager.

In particular, we have designed ITSCM for the IT incident management process. The ITSCM offers user interfaces for two roles: one for end users filing incidents (e.g., Eric), and the other for the help desk staff who handle incidents. The user interface for the end users such as Eric includes the submitter view which contains a specific gadget that allows to file an incident. ITSCM provides a robot called HelpIT that assists users in filing incidents, advises them on next steps and manages the initial interactions before a human representative joins the conversation.

Figure 4 shows the user interface of ITSCM for the help desk for a participant of the conversation from the help desk. On the left hand side it shows the conversation list of the participant and the list of the tasklets the user is involved in. On the right hand side, there are the details of a conversation with all views the participant has access to. When a person from the help desk is allocated to the case, he also sees the details of the incident (at the top-left in Figure 3 showing the view of Eric, Fred and HelpIT). In addition, at the bottom, there are two gadgets for the expert finder and the similar case finder. The HelpIT robot implements the notification service and the best next steps recommendation components. As part of the notification service, the robot also records all events relevant to the conversation and adds a corresponding notification message to the conversation thread.



Wavelet: Welcome!

HelpIT: Hello! Welcome to HP IMaaS!
Please fill out the following form regarding the incident. When done, please press "Submit" or "Cancel"

Incident Filing Form

UserID/Name : Erik
Date of Incident : Feb 11 2010
Description : While entering a new project's data, I pushd "Save" button, I lost all my projects data...
Prioroty : Urgent set the priority
Status : New Incident The incident handling status (not editable)
Submit Cancel

HelpIT: Thank you for the submission! The incident details have ben recorded. Somebody will contact you shortly.
A technitian (Fred) has been assigned to the incident

Wavelet: Help Desk

Waiting to be assigned to First Level Support...
Fred is assigned to this incident...

Reminder - Please Classify the incident.

Calssification Form

Classification : Red Choose from the existing classes

Classification recorded...

Figure 4. The screenshot of the frontend of the ITSCM prototype for helpdesk

The tasklets inbox window is expandable to enable internal participants to monitor the status and progress of the tasklets related to the case, as well as to update their properties. It also allows participants (e.g., Fred) to create a new tasklet model from existing templates for managing specific incidents (not illustrated in the figure). Other people can be invited to the conversation, as well.

B. *Use Case Study*

Using the example scenario introduced in Section 2.1, we now describe how ITSCM can be used to facilitate the tasks of incident submitters as well as representatives involved in incident and problem management, thus addressing the issues of rigid processes and information loss. The updated scenario is presented in Figure 5 (we omitted some details for simplification purposes).

Using ITSCM, submitter (Eric) creates a new conversation by clicking on the "Help IT!" link selecting "New Incident". This activates the HelpIT robot. HelpIT creates a new view called "Welcome" in which both HelpIT and Eric are participants. HelpIT and Eric are the only participants in the conversation at this stage. HelpIT greets

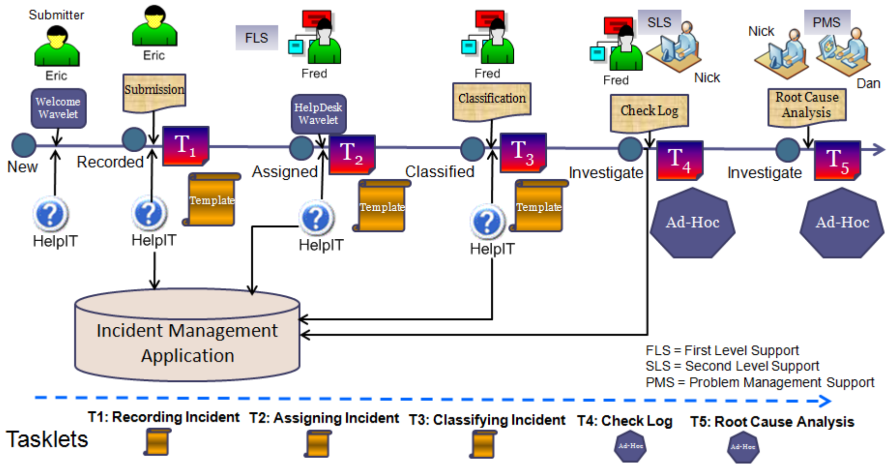


Figure 5. Applying the proposed approach to the incident management example scenario

Eric by adding a message to the Welcome view. HelpIT uses the process template for initiating the IT incident conversation.

