IT Support Conversation Manager: A Conversation-Centered Approach and Tool for Managing Best Practice IT Processes

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

HP Laboratories
HPL-2010-46R1

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 either supported by tools that hard code an interpretation of the process logic, or followed by people using productivity tools. This is inefficient because existing tools hardcode a rigid logic of the processes, and do not support collaborative and flexible realizations of processes. Moreover, there is a risk of information loss when people using rigid productivity tools, and are forced to collaborate outside of those tools. In this paper, we present a conversation-centered approach and a tool that enables dynamic and flexible definition and enactment of best practice processes in a collaborative and interactive manner. We address the issue of information loss by using the concept of a conversation as a container of information about the interactions among people in the context of a process. A conversation is backed with a semi-structured process model and process templates to support flexible and adaptive process realization. We showcase the approach using an illustrative use case in incident and problem management, based on best practice processes from ITIL.

External Posting Date: August 6, 2010 [Fulltext] Approved for External Publication
Internal Posting Date: August 6, 2010 [Fulltext]
To be published at 14th IEEE International EDOC Conference (EDCC 2010), 25-29 October 2010.

IT Support Conversation Manager: A Conversation-Centered Approach and Tool for Managing Best Practice IT Processes

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 either supported by tools that hard code an interpretation of the process logic, or followed by people using productivity tools. This is inefficient because existing tools hardcode a *rigid logic of the processes*, and do not support collaborative and flexible realizations of processes. Moreover, there is a risk of *information loss* when people using rigid productivity tools, and are forced to collaborate outside of those tools. In this paper, we present a conversation-centered approach and a tool that enables dynamic and flexible definition and enactment of best practice processes in a collaborative and interactive manner. We address the issue of information loss by using the concept of a conversation as a container of information about the interactions among people in the context of a process. A conversation is backed with a semi-structured process model and process templates to support flexible and adaptive process realization. We showcase the approach using an illustrative use case in incident and problem management, based on best practice processes from ITIL.**

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 frameworks identify *what* should be done (in terms of steps that are meant to be followed by people) rather than prescribing how those steps are performed. People often interpret and follow processes in these frameworks within a particular work and project context, and thus adapt them as needed. Recently, there has been a push in the enterprise towards facilitating and streamlining the realization of best practice processes to make them more flexible, repeatable, traceable and 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 whilst people are 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 use only productivity tools to carry out some of the best practice processes. This makes it hard to track the work, to provide visibility into how the work was done, and to learn from previous realizations of a process. b) When tools do exist, people are forced to interact outside these tools, because they do not support collaboration between people for defining and enacting the process (e.g. groups pulled together on an ad-hoc basis to handle specific cases, such as difficult IT incidents). c) When work spans more than one ITIL process (e.g., spanning incident management and problem management processes for dealing with more substantial cases requiring root cause analysis).

(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 during execution in each realization (e.g., enabling incidents to be managed differently based on the scope of the incident, required skills and effort). Rigidly defining these processes, by hard-coding them into tools, does not allow for the flexibility and dynamic adaptation that is needed. While straight-forward cases might 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 them 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 dynamic adaptation of descriptive processes in process frameworks (in particular ITIL). We leverage the advances in business processes and human interaction paradigms to design a novel approach and provide a supporting tool that bridge business processes, collaboration tools and enterprise applications. In particular, we make the following contributions:

(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 among participants (chat, email threads, attachments, etc), and a more structured flow

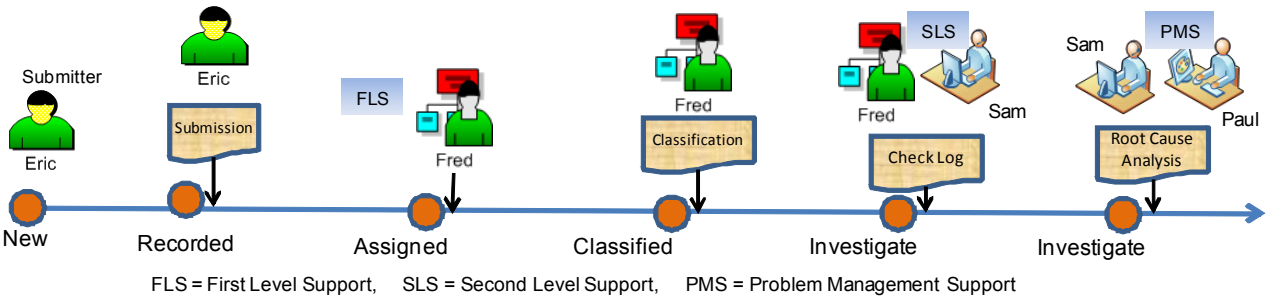

Figure 1. The example IT incident management scenario

of work activities consisting of a set of tasks and their dependencies. The task dependency model shows the precedence of tasks in terms of their order of execution. The dependency model is used by a monitoring engine that maintains the task dependency model and supports the execution of the tasks. The involvement of conversation participants 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.

