Service for Crowd-Driven Gathering of Non-Discoverable Knowledge

Jim Laredo[1], Maja Vukovic[1], and Sriram Rajagopal[2]

[1] IBM T.J. Watson Research Center, Hawthorne, NY 10128, USA
[2] IBM India Pvt. Ltd., Chennai - 600 089, India
{laredoj,maja}@us.ibm.com, srirraja@in.ibm.com

Abstract. Knowledge that cannot be discovered through automated methods, such as user practices, remains in informal mediums. It is unstructured, and in collective possession of the experts, yet it is key for business insights. Typically this "Non-Discoverable knowledge" is gathered in semi-automated way, which at best provides crude estimates, and doesn't scale. In this paper, we describe our novel approach to rapidly design a process solution for a family of business objects, gathering required knowledge through the use of social networking to identify the experts. We propose a "Deconstructed Survey" that captures the knowledge request, and manages its lifecycle through task forwarding and sub-tasking. We developed the system BizRay, instantiating the proposed approach as a general-purpose, self-service Web-based, crowdsourcing service. We demonstrate its effectiveness in accelerating knowledge discovery, through our experiences with deployments for IT Optimization and Services Delivery.

Keywords: crowdsourcing service, knowledge discovery, people intensive, business network, social network, exception handling, rapid design.

1 Introduction

The biggest strength of an enterprise is its people, specialists who understand and drive the business and the IT. What is the best way to tap into this knowledge and organize powerful knowledge networks to help ease and accelerate important initiatives within an organization? If experts are recruited beforehand as when writing a book, it is possible to subdivide the writing and assign sub-topics to each one. In this case it is possible to identify the people beforehand and subdivide the task at hand. A different type of knowledge capture is required when a business process has taken place, or time has passed and the subject has evolved through time. Experts that contributed to the business activity gained some knowledge that may or may not have recorded it for others to learn. Systems involved in the process management likely have recorded, through logs, the events that took place, even captured the interactions people may have had with the systems. This information is readily available and many, such as Li, *et.al.*, [1], van der Aalst, *et.al.*,[2], Neumann and Robeller [3] have explained techniques to process these logs for the sake of process understanding, troubleshooting or compliance auditing. For those portions of the process where a system was not involved, such as the process to reach a decision, or the very knowledge to use or access the system, the knowledge most of the time remains within the participants and is never extracted. Some unstructured forms like documentation, spreadsheets, or even wikis may capture some of this information but usually requires a proactive approach to store it and to retrieve it. We call "non-discoverable information" the one that the system cannot capture, lacks consistent structure and remains with those individuals who were involved in the process.

Non-Discoverable information is pervasive in domains such as IT Services Management, and traditional communications methods have been used, such as emails containing pre-formatted forms or spreadsheets, to capture it. A small team initiates emails for each knowledge request to be fulfilled, waits for responses, captures replies, assesses completeness and follows up if needed. In addition a great deal of time and corresponding resources needs to be dedicated to exception handling. The most common exceptions are: 1) *Non-response*: they need to be followed up by reminders or management escalations; 2) *Incorrect target:* when the expert has moved on to another role and has forgotten or has no access to current information; 3) *Unavailability:* when targeted expert may be away or no longer works in the organization. 4) *Requests for more information:* in this case the recipient of the request is not clear on how to proceed and usually replies back with questions. Traditional approaches can work when the number of initial requests is small but have difficulty scaling to larger numbers. In our initial study [4], we employed crowdsourcing to engage experts to capture compliance and application-hosting specifications. Previously a small team of 2 people was able to handle 140 requests in a 2-month period, which was acceptable. They created a spreadsheet and through email notified potential stakeholders that could source the information, they correlated emails and acted based on the responses received. However, at that pace, for the target of 4500 collections it would have required more than 64 months to complete. Instead our approach for the overall target took 10 weeks, with close to 90% completion achieved in just 5 weeks, and 50% completion in only 4 days.

In this paper, we describe how we extended a custom web application to a general-purpose, Web-based, enterprise crowdsourcing self-service expediting design and delivery of crowdsourcing campaigns. To achieve that we have designed a concept of a deconstructed survey, enabling request delegation and sub-tasking.

