Next Best Step and Expert Recommendation for Collaborative Processes in IT Service Management

Hamid Reza Motahari-Nezhad and Claudio Bartolini

Hewlett Packard Laboratory
1501 Page Mill Rd, Palo Alto, CA, USA
{hamid.motahari,claudio.bartolini}@hp.com

Abstract. IT service management processes are people intensive and collaborative by nature. There is an emerging trend in IT service management applications, moving away from rigid process orchestration to the leveraging of collaboration technologies. An interesting consequence is that staff can collaboratively define customized and ad-hoc step flows, consisting of the sequence of activities necessary to handle each particular case. Capturing and sharing the knowledge of how previous similar cases have been resolved becomes useful in recommending what steps to take and what experts to consult to handle a new case effectively. We present an approach and a tool that analyzes previous IT case resolutions in order to recommend the best next steps to handle a new case, including recommendations on the experts to invite to help with resolution of the case. Our early evaluation results indicate that this approach shows significant improvement for making recommendations over using only process models discovered from log traces.

Keywords: Best practice processes, Collaborative and Conversational Process Definition and Enactment, Step Recommendation, Expert Recommendation.

1 Introduction

IT incident management is one of the key ITIL (IT Infrastructure Library, www.itil-officialsite.com) processes [1]. ITIL includes a number of core processes that provide guidelines on managing the IT ecosystem of an organization. IT service management processes are people intensive and collaborative by nature. Existing IT management tools, including HP Service Manager [14], support ITIL processes by interpreting and implementing a specific interpretation of those processes. However, they do not support people collaboration. A new generation of IT management tools, including our earlier work, the IT Support Conversation Manager (ITSCM) [1], has arisen that relies on collaborative techniques to manage IT support cases. They take a collaborative and semi-structured approach to supporting ITIL processes. In particular, ITSCM enables people collaboration and flexible and adaptive process definition and enactment for IT case management. This enables IT staff to customize and define case management processes, while taking into consideration the specific organization and case information within the guidelines of the ITIL framework.

S. Rinderle-Ma, F. Toumani, and K. Wolf (Eds.): BPM 2011, LNCS 6896, pp. 50–61, 2011.

While collaboration-based IT process management systems advocate flexibility in defining the flow of activities, one issue is that staff may need to manually define the same or similar step flows for similar cases. IT staff may benefit from reusing the wealth of knowledge that is latent in the support organization about the best sequence of activities to follow to resolve a given case, based on learning from past similar case resolutions. One of the key desired functionalities of this new generation of IT case management tools is the ability – for a new case – to provide automated assistance to IT staff involved in case management on how to advance the process more efficiently.

The problem that we focus on in this paper is providing decision support for guiding the resolution of new cases, by providing automated recommendation of the best steps and experts to resolve an open case, based on the knowledge of past similar cases in the new generation of IT Support System that support collaborative and conversational case management. According to statistics from HP Technical Services (TS), HP handles more than 100,000 IT calls per day, on behalf of the customers that it represents. Some of these calls are about similar issues. Even if a small percentage of such cases benefit from automated guidance on activities that lead to quicker and effective resolution, large efficiency gains would result in the overall case management process.

In Section 2, we provide an overview of the collaborative case management advocated by ITSCM, and discuss the challenges of guiding IT staff in such settings. Section 3 presents the solution architecture and algorithmic details. In Section 4, we discuss the implementation and evaluation results. Finally, in Section 5 we discuss related work and conclude with a perspective on future work in Section 6.

2 Automated Guidance in Collaborative Processes: Background and Challenges

In the IT Support Conversation Manager (ITSCM) [1], we introduce the concept of *conversation* as a container for the interaction and collaboration of IT staff following a series of process-oriented actions to handle an open IT case. A conversation includes a set of participants, an informal thread of conversation among participants (textual chat data), and a structured part of conversation including a *step* sequence (or step flow). Each step defines an action. A subset of conversation participants may get involved in fulfilling an action following the RACI (Responsible, Accountable, Consulted, and Informed) model. Fig. 1 shows the data model of a conversation in ITSCM (referred as a *case* in IT case management).