(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.

(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 enables the collaborative and gradual definition and dynamic update of activities through the actions of the conversation participants. In addition, it provides automated assistance for monitoring and tracking the execution of tasks and for sending notifications about the progress of tasks to participants.

(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, characterizes best practice frameworks and presents the problem statement. Section III presents the concepts and definitions of a 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 this scenario is supported by 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, Sam 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. Sam 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. Sam 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, Paul 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. In the following, we characterize descriptive processes such as those presented in the ITIL (IT Infrastructure Library [1]) framework. We consider how they are typically used by people and describe 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* – 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; and (ii) *adaptive* – the identified steps may be updated for each specific realization of the process and they may be updated 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) they require *cross-process realizations and collaborations*, since there are usage scenarios that span several processes (e.g., both incident and problem management).

The problem that we investigate in this paper is how to support descriptive processes. In particular, we address two issues: (a) the rigid realization of descriptive processes in tools that do not allow for flexibility and adaptation, sometimes forcing users to work outside the tools (as demonstrated in the example scenario, where Fred had to contact the operations team outside of the tool), and (b) the 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 $c =< P, E, T >$ *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 task model representing the formal definition of activities (tasks) 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 task: "Responsible", "Accountable", "Consulted" or "Informed" (we have adopted the RACI model [3] for assigning roles and responsibilities). The participant that is responsible should perform the task, 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 are interested in being informed about the progress and the results of performing the task. Note that not all of these roles have to be assigned for a given task, but any task should have a participant assigned in a responsible and accountable role.

Events. The event sequence $E =< e_1, e_2, ..., e_n >$ is a time-ordered set of events in which $e_i \in E, 1 \le i \le n$ 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 (consisting of one or more consecutive sentences of text) sent by a participant in a chat or an email (note we may need adapters to intercept events from communication channels); and actions taken by the participant related to defining, updating or carrying out the process tasks. 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.

Tasks. A conversation includes a set of tasks. A task represents either a predefined activity or an ad-hoc activity that is defined on-the-fly by participants during the conversation. Tasks can be atomic or can be composed from other tasks. An atomic task can be carried out by participants, while a composite task is abstract and its completion requires the completion of its subtasks. A task 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 task state transitions.

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

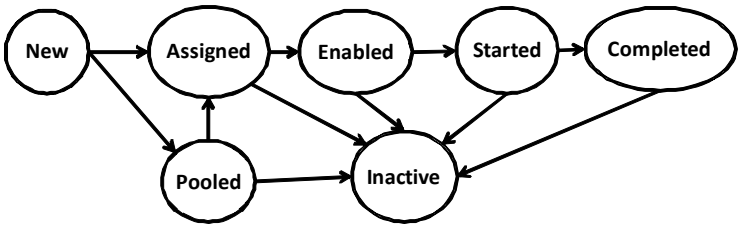

Figure 2. The lifecycle of an atomic task

Definition 2 (Task model). A task model T for a conversation c is a hierarchical directed graph represented with the pair $T = <\Gamma, X>$ *where* Γ *is the set of tasks (nodes in T), and* $X \subseteq \Gamma \times \Gamma$ *is the set of transitions. Each task* $\gamma = <\Gamma_\gamma, I, O, s>$, $\gamma \in \Gamma$ *in which* $\Gamma_\gamma \subset \Gamma$ *is the set of child tasks of* Γ *and* $I(,O) \subseteq D_c$ *is the set of inputs (and outputs respectively), and* $s \in$ *{new, assigned, pooled, enabled, started, completed, in-active} is its status. A transition x is represented as the triple* $< t_1, t_2, q > \in X$ *meaning that the performance of task* 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 task t is composite, there is a child task model T' associated with it.*