The key contributions of our proposed approach and its realization as a crowdsourcing service, called BizRay, are:

1. Model for deconstructed survey: Consists of the underlying artifacts for defining a survey and capturing and managing for each instance the responses and flows in the form of tasks, similar to a worklist in a workflow system [5]. The model facilitates delegation of requests and their subtasking to others. It generalizes our approach and accelerates deployment of crowdsourcing campaigns.
2. Centralized capture of the knowledge: the system implementation provides a central repository that stores and organizes all the responses, which eliminates

labor-intensive email correlation and transcription of the results received via e-mail and instant messaging.

3. Social network features: More than one user can complete each task. If the information gathered is partial or unknown, the user has the choice to forward the task to another party of their choosing and request their assistance. As people contribute their knowledge, the system keeps track of their identity resulting in the formation of micro communities around the object of that inquiry.
4. General purpose and self-service: Generalization and self-service enablement has significantly shortened the design time to be able to accommodate the different crowdsourcing campaigns, often in less than a day; including interoperability with other systems.

The paper is structured as follows. Section 2 reviews related work in crowdsourcing. Section 3 provides an overview of the proposed approach. Section 4 describes data model, architecture and implementation specifications of the BizRay service that realizes the proposed model. Section 5 presents our experiences from the deployment of the proposed model. Finally Section 6 concludes and outlines future work.

2 Related Work

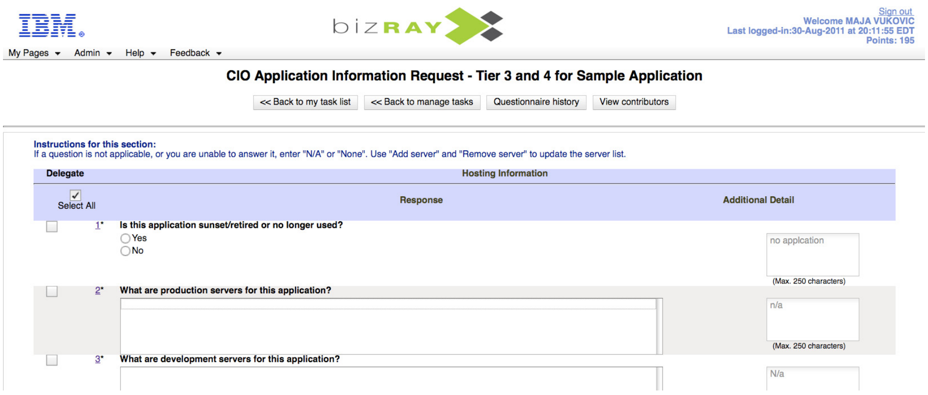
Early examples of crowd engagement can be found in the marketing domain, where companies have run competitions for end-users to contribute towards certain enterprise functions, such as advertising campaign design or specific problem resolution challenges. Xerox's Eureka system [6] is an example of knowledge harvesting system within the scope of the enterprise. ReferralWeb [7] allows for content co-creation, enabling end-users to become aware of their existing communities that are generated by data mining of Web documents and forums. Wikipedia [8] relies on motivating humans to share information and by that either gain creditability or obtain the equivalent information. Amazon's Mechanical Turk[9] is a platform for micro-tasks (e.g. image tagging, classification and transcription).

SurveyMonkey [10] and AdobeFormsCentral[11] are existing commercial solutions, which enable Web-hosted, customizable surveys enabling opinion from end-users. They do not provide mechanism for survey delegation and sub-tasking, which are the core contributions of our work. Facebook Poll application allows for surveys within social network, however it is dependent on the existing network, whereas in our approach network is iteratively built as survey is being propagated.

3 Deconstructed Survey: Capturing Non-Discoverable Knowledge

Knowledge gathering is a fragmented process, it requires hopping from source to source and inquiring at each step for the missing elements. It is important to know who the potential sources might be or at least know someone that may know where to inquire next. If the source is not known, the possibility of crowdsourcing with an open call, as described by Howe [12] is an option. Depending on the setting crowdsourcing may or may not appeal the crowd that may be targeted. For example when looking for information on a topic, the setting of a forum with people that share interest in that topic may prove successful. The open call replaces the need to search for the expert, and anyone available may answer the inquiry. However, when seeking information on a specific object, the potential community that may respond successfully is much smaller and may not be necessarily available for such inquiries. In this case it is easier to establish a chain or several chains of inquiries until the necessary knowledge is captured. As a result two challenges surface: 1) how to identify the next contributor(s); and 2) how to collect fragmented knowledge.