In the following, we describe the most important attributes that are used in the analysis of conversation data and the automated guidance to participants:

- *title* is the title chosen for the IT case by the submitter;
- *tags* which are a set of keywords that are attached to the case by the support staff handling the case;
- *step* represents an individual action;
- *step flow* represents the sequence of steps performed during the course of handling the open case;
- *participant* refers to a member of the support staff involved in the case

handling or the case submitter;

- *type* refers to a particular case type, from a pre-defined classification of types allocated by the support staff;
- *priority* is a category attribute that specifies the level of severity of the issue. The value for this attribute is first identified by the submitter and may be changed later by the support staff;
- *status* which specifies the status of the case, e.g., draft, open, closed, etc.
- *other text attributes* which include participant comments and a case description, that are treated as a bag of keywords in the analysis.

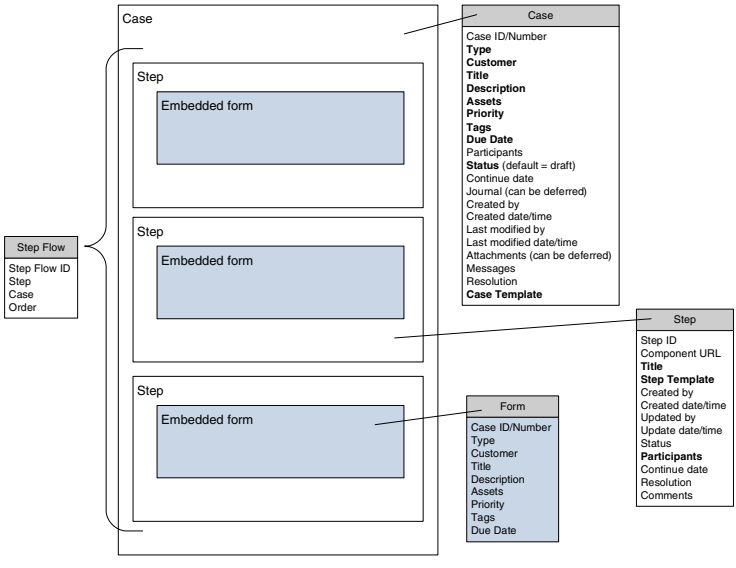

Fig. 1. The case data model in the collaborative IT case management system (ITSCM).

In ITSCM, depending on the type of a case, IT staff from the helpdesk may choose to use a pre-existing step flow *template* for the standard initial task, such as capturing case data and classifying it. Then, ITSCM empowers users to define new steps as the informal discussions among the participants progresses; steps may be assigned to participants and new people may be invited to participate in steps or in the conversation, until the case is resolved. The ITSCM keeps a model of the task flow in the backend, to support the enactment and tracking of steps, to send notifications and to watch due dates/times for escalation purposes.

Challenges. In a collaborative and conversational approach for case management, the IT staff defines custom step flows. One issue is that the same step may be defined/named differently in two different cases. The challenges of automated assistance for recommending next steps include capturing the similarity between custom-defined step flows of different cases. Another challenge is considering the contextual information of the informal conversations before and after a step. A more general challenge is identifying the similarity of the step flows of two cases which

requires taking into consideration the unstructured, textual information that is exchanged among the conversation participants. Finally, there is the need to find matching steps for a given open case and for measures to rank the steps based on an interpretation of the "best" next steps.

3 Next Step and Expert Recommendation

We have developed a solution that recommends best next steps and experts for an open case at hand by: 1) building a model of step flows of past cases annotated with metadata from the threads of conversations from informal interactions, and 2) matching the current flow of activities in the open case with the annotated step flow model to recommend the next best step(s), and experts.