To support users in the execution of tasks in a conversation, we conceptually map the hierarchical task model into a hierarchical colored Petri net (HCP-net) [10] and adopt its execution semantics. According to this semantics, events lead to firing various transitions, and therefore enacting tasks. In particular, using this model the monitoring engine reacts to events related to the process definition and enactment by sending notifications and reminders to people about the progression of the process and status of tasks in which they are involved. The mapping of the task model to HCP-nets is conceptual, meaning that we do not use Petri nets directly but use HCP-net execution semantics for evaluating the task dependency graph. In general, the dependency graph of a task model may consist of a set of disconnected sub-graphs. Each connected subgraph in this model is related to a set of dependent tasks. 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 task 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 tasks 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 description that follows are generic, we focus on descriptive processes for IT service management. Therefore, we refer to the system as the IT Support Conversation Manager (ITSCM). However, 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 task 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:

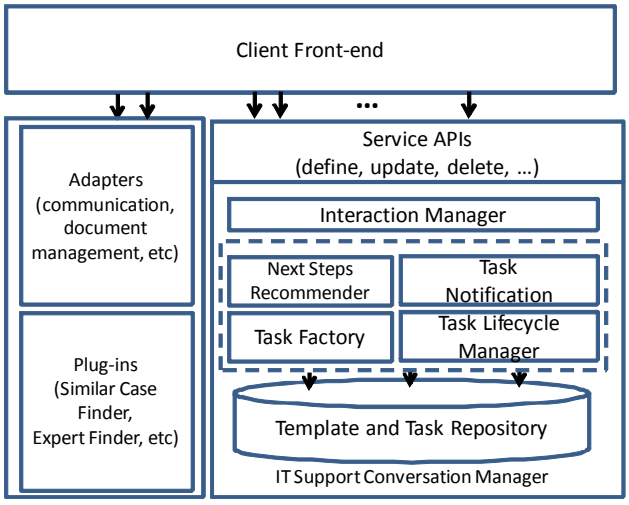

Figure 3. The architecture of IT Support Conversation Manager

a) The interaction manager: This component is responsible for establishing and managing the informal thread of interactions among participants. It records the events related to the interactions among people as well as between task 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 task definition, notification, recommendation and the lifecycle manager: These components provide automated assistance for creating new tasks, updating their status during their lifecycle and notifying interested parties of status update. They also recommend next best steps, manage the lifecycle of tasks and tasks their dependencies, and maintain the task model of a conversation.

c) Templates and task 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 task models and the execution information for each task. Note that the templates enable us to address the issue of rigid processes by extracting and representing process tasks 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, supporting informal user interactions, and their actions to define, view and update tasks 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, supported by the ITSCM architecture described above. In the following, we explain how each of the main ITSCM 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 *task 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 among 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 task added by a participant) according to the definition of *E*. The replay feature enables conversation participants to review the sequence of events in interactions and the task 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 during the progress of the process.

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

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

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

Task Lifecycle Manager. The Lifecycle Manager is a component that makes sure that the constraints expressed by the task dependencies are maintained, and that necessary task

state transitions are executed according to the execution semantics. When a task is created and instantiated by the factory, the lifecycle manager sets its status to the initial "new" status. During the task's lifecycle, the lifecycle manager manages its state transitions consistently according to actors' actions and notifications from tasks it depends upon. The lifecycle manager maintains the task model and respective dependencies between tasks based on actions that users take, that result in the adding, removing and updating of the status of tasks. In particular, for removing a task we take a consensus-based approach. When the removal request for a task 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 task. Recipients can object to the removal. If no objections are received within a specified timeframe, the task is removed. Upon removal, the dependency list, of each task dependent on the removed task is updated to replace the removed task with the tasks in its dependency list. The status of the removed task is set to "in-active", meaning that it is no longer in the task model of the conversation. The removed task 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 next best actions to participants. This component works based on two sources of information: the predefined best practice conversation template, and the task models of similar conversations in the past. Note that the task 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 task 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 task models. This algorithm returns the number of edits (insert, delete and relabeling) needed to transform the task model of the current conversation to one of a past conversation. Conversations relating to the same class of incidents are considered more similar than others. The recommendation approach generates a list of tasks as suggestions for new tasks to be performed in the conversation. The list has two parts: task(s) from the templates and frequently used tasks from past conversations. The number of recommended tasks is configurable. This feature enables reuse of past experiences and provides knowledge for refined best practice templates.