In this section we present our deconstructed survey model and address exceptions listed in Section 1. Figure 1 shows a sample survey, automatically generated from a given template by our BizRay system, described in Section 4. This figure shows a subset of the questions about an asset, in this case business application.



Sign out
Welcome MAJA VUKOVIC
Last logged-in:30-Aug-2011 at 20:11:55 EDT
Points: 195

CIO Application Information Request - Tier 3 and 4 for Sample Application

Instructions for this section:
If a question is not applicable, or you are unable to answer it, enter "N/A" or "None". Use "Add server" and "Remove server" to update the server list.

Delegate		Hosting Information	
✓ Select All		Response	Additional Detail
☐	1*	Is this application sunset/retired or no longer used? ◯ Yes ◯ No	no applcation (Max. 250 characters)
☐	2*	What are production servers for this application?	n/a (Max. 250 characters)
☐	3*	What are development servers for this application?	N/a

Fig. 1. Sample Survey

There are two main data artifacts that we handle and for which we define a lifecycle: 1) the survey, or knowledge that we want to capture and 2) the task that can be associated with a survey. Note that there is a *0 to n* relationship between the two. For a given survey, there can be 0 or more tasks open for knowledge gathering.

3.1 Surveys

In the first version of our system [4], the knowledge gathering information was custom to the use case, and based on a custom User Interface. The design and implementation iteration proved too lengthy and it became a necessity to have a quick mechanism to facilitate the definition of the use case. A form based approach, allowing for quick design was implemented. We created typical data collection fields.

The notion of a survey was a natural representation of the knowledge capture, it allows for quick design, and allows structuring the data elements that are being gathered. It is likely that a question may not be addressed as specified, as answers received may not be part of the original choices. For that, an optional comment field allows for the "write-in" opinions in those questions that disambiguation is required.

Our surveys are not about someone's opinion but rather the knowledge around a subject or object, a topic or an asset. We do not expect one person to be the sole contributor but rather a team of specialists. A fraction of a survey is a survey, that is, it allows us to break down a survey in parts and treat the resulting parts as surveys on their own, this creates a natural recursion to break down and reconstruct the parts of a survey as we request and address contributions from other parties. A survey artifact is mapped directly to the user interface perspective; it is divided in sections and for each section multiple questions are possible. A question is of a given type, based on the type of an answer expected, and represented in the user interface as described above. An *isMandatory* attribute determines if a question is mandatory or optional. As the name of the attribute indicates, only mandatory questions need to be completed to close a survey.

It is possible that there may be more than one answer for a given question. It is important to timestamp and track the person that provided each answer. This history will present the chronology of answers, with the latest response being the current one. More importantly the history trail forms a small community around each question, subsection or the overall survey. At the end of the survey, a group of people that are knowledgeable about the subject that was being investigated has been captured. This micro-community could be applied to either validate any data entry or perhaps gather further details on the subject in question or a related subject.

3.2 Tasks

A task is a unit of work associated with a portion of the survey. When a survey is launched, the top task is created. The user has several choices to manage the task:

1. Complete the survey. Answer all mandatory questions to the best of his ability.
2. Forward the task to another expert. As a result the responsibility of the survey is transferred to someone else.
3. Invite others. Segment the survey and invite other experts to contribute to the selected questions. A new task is created for each invitation generated, the current parent task remains open until all the mandatory questions associated with the survey are addressed.

Once user accepts the task he has the same set of choices:

1. Complete the questions associated with that task. If all mandatory questions are completed the task is closed.
2. Forward the task to a more knowledgeable party. The responsibility of the tasks is transferred to someone else.

3. Invite others. Segment the task and invite other parties to contribute to the corresponding sections. A new child task is created for each invitation generated, the current parent task remains open until all the mandatory questions associated with the parent task are addressed.

In essence a task tree is created. Figure 2 shows an example, where T0 is the task associated with the whole Survey and subsequent tasks capture a subset of the parent task questions, for example, T4 will include a number of questions that are also part of T2 that belong to the survey that T0 is managing.