Table 1 shows a fragment of tasklet templates related to incident management. Templates are used by HelpIT and interpreted during a conversation based on the current tasklets that are performed to guide the conversation. The fragment shows four tasklet templates and their dependencies. Incident-ManagementProcess is the general TaskLetTemplate for the overall incident management process. It is part of ITILOperationalManagement and classified as Required. A number of actors are listed, such as HelpDesk, FLS and SLS. CompositeSteps lists a break down into sub-activities following the ITIL methodology. The tasklet templates for the first three steps are shown as T1-T3 in Table 1. The first step is IncidentDetectionAndRecording (T1), initiated by the Requestor and performed by FLS. The second step is the Assignment of incidents (T2). T2 has a dependency on T1, which is reflected by the dependsOn property in its definition. The third step is Classification (T3), which has a dependency on T2. The remaining TaskLetTemplates (InitialSupport, InvestigationAnd-Diagnosis, ResolutionAndRecovery and Closure) follow a similar pattern and are not shown in Table 1.

As the first tasklet in the process template is for recording the incident, HelpIT adds the IncidentSubmission gadget to the Welcome view and invites Eric to fill in the information about the incident. The gadget has information fields about the incident, about the submitter and a field Status. In addition, HelpIT creates a tasklet in the tasklet model of the conversation assigned to HelpIT, which is represented by the tasklet T1 in Figure 5. Eric fills in the form and submits. The IncidentSubmission gadget notices the submission. It stores the information from the information fields in the form. HelpIT replies to Eric with a thank you message and with the information that someone will be assigned to it shortly.

Then, HelpIT adds a message to the Welcome view identifying that the incident has been recorded. It updates the status of the tasklet T1 to “completed”. Accordingly, HelpIT changes the Status field in the IncidentSubmission gadget to "recorded".

Next, according to the template, HelpIT creates a new view called HelpDesk view and adds a message indicating that the incident is waiting to be assigned to someone in First Level Support (FLS). The purpose of the HelpDesk view is to provide isolation between the submitter and the FLS team. It also creates a tasklet (T2 in Figure 5) related to the assignment of the case to FLS. HelpIT asks the incident management application to add this case to the queue of incidents waiting to be assigned to FLS.

Assuming that Fred from FLS is assigned to the incident, HelpIT adds Fred to the participant list, and to the Welcome and HelpDesk view. HelpIT adds a message to the Welcome view notifying Eric that a technician (Fred) has been assigned to the case. HelpIT updates the status of the T2 tasklet to “completed” and the status of the incident in the IncidentSubmission gadget in the Welcome view to “assigned”. Fred reviews the incident submission process using the replay feature.