The solution architecture. Fig. 2 shows the architecture of our solution. The repository contains the historical case information. It has two main components: an *annotated step-flow-model discovery* component that analyzes the previous cases present in a repository and builds a *step flow model* in which each step is annotated with the metadata information of previous cases that took that particular step. In order to make models more manageable in terms of size and easier to understand, we have decided to discover a separate step flow model for each case type. The *step recommendation* component uses a discovered step flow model corresponding to the type of the open case and takes into account the information of the open case to find the best match(es) of paths in the model to that of the open case, and returns to the user an ordered list of the recommended step(s). Along with each step recommendation, the solution also recommends experts: IT support staff who have performed the recommended step during the resolution of a past case.

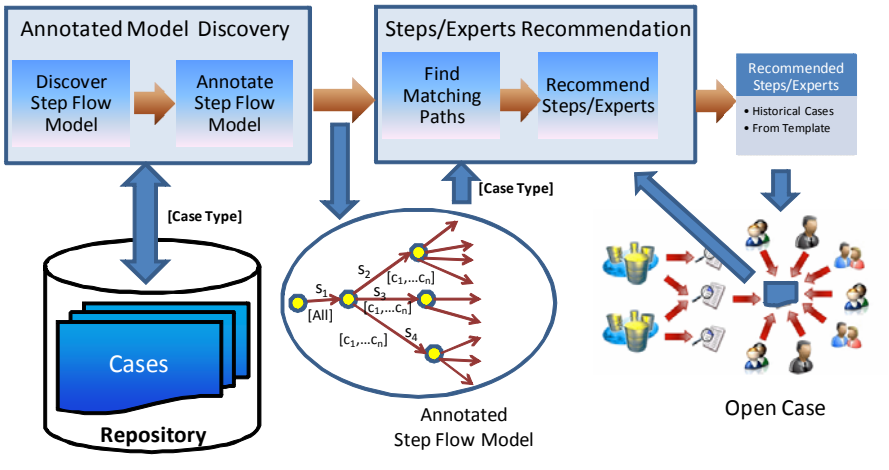

Fig. 2. The component architecture of the Next Best Steps/Expert Recommender solution

3.1 The Model Discovery Component

This component analyzes the existing cases in the repository and creates an annotated process model in two phases: (i) discovering the step flow model and (ii) annotating the process model.

Step flow model discovery. We have designed an algorithm that reads the step flows for the cases of a given type from the repository and discovers a model. The first decision we needed to make about the algorithm wasthe language (with formal and visual definitions) to represent the discovered step flow model. We chose finite state machines (FSM) for representing step flow models in this work. FSMs are a well-known formalism, easy to understand and visualize, and suitable for modeling reactive behaviors. There are a number of methods and tools that enable analysis and management of step flow models specified in FSMs too. Each transition in the model is labeled with a step name. By traversing this model from its start state, we can re-produce individual step sequences of the past cases that are used in the discovery phase. We build on top of our previous work [2] for discovering the process model from process event logs for this purpose.

The flow discovery algorithm is a hybrid of grammar inference and probabilistic approaches. The algorithm uses the Markov model to identify step sequences based on statistics about steps that frequently follow each other, and to build an FSM from these sequences. We build the step flow graph as follows: one vertex is created for each distinct step in any step flow in the repository. For a given order n (Markov order), and for each sequence of length n + 1, uniquely labeled edges will connect each step (which is now a vertex) in the sequence to the immediately following step (vertex) in the graph. The resulting graph is converted to an FSM [2, 3].

The FSM generated by the previous step may contain equivalent states that could be reduced. To reduce these states, we merge them. The following criteria are applied to find candidate states for merging: i) states with the same outgoing transitions, i.e., transitions with the same steps to the same target states; ii) final states with transitions labeled with the same incoming step name; and iii) states with the same outgoing