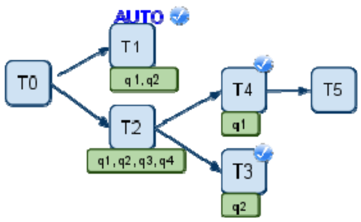


Fig. 2. Task Completion Sequence

Accepting Tasks

There are two basic modes of accepting a task, by assignment and by selection. By assignment occurs when the task is pre-assigned by a system or another person. By selection occurs when a person chooses a task that he is able to manage and proceeds to complete according to the steps described above. In the former case the notion of a network (business or social) is leveraged to identify potential users that can assist in completing the task. In the latter case, the task is available as an open call ready to be crowdsourced, anyone willing or meeting required criteria may accept the task and proceed to handle it. Whoever accepts a task we call the Task Owner. We introduce the notion of rejecting a task to allow for those cases where the assignment of the task has been performed incorrectly in that case the task is returned to the previous owner, if the task was just created it is assigned to the parent task owner.

Completing Tasks

For a task to be completed, it is sufficient that all its mandatory questions be completed. Since there are no constraints on the questions that form the tasks, other than being a subset of the parent task, a question may find its way to other tasks and as such once completed in one task it will contribute to the completion of those tasks that also have that question. Figure 2 further illustrates the task completion process. Assume that T1 has questions q1 and q2, and T2 has questions q1, q2, q3, and q4, and T3 has questions q2, and T4 has questions q1, all questions mandatory. When T4 completes, it means q1 has been answered. This is not sufficient to complete T1 because q2 is also required for T1 to complete. When T3 completes, it means that q2 has been answered. At this point, T1 will also complete, as both q1 and q2 have been answered. The task will be automatically removed from T1's task list.

There are several reasons to adopt this approach. First, it accelerates the completion of the survey. In essence the asynchronous completion of a task prunes the whole sub tree underneath it, basically all tasks complete by transitivity as they only have a subset of the completed questions. Second, although multiple answers may help validate or identify additional detail, this approach helps identify willing and strong knowledge sources. In the example above T3 and T4 task owners together are likely to be more willing than T1's task owner. This information can be applied by T0 to decide to target additional questions to the willing users. Finally, it allows creating a notion of collaboration, two or more tasks can be created with similar set of questions, and assigned to different owners. Each owner is now contributing to a common effort and as the questions get answered it is possible to share the knowledge and allow shifting focus to those questions that still require attention.

When a task completes, the parent task owner is notified. This allows for the parent task owner to track progress, and allow him to decide if he concurs with the answers provided, and to pay attention to the questions that are still open. For tasks owners for which their task was automatically closed, they will get notified if they had opened already the task, otherwise the task will simply be removed from their task list.

It is possible for administrators to cancel tasks in progress, reopen canceled/ completed tasks and transfer tasks to other users. Administrators can search for tasks based on the user it is assigned to, date of assignment and state of the task. Administrators can close surveys that are no longer required which in turn cancels all pending tasks. Administrative tasks come in handy when handling exceptions that have not been considered.

3.3 Exception Handling

The ability to forward a task and reassign is a necessary mechanism to deal with leaf nodes in the tree that would otherwise end in a dead end. The task may have reached a user by accident by incorrectly typing their name or address. Either the task is rejected and reassigned to the parent, or someone that at least may be familiar with a more knowledgeable party has the chance to forward the request to that party.

When dealing with unresponsive users, but possibly knowledgeable it is necessary to send reminders and escalations, manually and automatically, for example, reminders can be sent for tasks beyond a certain age (in days). Reminder intervals can be set as a function of the age of the task, and level of escalations can be determined based on the number of reminders that have taken place.

The history of responses provided by users for each question is recorded. This provides information about who, when and what information was provided for each question. The history information, which includes details of the user and timestamp, can be viewed at the task level and also at each question level. We have used this information to understand the relationships amongst those that forward or create sub tasks and to whom they engage in their knowledge capture. In some cases this information can point us to other potential contributor, we have studied this problem in part in [13].

4 System Architecture

This section describes the BizRay system, which instantiates the proposed framework for social gathering of distributed knowledge gathering. We describe the service components and how their interactions realize a self-service Web-enabled, software as a service enterprise crowdsourcing service that allows us to expedite design and delivery of crowdsourcing campaigns. We describe the internal representation of knowledge in BizRay and the technologies that were used to implement, deploy and host this system.

4.1 Knowledge Representation

BizRay employs artifact centric approach to modeling business objects and processes upon them. Figure 3 shows the set of data entities that capture survey, its questions, answers and relationship to the user profile. Responses to knowledge requests are stored in the `SurveyResponse` artifact. The `answerChoiceIdList` field stores the list of IDs of selected options in case of multiple-choice questions. The same field is used to store a list of values for questions that allow users to enter more than one value as an answer.