Table 1. The RDF/N3 representation of the incident management process for the example scenario

```
@prefix m1:        <http://sm.hp.com/conceptdefs/itil/opman/incman> .
@prefix :          <http://sm.hp.com/templates/itil/opman/incman> .

:IncidentManagementProcess a m1:TaskLetTemplate ;
  m1:partOf :ITILOperationalManagement ;
  m1:rType m1:Required ;
  m1:actorRole m1:HelpDesk, m1:FirstLevelSupport, m1:SecondLevelSupport;
  m1:compositeTasklets (
    m1:IncidentDetectionAndRecording
    m1:Assignment
    m1:Classification
    m1:InitialSupport
    m1:InvestigationAndDiagnosis
    m1:ResolutionAndRecovery
    m1:Closure
  ) .


:IncidentDetectionAndRecording a m1:TaskLetTemplate ;   # T1
  m1:partOf :IncidentManagementProcess ;
  m1:rType m1:Required ;
  m1:initiatorRole m1:Requestor ;
  m1:actorRole m1:FirstLevelSupport ;
  m1:view:Welcome ;
  m1:compositeTasklets (
    m1:GatherFilingInformation
    m1:ProduceIncidentRecord
  ) .

:Assignment a m1:TaskLetTemplate ;     # T2
  m1:partOf :IncidentManagementProcess ;
  m1:rType m1:Required ;
  m1:initiatorRole m1:FirstLevelSupport ;
  m1:actorRole m1:FirstLevelSupport ;
  m1:view:HelpDesk ;
  m1:dependsOn :IncidentDetectionAndRecording ;
  m1:compositeTasklets (
    m1:PushInFLSQueue
    m1:UpdateAssignedSubmitter
  ) .

:Classification a m1:TaskLetTemplate ;     # T3
  m1:partOf :IncidentManagementProcess ;
  m1:rType m1:Required ;
  m1:initiatorRole m1:FirstLevelSupport ;
  m1:actorRole m1:FirstLevelSupport ;
  m1:view:HelpDesk ;
  m1:dependsOn :Assignment ;
  m1:compositeTasklets (
    m1:SelectClassificationSchema
    m1:ClassifyIncident
    m1:UpdateClassifySubmitter
  ) .
```

The next tasklet in the tasklet model is classifying the incident. HelpIT adds the Classify gadget to the HelpDesk view. HelpIT adds a tasklet called “Classifying Incident” (T3 in Figure 5) to the tasklets with the status of “assigned” (to Fred). HelpIT notifies Fred to classify the incident. Once Fred has finished classifying the incident (by pressing the “save” button), HelpIT updates the status in the IncidentSubmission gadget in the Welcome view to "classified", as well as updating the status of the T3 tasklet to “complete”.

Fred refers the case to Second Level Support (SLS) since it is outside the scope of the FLS task. Nick from SLS is assigned, and reviews the incident using the replay feature. As Nick needs to investigate further, he adds comments indicating that this case requires further investigation. He looks at the next best tasklet recommendation offered by HelpIT. The list shows that the template does not offer any more tasklets. However, the list of frequent tasklets from past conversations includes a number of tasklets with the most frequent tasklet called Check Log. HelpIT offers to assist Nick in creating the corresponding tasklet. Nick accepts. HelpIT adds a gadget for this purpose where Nick fills in the form and creates the tasklet Check Log (T4 in Figure 5). He adds people from the ABC's operation team with the role of "consulted" for this tasklet. HelpIT updates the status in the IncidentSubmission gadget to "investigate".

Nick interacts with ABC's operation team in the conversation and concludes that the problem is not with the operation, but that there is a software bug. Nick creates a tasklet called Root Cause Analysis (T5 in Figure 5) and assigns it to the problem management team. The tasklet T5 is routed to the queue of the problem management team, where Dan is assigned to this tasklet. Dan reviews the case using the playback feature. He can also look at the list of tasklets performed within this case. He agrees that the case is a problem that requires investigation and therefore creates a new view for the problem management team.

This scenario shows how participants can collaborate in the definition and enactment of best practice processes. In this scenario, tasklets T4 and T5 are not part of the process templates, but are defined while this specific incident/problem management case evolves, one based on previous conversations and the other by the participants interacting among each other. In addition, this approach addresses the information loss issue in the enactment of the process as well as for hand-offs between teams using a lightweight and flexible tool.

VI. Related Work

The presented approach and the tool draw on concepts from collaboration systems and business process management tools, creating a bridge between these technologies for the dynamic, flexible and adaptive definition and enactment of best practice frameworks such as ITIL.