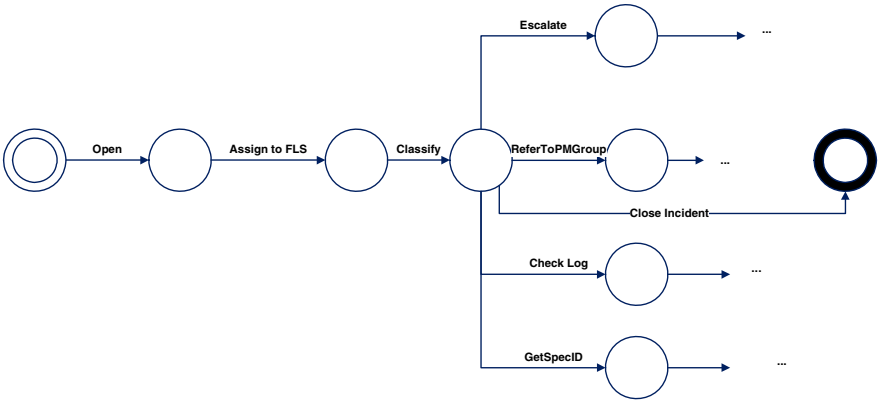

Fig. 3. An example step flow model for an incident management case repository

transitions, excluding the transition(s) that goes to the other state(s) to be merged with it. In this case, preexisting transitions between merged states are represented as self-transitions on the newly created state. The resulting model is an FSM in which each transition is annotated with statistical metadata. Each number on a transition is normalized into the range of zero to one denoting the probability that the current transition is taken among the outgoing transitions of the source state.

Annotating the step flow model with case information. The discovered process model in the previous phase is an FSM graph that shows all possible steps taken after a particular step, annotated only with statistical metadata but not contextual information on the cases. In a second phase, the algorithm annotates each transition in the process model with information about the conditions under which a transition representing a step is taken in previous cases.

For each transition in the FSM, which is labeled with a step name 'a', we build an index of the cases that contain 'a' in the same order that it is observed in the FSM. For instance, in Fig. 3 all the cases that take the "Escalate" step right after "Classify" are identified. We extract information from case title, tags, priority, description, status and the step title, description and participants. We then summarize information of each attribute from the set of cases that include a given step in the model. For categoryattributes such as priority and case status, we record all values seen in the cases and their frequencies.

For text attributes, we employ an information retrieval technique to extract keywords that uniquely identify the attribute by focusing on proper names and domain-specific terms (after removing all the stop words). In this approach, besides regular keywords, we extract semantic associations between specific words and identify words as names of companies, people or locations. In IT cases, names of locations and companies (such as HP and Microsoft) are useful for unique identification of a step. The set of keywords are kept along with their frequency as part of the step metadata. Finally, we also keep a list of the support staff who have carried out the step represented by a transition. We use this information to annotate the transition representing the step. The annotation is in the form of a set of key value pairs, where the keys are the attributes and the values specify the frequent values (e.g., keywords, category values) observed for the attribute, along with statistical data.

Note that the discovery and annotation of the step flow models is performed offline. The frequency of the update of the model is configurable, with a default setting of 12 hours for updating the models with the step flow of new cases that have not previously been considered in the creation of the model. This approach incrementally updates the model rather than re-building it every time. The system is configurable to take cases between a specified date range, e.g. past 3 days only.

3.2 The Steps/Experts Recommendation Component

This component uses the step flow information of an open case and the step flow model of the same type as that of the open case to recommend the next steps. Next steps recommendations are then presented to the user, who may either accept or ignore them. In order to recommend the next steps, we find the step in the flow model that corresponds to the last step in the open case. In some situations, there may

be more than one step in the model that matches the current step in the open case, based on the similarity between the case information and that of the metadata attached to the steps in the model.