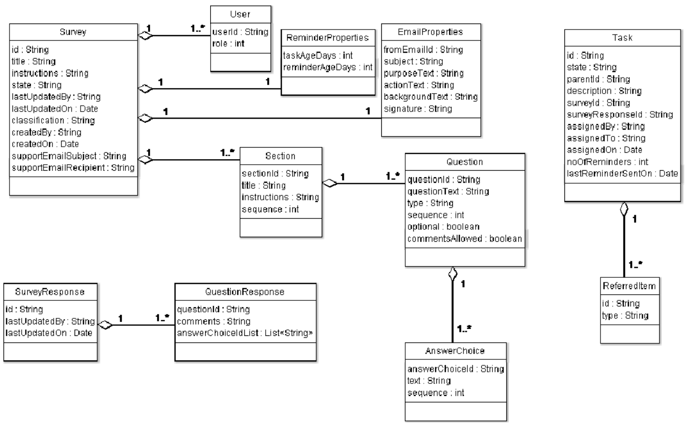


Fig. 3. Class diagram representing survey data model

When a user forwards or segments a task, a new task instance is created which contains the set of questions referred (as a list of ReferredItem instances). In case of a forward, all the questions referred to the forwarding user are part of the new task. For segmentation, the user has to select one or more questions, which form new task instance. For a given survey instance, a user might have more than one task assigned against his/her name. These tasks are consolidated into a set of logical tasks based on the instance id while displaying the task list and for other actions in the system.•

4.2 Services Platform

Figure 4 shows the overview of the BizRay components that implement our approach. At design time, a simple configuration file in the form of a spreadsheet allows to define the list of questions and sections that are part of a survey, capturing question type, optional comments, and mandatory flag. QuestionnaireManagement service enables survey creation, activation and closure. Once the survey is activated, administrator can assign task instances to the initial pool of users, by uploading a spreadsheet with task assignments. Expert Discovery Service[13] leverages internal social networking services to proactively discover alternate task assignees for tasks that are opened.

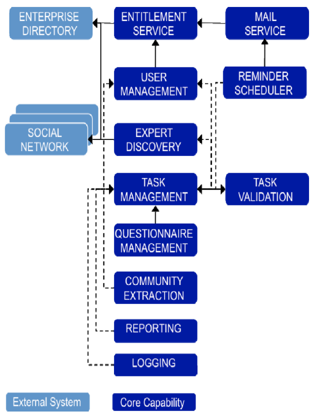


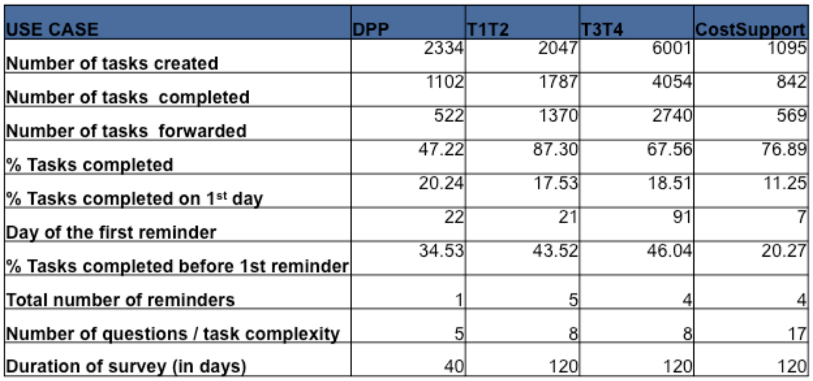
Fig. 4. Components providing the core capabilities of the framework

BizRay is built using Struts to support Model-View-Controller (MVC) design approach. It uses Java Server Pages (JSP), Servlets, HTML, JavaScript and Dojo for rendering the UI. Velocity framework is used to provide template capabilities, in particular to create dynamic email content. BizRay connects to the IBM LDAP server using Enterprise directory API (called BluePages) and to the database using Siena[14] (an artifact centric business process modeling tool developed by IBM tool which enables REST based communication).

5 Experiences in Deploying Deconstructed Surveys

Over the past two years we have been deploying the BizRay system, as the implementation of the deconstructed survey approach, to accelerate knowledge discovery in Services Delivery and IT Optimization domains. Specifically, the crowdsourcing campaigns have often been run to discover knowledge about a particular asset. E.g. in the IT Asset Management domain, by asset we mean a system, server or a business application that is hosted in the infrastructure. However, in the context of the BizRay system itself, an asset can be any business or IT entity whose properties you want to discover and enrich.