V. Implementation and Use Case Study

We describe the implementation of a prototype realization of the IT support 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 task factory, the task lifecycle manager, notification and the 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 similarly 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 the user an incident to be filed. 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 ITSCM user interface for a participant of the conversation from the help desk. On the left hand side it shows the participant's conversation list and the list of the tasks the user is involved in. On the right hand side, there are the details of a specific conversation with all the views the participant has access to. As a member of the help desk staff, 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 next best 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.

The task inbox window is expandable to enable internal participants to monitor the status and progress of the tasks related to the case, as well as to update their properties. It also allows participants (e.g., Fred) to create a new task 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.

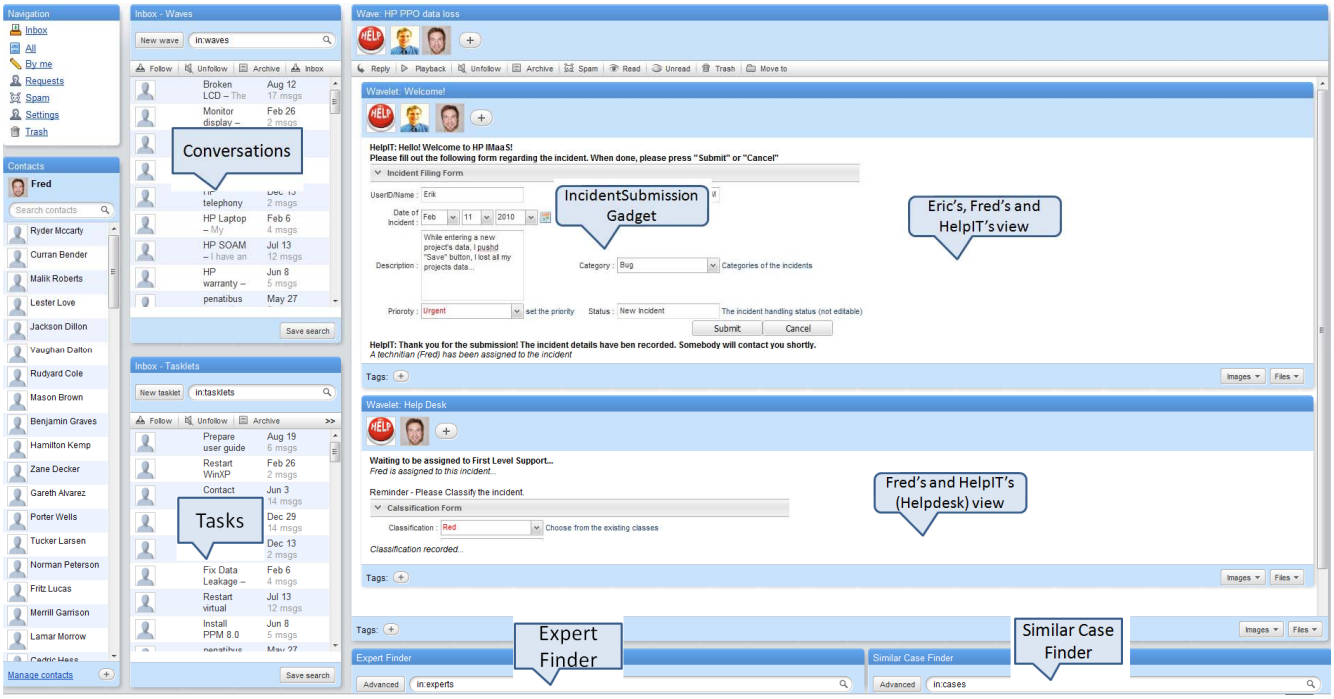

Figure 5. The screenshot of the frontend of the ITSCM prototype for helpdesk

Using ITSCM, submitter (Eric) creates a new conversation by clicking on the "Help IT!" link and selecting "New Incident". This activates the HelpIT robot. HelpIT creates a new view called "Welcome" in which HelpIT and Eric are the only participants in the conversation at this stage. HelpIT greets 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 task templates related to incident management. HelpIT uses the templates during a conversation to guide it, to determine the next task given the current task. The fragment shows four tasks and their dependencies. IncidentManagementProcess is the general TaskTemplate for the overall incident management process.