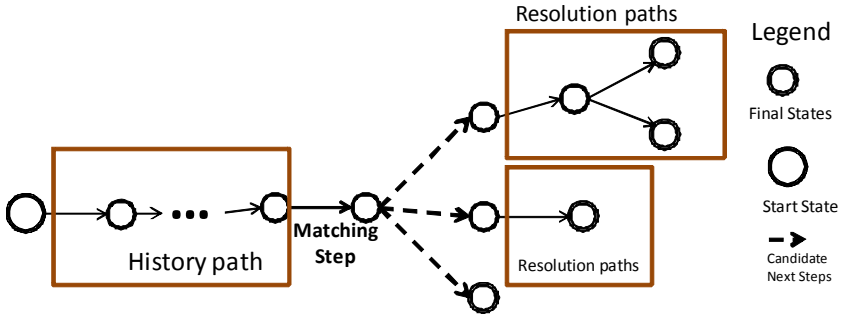

Fig. 4. The step recommendation algorithm considers both the path leading to a matching step (history path) and paths leading to a possible resolution in making recommendations

To identify the matching steps, the algorithm considers the path leading to a matching step candidate; looking for a match with the path to the current step in the open case. The recommendation algorithm then considers the resolution paths (the paths from a candidate next step to final states with resolutions) in order to make a recommendation (see Fig. 4 -- The figure shows the history path for one matching step and resolution paths for its candidate next steps in the model). In this sense, the algorithm employs an Occam's razor argument (http://en.wikipedia.org/wiki/Occam's_razor) in giving a higher priority to steps that lead to a resolution in fewer steps and to steps that match the context of the current open case.

Step Matching. The step matching method incorporates the approximate sequence matching between the step sequence of the open case and the paths in the process model where the candidate steps are located. Since the step flow in the open case is not complete, we only consider paths of the same length or at most three more steps in the model, counting from the Start state of the FSM. We call such paths candidate paths.

The algorithm first computes the similarity between case information in the current step and metadata of the steps in the model (the steps in the candidate paths), without considering their location in the path. The following is the measure of similarity between steps i of the case (denoted as C_i) and j in the step flow model (denoted as M_j):

$$StepSimilarity(C_i,M_j) = titleSimilarity * titleCE + tagsSimilarity * tagsCE + descriptionSimilarity * descriptionCE + priorityDimilarity * priorityCE;$$

In this function, *titleCE, tagsCE, descriptionCE and priorityCE* are coefficients that specify weights of different attributes. These coefficients sum to One. Priority measure defines a filter so that steps from cases with at most one priority level

difference, over a 1-to-5 scale, are considered. For example, if the priority of the ongoing case is 2, cases with priorities of 1, 2 and 3 are considered.

Then, we choose the most similar steps in the model to that of the last step in the case (top 5 in practice). For each of these steps, we compute the similarity of the path leading to the step in the model with the path leading to the last step in the step flow of the open case. The step flow similarity is computed as follows:

$$flowSimilarity(lastStep\_C_i, matchingStep\_M_j) = StepSimilarity(lastStep\_C_i, matchingStep\_M_j) + pathSimilarity(1..i, 1..j);$$

in which

$$pathSimilarity(1..i,1..j) = \sum_{1 \le l \le k} StepSimilarity(C_l, matchingStep\_M_l), k = i-1, i \le j; k = j-1, j \le i$$

This equation computes the similarity of pair-wise steps in the path immediately before *lastStep* in the case and the *matchingStep* in the model. This equation considers the position of steps in the paths and whether steps are similar. As a result, when compared to the last step in the open case, the steps in the model with a more similar history of actions are ranked higher. Note that for recommending next steps, we give preference to steps that lead to a resolution in a fewer number of steps, as discussed below.

Step Recommendation. The step recommendation builds on and extends the approach for finding matching steps in the model. The algorithm finds the next steps immediately after the matching steps and ranks them according to their potential match to the context information of the open case, and the likelihood that they lead to a faster resolution based on information from past cases. We define a matching measure of a step C_i for next steps for recommendation as follow:

$$nextStepMatch(C_i) = flowSimilarity(lastStep\_C_i, matchingStep\_M_j) + stepCaseMatch(C_i) \times resolutionPath(M_j);$$

in which

$$stepCaseMatch(C_i) = tagsSimilarity * tagsCE + descriptionSimilarity * descriptionCE + priorityPathSimilarity * priorityCE;$$

and

$$resolutionPath(M_j) = \begin{cases} 1 & resPathLen = 0 \\ Max\{(resPathLen^{-1} \times \sum_{1 \le k \le resPathLen} stepSimilarity(C, M_k)), \forall path \in resPath\} & resPathLen \neq 0 \end{cases}$$

The *resolutionPath* function favors a candidate next step with a resolution path of smaller length and higher similarity. The *stepSimilarity* in this function computes the sum of the similarities of individual steps in the path to the metadata of the case. For each candidate next step, it returns the similarity of the best match to be multiplied by the *stepCaseMatch* similarity value.