Figure 5 shows a deployment setup of 4 new use cases, in terms of number of tasks triggered, completed, complexity of the survey (number of questions) and number of reminders triggered.



USE CASE	DPP	T1T2	T3T4	CostSupport
Number of tasks created	2334	2047	6001	1095
Number of tasks completed	1102	1787	4054	842
Number of tasks forwarded	522	1370	2740	569
% Tasks completed	47.22	87.30	67.56	76.89
% Tasks completed on 1st day	20.24	17.53	18.51	11.25
Day of the first reminder	22	21	91	7
% Tasks completed before 1st reminder	34.53	43.52	46.04	20.27
Total number of reminders	1	5	4	4
Number of questions / task complexity	5	8	8	17
Duration of survey (in days)	40	120	120	120

Fig. 5. Deployment setup of surveys

In T1T2 and T3T4 use cases we are collecting hosting and compliance data for 7000 applications in order to ensure solutions must conform to the regulation(s), requiring that the applications be hosted within a particular country and administered only by persons that are legal citizens of that country. T1T2 and T3T4 denote tier (value/importance classification of the application). For a portion of this use case, 1800 applications were selected, 12% of the tasks opened where child tasks, similar to the results we saw in [4], and close to 50% of the tasks were reassigned to a better source, and within 8 weeks 60% completion was achieved. BizRay was further deployed to capture insights about defect prevention process (DPP) in the services delivery domain. This 5-question survey was assigned to 2334 quality analysts to better understand the impact of tools used for defect prevention. This case is a good collaboration example – over 25% tasks were forwarded. In CostSupport, Survey campaigns were centered about discovering financial and support application data, 1000s of applications, in the context of Application Portfolio Management, to eliminate significant fraction of current IT expenses needed to be evaluated. This particular use case leveraged use of BizRay system to eliminate the effort of data transcription (when surveys are sent out and managed through e-mail), as well as to support knowledge request size scaling (transcribing 4 questions per survey is somewhat "manageable", as opposed to having to manually collect and transcribe e.g. 17 questions, as was the case in this scenario).

Figure 6 shows how the task completion evolved in three different surveys: T1T2, T3T4, and DPP. Figure 7 shows the effect of reminders on T1T2 survey, and visualizes how reminders have a blasting effect on task completion. In all of these use cases, we have seen exploding patterns that coincide with the notification of tasks open. Simpler tasks, those that inquire up to 10 data elements are attended to, much quicker than complex tasks (e.g. that may require lookup to server repositories, etc.). In some cases complex tasks those that would take more than a few minutes are often left unattended and require several escalations.