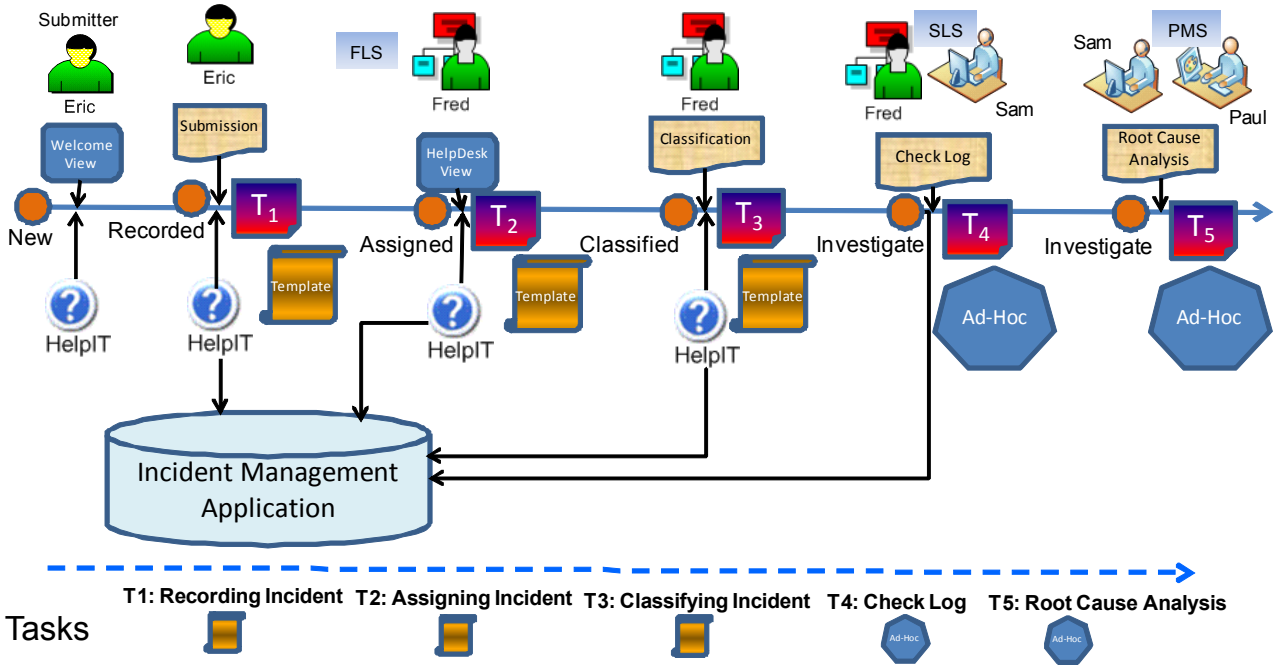

Figure 4. Applying the proposed approach to the incident management example scenario

It is part of ITILOperationalManagement and classified as Required. A number of actors are listed, such as HelpDesk, FLS and SLS. CompositeSteps lists a breakdown of the incident management process into sub-activities, following the ITIL framework. The task templates for the first three steps are shown as T1-T3 in Table 1. The first step is IncidentDetectionAndRecording (T1), initiated by the Submitter and performed by FLS. The second step is the Assignment of incidents (T2), which HelpIT performs, including pushing the new incident into the queue of FLS and updating the status in Eric's view. T2 has a dependency on T1, which is shown by the dependsOn property in its definition. The third step is Classification (T3), which has a dependency on T2. The remaining Task Templates (InvestigationAndDiagnosis, Closure and ResolutionAndRecovery) follow a similar pattern and are not shown in Table 1.

The HelpIT robot is programmed to assist in enacting a number of built-in tasks (part of the templates): such as recording an incident (by providing a form); participating in interactions such as sending welcome messages; or giving notifications of actions performed by participants in the conversation text thread.

For the first task in the template, which is recording the incident, HelpIT performs this task on behalf of FLS. It 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 task in the task model of the conversation assigned to HelpIT, which is represented by the task 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 the incident shortly.

HelpIT then adds a message to the Welcome view indicating that the incident has been recorded. It updates the status of the task T1 to "completed". Accordingly, HelpIT changes the Status field in the IncidentSubmission gadget to "recorded".

The next step is assignment of the incident. As can be seen from the template, this operation needs to be carried out in a view called "HelpDesk", rather than the Welcome view that is visible to Eric. The purpose of the HelpDesk view is to provide isolation between the submitter and the FLS team. HelpIT creates the HelpDesk view and adds a message indicating that the incident is waiting to be assigned to someone in First Level Support (FLS). It also creates a task (T2 in Figure 5) related to the assignment of the case to the FLS queue.