We then rank the next steps based on the value of *nextStepMatch* and return them to the user interface to be presented to the user. Along with each next best step, the recommender also suggests experts – IT staff who previously carried out the

recommended step on similar cases. The list of IT staff is sorted, based on the frequency of handling this particular type of case, and the relevance of their profile to the open case, where available. The textual similarity is calculated with a cosine similarity function, which is a well known technique for comparing vectors that are represented by keyword sets [13].

Step and model standardization. Using a standardized approach for naming steps improves the quality of step recommendation. However, often no standardized names are adopted. To compensate for this fact, we support storing previously used names for steps and recommending them for reuse in naming new tasks. In the IT Support Conversation Manager [1] we add all newly added steps of a conversation to a step library for that conversation type. When a conversation participant adds a new step, we provide a search box over past names and an auto-complete feature.

Nevertheless, the step flow model of a conversation type may contain steps that are similar to each other but use different step names. We analyze the model for each conversation type to identify similar steps. We present them to a case domain expert for decision making on whether they are considered the same step, therefore enabling them to be merged to build a more concise and compact model.

4 Implementation and Experiments

We have created a proof-of-concept prototype of the next best step recommender solution, in connection with the 48Upper.com project within the context of an HP R&D project on next generation IT Service Management solutions that adopt a collaborative approach for ITIL processes. Our prototype is implemented in Java, on top of Eclipse. Fig. 5 shows a screenshot of the recommendation solution integrated with the IT Conversation Manager [1].

To evaluate the approach, we used a set of case data from simulation and from early trials of the tool (more than 250 cases, all related to issues in a human resource software application and incident management). We evaluated the next best step recommender solution in two modes: using information on the sequence of activities alone, and using the information on the sequence of activities combined with the case's informal thread of interactions among participants. The goal of this experiment is to validate the significance of the case contextual information, as well as that of the approximate matching of step flows with respect to the quality of recommendations made purely based on a process model that includes prescribed action flows but is not augmented with contextual case information.

The results of the evaluation show that using only process model information leads to a lower quality of recommendations in terms of ranking the steps, as the algorithm can only rank the steps based on the frequency of the step usage. However, when case information such as case type, title, tags and contextual information on the people interactions are considered, the recommended steps are more likely to be relevant to the context of the case. Based on the 250 cases in the dataset (4 types of cases), we used the recommendation approach on 20 cases. The number of times that users found any of the recommendations useful when using only a process model (structural information) augmented with statistics was 7 times (35%); whereas recommendations made using the presented approach was considered useful in 16 cases (80%). This

initial evaluation is an indicator that the step recommendation approach is effective in finding relevant cases for IT cases with adaptive and collaboratively defined processes for case resolution.

In terms of efficiency and performance, this approach has a very important advantage in pre-building a model of the step flows of past cases in offline mode. At runtime, we only need to perform in-memory operations to find the recommended steps. In our experiments, the results are computed in the order of seconds.