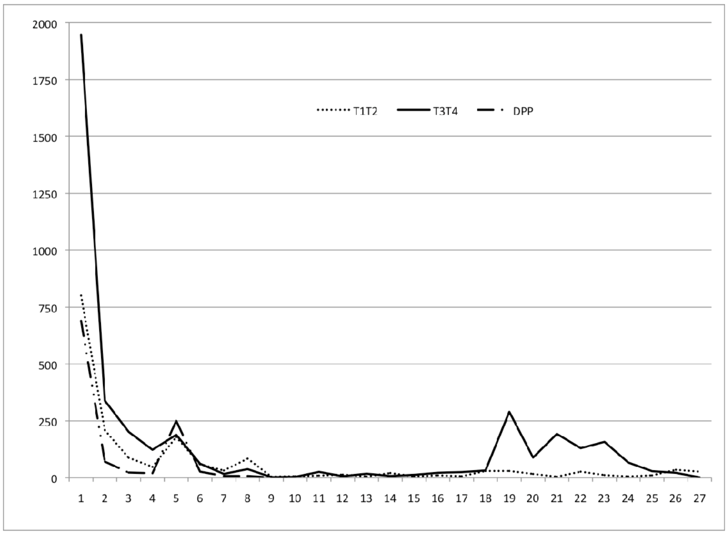


Fig. 6. Number of tasks completed in 5-day segments

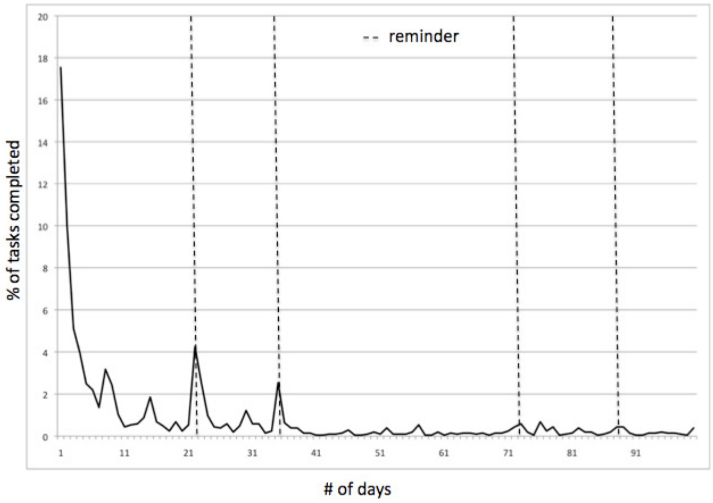


Fig. 7. Effect of reminders on task completion

Use case design has been shortened to about a person-week, often including the tool introduction, requirement gathering, and initial target community identification; and deployment in a multi-tenant environment is down one day. Finally, as many of these use cases may trigger information requests about the same application, BizRay is becoming a central point for managing the data about current applications, servers, etc. Updated data on e.g. application owners is fed into subsequent surveys.

6 Conclusions and Future Work

We extended our approach by designing a generic form to describe a survey structure and a task lifecycle. The deconstructed survey can be broken down into smaller pieces and distributed to multiple users. The approach is flexible enough allowing to be applied to a variety of use cases that seek knowledge discovery. This generic process has proven useful when scaling to a large number of instances, such as assets in a data center. Our ongoing and future work falls into the following areas:

1) Extend the knowledge capture process to context-based follow-ups.
2) Design incentives suitable for enterprise environments, building on the work in the behavioral economics [14].
3) Reuse communities that are discovered in one use case, to uncover sources of information for subsequent ones.

References

1. Li, N., Kang, J., Lv, W.: A hybrid approach for dynamic business process mining based on reconfigurable nets and event type. In: ICEBE 2005 (2005)
2. van der Aalst, W., Weijters, T., Maruster, L.: Workflow mining: discovering process models from event logs. IEEE Transactions on Knowledge and Data Engineering 16(9), 1128–1142 (2004)
3. Neumann, F., Robeller, A.: Enhanced audit log data analysis and query for BPEL processes with Process Choreographer 5.: IBM DeveloperWorks (2005), http://www.ibm.com/developerworks/websphere/library/techarticles/0503_neumann/0503_neumann.html
4. Vukovic, M., Lopez, M., Laredo, J.: PeopleCloud for the Globally Integrated Enterprise. In: Dan, A., Gittler, F., Toumani, F. (eds.) ICSOC/ServiceWave 2009. LNCS, vol. 6275, pp. 109–114. Springer, Heidelberg (2010)
5. Workflow Management Coalition, The Workflow Reference Model (1995)
6. Bobrow, D.G., Whalen, J.: Community Knowledge Sharing in Practice: The Eureka Story. Journal of the Society of Organizational Learning 4(2) (Winter 2002)
7. Kautz, H., Selman, B., Shah, M.: Referral Web: Combining Social Networks And Collaborative Filtering. Communications of ACM 40(3), 63–65 (1997)
8. Bryant, S.L., Forte, A., Bruckman, A.: Becoming Wikipedian: transformation of participation in a collaborative online encyclopedia. In: GROUP 2005: Proceedings of the 2005 International ACM SIGGROUP (2005)
9. Amazon Mechanical Turk, http://www.mturk.com
10. Survey Monkey, http://www.SurveymMonkey.Com/
11. Adobe FormsCentral, https://formscentral.acrobat.com/welcome.html
12. Howe, J.: The Rise Of Crowdsourcing. Wired 14(6) (June 2006)
13. Ypodimatopoulos, P., Vukovic, M., Laredo, J., Rajagopal, S.: Server Hunt: Using Enterprise Social Networks for Knowledge Discovery in IT Inventory Management. In: SERVICES (2010)
14. Cohn, D., Dhoolia, P., Heath III, F., Pinel, F., Vergo, J.: Siena: From PowerPoint to Web App in 5 Minutes. In: Bouguettaya, A., Krueger, I., Margaria, T. (eds.) ICSOC 2008. LNCS, vol. 5364, pp. 722–723. Springer, Heidelberg (2008)
15. Ariely, D.: Upside of Irrationality. Harper (2010)