There have been efforts to support people in best practice frameworks before. However, existing work has focused on formalizing and incorporating best practice processes as passive knowledge bases incorporated into Semantic Wiki [7] using ontologies [8]. The unified activity management approach in [17] proposes capturing the relationships between activities of people working together using ontological models and activity patterns. Orbus [13] formalizes ITIL processes as business process models expressed in BPMN. However, similar to encoding rigid ITIL processes in tools, this approach does not allow flexible and collaborative realization of descriptive processes.

Current workflow management systems [9] do not support descriptive processes which require ad-hoc interactions and collaboration among people. Workflow systems need a well-defined process model defined ahead of the execution time and allow only limited and restricted adaptation at runtime [10]. There is work on adding flexibility to workflows by enabling the refinement of tasks in pre-defined process models into subtasks [11] or incorporating business rules [12]. These approaches work within the framework of current workflow systems and explicit modeling of workflows at design time. Our approach and tool complements them by providing an adaptive, collaborative and conversation-centered approach to the definition and enactment of descriptive processes and also enables the capturing of informal threads of interactions related to the process realization.

Collaboratively defining process models is another thread of related work. Caramba [14] allows the definition of processes for virtual teams. In this work, the process definition is explicitly defined by the team members using a graphical process modeling tool. Also, Gravity [15] from SAP Research and Workflow-on-Wave (WoW) [16] allow users to collaboratively define process models. In our work, we take a rather complementary approach by leveraging the informal thread of interactions among people (conversation) as a context for the definition and enactment of descriptive processes. In addition, we do not expose process models explicitly to users (as they are knowledge workers) but we use the process model implicitly to help drive the work, monitor and assist tasklets progression.

Document sharing systems used in the enterprise such as Microsoft Sharepoint are passive repositories of documents. Our system can be built on top of them. Our system also complements document management systems which manage predefined document workflows. Collaboration tools simplify the communication between people and allow the creation and sharing of content in a collaborative manner. However, they are unaware of the work context and the structure of the processes in which they are used. In our approach, we leverage the advances in collaboration tools such as Google Wave to capture informal interactions between people. The proposed system differs from wiki-based collaboration systems (e.g. Semantic Media Wiki [7]) as Wikis provide passive information repositories in which work context and the interactions between people are not captured.

VII. Conclusion and Future Work

In this paper, we have presented a novel light-weight and enterprise-oriented approach and tool for establishing and managing conversations related to the definition and enactment of best practice processes. This work is backed by a semi-structured tasklet model and automation assistant components such as tasklets lifecycle manager, notification service and next best steps recommender. To the best of our knowledge, this is the first system that links concepts from informal conversations among people to the structured model of the tasklets and provides automation supports. It bridges the worlds of collaborative applications and business process management systems for descriptive and ad-hoc processes. We have implemented a prototype system and demonstrated it using a use case scenario from the ITIL incident management and problem management domains.

As future work, we are considering enhancing the capabilities of the tasklet assistant to make it an active participant who can proactively participate in conversations. An example in this direction is incorporating the language/action framework [20] as a basis to analyze text messages from participants as well as considering the informal thread of interactions in the conversation to make suggestions (including for the best next steps) to participants. As starting points in this direction, we are also in the process of incorporating the plugins of the similar case finder and expert finder in the tool, to allow the assistant agent to find similar cases and relevant experts to the conversation participants by tapping into databases of existing incidents in IT organizations. Another opportunity is learning from the existing threads of conversations to create more customized or refined process templates. As a next step, we are in the process having people in HP IT department to use it in their environment and finding opportunities for integrating it into the existing IT incident management tool in HP.

In summary, the introduced abstractions and tool simplify the work environment in defining and managing processes for best practice frameworks.

REFERENCES

[1] L. Hendriks and M. Carr, ITIL: Best Practice in IT Service Management, in: Van Bon, J. (Hrsg.): The Guide to IT Service Management, Band 1, London u. a. 2002, pp 131-150.