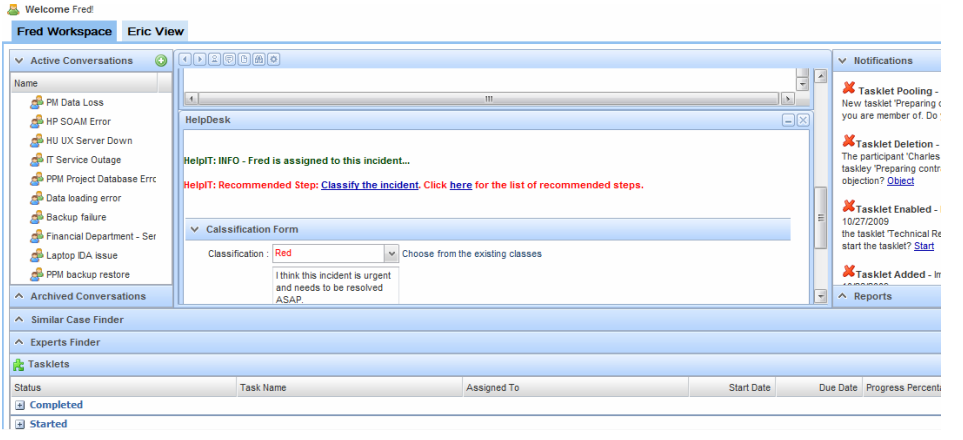

Fig. 5. A screenshot of IT Conversation Manager, where the next best step recommender is integrated

5 Related Work

To the best of our knowledge, no other tools exist today that address the problem of recommending activity steps in collaborative IT support systems. The work presented in this paper advances our collaborative approach for process definition and enactment introduced in the IT Support Conversation Manager [1].

The process mining techniques [2, 3] enable the discovery of a process model from a set of process instances of ill-documented business processes, and understand changes in process execution. These techniques discover the structural model of the process. In this paper, we have introduced a method for the discovery of an annotated process model with contextual process information. Our model discovery component leverages text analytics and process mining techniques to discover the annotated process model and associated experts. We use the discovered annotated process model to make next step and expert recommendations as an open case progresses.

Step recommendation for business processes has been investigated in [4, 5]. In particular, [4] proposes a step recommendation approach for an existing process, based on past process execution data. The authors in [5] propose an approach for learning the process models used among office workers and using that knowledge to recommend process tasks to other colleagues. However, these approaches only consider the structure of the process model as a basis for recommendation. Our experiments in the

IT case management domain show that using an annotated model significantly improves the quality of recommendations. The authors in [15][16] propose a complementary approach to our work by constructing decision trees based on the structured data attributes available in a process log for each decision point of the process. [16] uses this information to make predications on the future route of a process. In this paper, we use both structured and unstructured information to annotate steps in the model, and use a similarity-based approach for making step recommendations for collaborative, ad-hoc processes in which no standardized naming scheme for steps is used.

The problem of routing incident tickets to the right team/individual to handle it is investigated in [6, 7, 8]. In this line of work, the process model captures the routes in which tickets circulate among experts on the help desk and in other supporting offices. They consider the structure of this routing graph [6, 7] as well as the textual information of the ticket description [8] in making group/expert recommendations. However, they do not investigate the problem of recommending next process steps on how to handle a case.

Prior work on workflow adaptation in [10, 11, 12] proposes workflow adaptation techniques that use observations on past workflow variants to inform a given new workflow execution case. These approaches find prior change cases that are similar to a new workflow change request, to enable the reuse of previous solutions in similar new cases. However, there is an assumption that there is a pre-defined workflow model whose execution may vary depending on the situation.

6 Conclusions and Future Work

To the best of our knowledge, this is the first approach that addresses the problem of recommending next steps and experts for IT case management systems. The main components of the system – the automated discovery of annotated models and the recommender – introduce novel techniques that advance the state of the art in next step and expert recommendation, by employing information on conversational and social interactions among people.

In particular, the IT incident cases are richer in their attribute sets (consisting of text, categoryand structured step flows) compared to workflows (that consider metadata such as frequencies, time, etc). In case step flows, the same/similar steps may be defined differently across different cases; and the case step flow for an open case is built gradually (no predefined workflows exist). The convergence of information retrieval techniques and a novel process mining technique in our case addresses the next step and expert recommendation problem given the conversational and collaborative nature of step flow definition.