Let us assume that Fred from FLS is assigned to the incident. HelpIT adds Fred to the participants list in the conversation, 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 task 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:TaskTemplate ;
 m1:partOf :ITILOperationalManagement ;
 m1:rType m1:Required ;
 m1:actorRole m1:HelpDesk, m1:FirstLevelSupport, m1:SecondLevelSupport;
 m1:compositeTasks (
  m1:IncidentDetectionAndRecording
  m1:Assignment
  m1:Classification
  m1:InvestigationAndDiagnosis
  m1:ResolutionAndRecovery
  m1:Closure
 ) .


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

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

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

The next task in the task model is classifying the incident. HelpIT adds a message to the HelpDesk view recommending the classification of the incident and, if Fred accepts the recommendation, it adds the Classify gadget to the HelpDesk view. Correspondingly, HelpIT adds a task called "Classifying Incident" (T3 in Figure 5) to the tasks with the status of "assigned" (to Fred as the responsible person). 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 task to "complete".

After reviewing the incident, Fred concludes that the case needs to be referred to Second Level Support (SLS) since it is outside the scope of FLS. Fred creates a new task for investigation and diagnosis, and assigns it to the queue of SLS (not shown explicitly in Figure 5). HelpIT updates the status in the IncidentSubmission gadget to "investigate". Next, Sam from SLS is assigned. Sam reviews the case and mentions in the chat stream that as part of investigation the application log needs to be checked. He checks it and notices that it requires interaction with ABC's operation team (see the scenario in Section II.A). Sam creates a new task called "Check Log" (T4 in Figure 5) by filling in a form in the task creation gadget offered by HelpIT. Sam also adds people from ABC's operation team with the role of "consulted" for this task. Note that this task is not part of the template, and therefore it is treated as an ad-hoc update of the process.

Sam interacts with ABC's operation team in the conversation and concludes that the problem is not with operations, but it is a possible software bug. Sam creates a task called Root Cause Analysis (T5 in Figure 5) and assigns it to the problem management team. The task T5 is routed to the problem management team's queue, where Paul is assigned to this task. Paul reviews the case using the playback feature. He can also look at the list of tasks 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 flexible and adaptive definition and enactment of best practice processes. For tasks T4 and T5, that are added to the process via ad-hoc changes, the system creates T4 as a successor of T3 and T5 as a successor of T4 in the dependency graph. This approach addresses the information loss in the process by encapsulating all the interactions between people regarding the handling an incident, including the interactions related to the hand-offs between teams. It also captures what information (in the forms) changes during which interactions, which is verifiable in the playback.

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. We share the common motivation of enabling dynamic adaptation of processes with the idea of "fluid processes" [23]. In contrast to the works above, we do not expect users to define process models explicitly (as we focus on descriptive processes rather than prescriptive ones, and our users are knowledge workers) but we build and use the process model implicitly to help drive the work, monitor and assist tasks progression, and therefore the refinement and adaptation of the process is performed by the system on behalf of users.

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, 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 task model and by automation assistant components, such as the tasks 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 tasks and provides automation support. 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 domain.

As future work, we are considering enhancing the capabilities of the task 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 next best steps) to participants. As starting points in this direction, we are also in the process of incorporating plugins for a similar case finder and expert finder in the tool, to allow the assistant agent to find similar cases and experts relevant 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. We are also in the process of having people in HP IT 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 for 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] K. Brennan, "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] Y. Choo, "Hierarchical Nets: A Structured Petri Net Approach to Concurrency," Technical Report CaltechCSTR:1982.5044-tr-82, California Institute of Technology, 1982.

[6] S. Graupner, H. R. Motahari-Nezhad, S. Singhal, & S. Basu, "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] Z. Shangguan, Z. Gao, K. Zhu, "Ontology-Based Process Modeling Using eTOM and ITIL," In CONFENIS (2) pp. 1001-1010, 2007.

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

[10] M. Reichert, P. Dadam, "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] T. Eijndhoven, M. Iacob, and M. L. Ponisio, "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] A. Dreiling, Gravity – Collaborative Business Process Modelling within Google Wave, SAP Research, Sep. 2009.

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

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

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

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

[23] M. Reichert, "A Thing Called "Fluid Process" - Beyond Rigidity in Business Process Support," In: 3rd Int'l Workshop on Enterprise Modelling and Information Systems Architectures (EMISA'09), September 2009, Ulm, Germany.