[2] HP BTO Software, HP Service Manager, http://www.managementsoftware.hp.com

[3] Brennan, Kevin. A Guide to the Business Analysis Body of Knowledge (Babok Guide). International Institute of Business Analysis. pp. 29. ISBN 0981129218. 2009.

[4] K. Jensen: Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use, Volumes 1-3, Monographs in Theoretical Computer Science, Springer-Verlag, ISBN: 3-540-60943-1, 2nd corrected printing, 1997.

[5] Choo, Y.: Hierarchical Nets: A Structured Petri Net Approach to Concurrency, Technical Report CaltechCSTR:1982.5044-tr-82, California Institute of Technology, 1982.

[6] Graupner, S., Motahari-Nezhad, H. R., Singhal, S., & Basu, S. (2009). Making Process from Best practices Frameworks Actionable. DDBP 2009: Second International Workshop on Dynamic and Declarative Business Processes, Co-located with EDOC 2009, September 1, 2009, Auckland, New Zealand.

[7] University of Karlsruhe, "Semantic Media Wiki," http://semantic-mediawiki.org.

[8] Zhenning Shangguan, Zhipeng Gao, Kai Zhu, "Ontology-Based Process Modeling Using eTOM and ITIL," In CONFENIS (2) pp. 1001-1010, 2007.

[9] Marlon Dumas, Wil M. van der Aalst, Arthur H. ter Hofstede, "Process Aware Information Systems: Bridging People and Software Through Process Technology," WileyBlackwell, 2005.

[10] Reichert, M.; Dadam, P.: ADEPT flex Supporting Dynamic Changes of Workflows Without Loosing Control. Journal of Intelligent Information Systems, 10 (1998) 2, 93-129.

[11] M. Adams, A.H.M. ter Hofstede, D. Edmond, and W.M.P. van der Aalst. Implementing dynamic flexibility in workflows using worklets. Report BPM-06-06. BPMCenter.org, 2006.

[12] Eijndhoven, T. v., Iacob, M., and Ponisio, M. L. Achieving Business Process Flexibility with Business Rules. In 12th international IEEE Enterprise Distributed Object Computing Conference (EDOC 2008). IEEE Computer Society, pp. 95-104. 2008.

[13] Orbus Software, The iServer ITIL Solution, http://www.orbussoftware.com/business-process-analysis/products/itil-solution/.

[14] S. Dustdar, "Caramba — A Process-Aware Collaboration System Supporting Ad hoc and Collaborative Processes in Virtual Teams," Distrib. Parallel Databases 15, 1 (2004), 45-66.

[15] Alexander Dreiling, Gravity – Collaborative Business Process Modelling within Google Wave, SAP Research, Sep. 2009.

[16] Itensil, Workflow-on-Wave (WoW), http://itensil.com.

[17] Moran, T. P., Cozzi, A., and Farrell, S. P. 2005. Unified activity management: supporting people in e-business. Commun. ACM 48, 12 (Dec. 2005), 67-70.

[18] Stacy Hobson, Sameer Patil, Public Disclosure versus Private Practice: Challenges in Business Process Management, Workshop on SOA, Globalization, People, & Work (SG-PAW 2009), Sweden.

[19] TeleManagement Forum, "Enhanced Telecom Operations Map (eTOM) – The Business Process Framework," www.tmforum.org/BestPracticesStandards/BusinessProcessFramework/1647/Home.html.

[20] K. Finn, and T. Winograd, The language-action approach to design of computer support for cooperative work, Proceedings of the IFIP TC8 Conference on Collaborative Work, Social Communications and Information Systems, Helsinki, Finland, 27-29 August, 1991.

[21] Searle, J.R. Speech acts: An essay in the philosophy of language Cambridge University, Cambridge, England, 1969.

[22] Philip Bille. A survey on tree edit distance and related problems. Theor. Comput. Sci., 337(1-3):217–239, 2005