As future work, we plan to incorporate users' feedback on recommended next steps to improve recommendation results. We also plan to expand the scope of this solution in a Software-as-a-Service (SaaS) environment. In particular, we want to exploit the fact that multi-tenant implementation of case management tools offers opportunities for analyzing activities across different tenants, thereby improving our suggestions.

References

1. Motahari-Nezhad, H.R., Bartolini, C., Graupner, S., Singhal, S., Spence, S.: IT Support Conversation Manager: A Conversation- Centered Approach and Tool for Managing Best Practice IT Processes. In: The Proc. of 14th IEEE EDOC (EDOC 2010) (October 25-29, 2010)
2. Motahari-Nezhad, H.R., Saint-Paul, R., Benatallah, B., Casati, F.: Protocol Discovery from Imperfect Service Interaction Logs. In: ICDE 2007, pp. 1405–1409 (2007)
3. Van der Aalst, W.M.P., Van Dongen, B.F., Herbst, J., Maruster, L., Schimm, G., Weijters, A.J.: Workflow mining: a survey of issues and approaches. Data Knowl. Eng. 47(2), 237–267 (2003)
4. Schonenberg, H., Weber, B., van Dongen, B.F., van der Aalst, W.: Supporting Flexible Processes through Recommendations Based on History. In: Dumas, M., Reichert, M., Shan, M.-C. (eds.) BPM 2008. LNCS, vol. 5240, pp. 51–66. Springer, Heidelberg (2008)
5. Dorn, C., Burkhart, T., Werth, D., Dustdar, S.: Self-adjusting recommendations for people-driven ad-hoc processes. In: Hull, R., Mendling, J., Tai, S. (eds.) BPM 2010. LNCS, vol. 6336, pp. 327–342. Springer, Heidelberg (2010)
6. Shao, Q., Chen, Y., Tao, S., Yan, X., Anerousis, N.: Efficient ticket routing by resolution sequence mining. In: KDD 2008, pp. 605–613 (2008)
7. Miao, G., Moser, L., Yan, X., Tao, S., Chen, Y., Anerousis, N.: Generative Models for Ticket Resolution in Expert Networks. In: Proceedings KDD 2010 (2010)
8. Sun, P., Tao, S., Yan, X., Anerousis, N., Chen, Y.: Content-Aware Resolution Sequence Mining for Ticket Routing. In: Hull, R., Mendling, J., Tai, S. (eds.) BPM 2010. LNCS, vol. 6336, pp. 243–259. Springer, Heidelberg (2010)
9. Kang, Y.-B., Zaslavsky, A.B., Krishnaswamy, S., Bartolini, C.: A knowledge-rich similarity measure for improving IT incident resolution process. In: SAC 2010, pp. 1781–1788 (2010)
10. Weber, B., Reichert, M., Rinderle-Ma, S., Wild, W.: Providing Integrated Life Cycle Support in Process-Aware Information Systems. Int. J. Cooperative Inf. Syst. 18(1), 115–165 (2009)
11. Lu, R., Sadiq, S.W., Governatori, G.: On managing business processes variants. Data Knowl. Eng. 68(7), 642–664 (2009)
12. Madhusudan, T., Leon Zhao, J., Marshall, B.: A Case-based Reasoning Framework for Workflow Model Management. Data and Knowledge Engineering: Special Issue on Business Process Management 50(1), 87–115 (2004)
13. Navarro, G.: A guided tour to approximate string matching. ACM Computing Surveys 33(1), 31–88 (2001)
14. HP BTO Software, HP Service Manager, `http://www.managementsoftware.hp.com`
15. Lakshmanan, G.T., Duan, S., Keyser, P.T., Curbera, F., Khalaf, R.: A Heuristic Approach for Making Predictions for Semi-Structured Case Oriented Business Processes. Business Process Management Workshops, pp.640–651 (2010)
16. Rozinat, A., van der Aalst, W.M.P.: Decision Mining in ProM. In: Dustdar, S., Fiadeiro, J.L., Sheth, A.P. (eds.) BPM 2006. LNCS, vol. 4102, pp. 420–425. Springer, Heidelberg (2006)