Using Status Feeds for Peer Production by Coordinating Non-predictable Business Processes

Simon Vogt and Andreas Fink

Institute of Computer Science
Faculty of Economics and Social Sciences
Helmut-Schmidt-Universität Hamburg
Holstenhofweg 85, 22043 Hamburg, Germany
{simonvogt,andreas.fink}@hsu-hamburg.de

Abstract. Peer production uses the collaborative intelligence of its environment by relying on self-managed, decentralized coordination. Social software offers a broad variety of methods and applications for simplifying communication and harnessing collective intelligence. Status feeds, which are regularly used within social networks, may be considered as an important feature of these approaches. This article examines the use of status feeds for supporting the execution of non-predictable business processes. Given the context of Enterprise 2.0, existing business process management approaches will be discussed before developing resulting requirements for a feed-based system which will then be implemented as a prototype and showcased via an exemplary peer production process. The implementation is followed by an evaluation of the findings and results.

Keywords: peer production, business process management, status feeds, flexible workflows, Enterprise 2.0.

1 Introduction

"*The term 'peer production' characterizes a subset of commons-based production practices. It refers to production systems that depend on individual action that is self-selected and decentralized, rather than hierarchically assigned*" ([1] p. 62). Benkler's concept of peer production (also known as "social production") is based on the benefits that effective collective action brings to production processes. The recent progress in web technology and resulting new use cases open up a new horizon for the IT-based support for collaboratively executing business processes, summarized by the expression "Enterprise 2.0". This article focuses on the flexible and adaptive execution of business processes in the context of Web 2.0 and social software in a self-managed and decentralized environment. We take a design-oriented approach by identifying gaps in the current literature and then discussing new opportunities for the coordination of business processes, which will be realized and implemented by using a showcase example.

Section 2 starts by briefly outlining Enterprise 2.0 and discussing its use for supporting business process execution. This leads to new requirements and objectives (Section 2.1) and the design of the software architecture for the execution of business

F. Daniel et al. (Eds.): BPM 2011 Workshops, Part I, LNBIP 99, pp. 253–265, 2012.

processes (Section 2.2). In Section 3, we will describe the concept and the implementation of a status-feed-based system for supporting the execution of business processes. The resulting concept will be demonstrated using an exemplary implementation and finally be reviewed and evaluated (Section 4). Conclusions are summarized in Section 5.

2 Business Process Management in the Era of Enterprise 2.0

When Web 2.0 first rose, it primarily provided techniques and addressed applications for private users, not yet aiming at organizations and enterprises, but focusing on the communication and collaboration of multiple users ("social software"). Recently, especially short personal status updates became more and more relevant and popular. Wikis and blogs (weblogs) have already been transferred into the organizational context, named as Enterprise 2.0 ([2] p. 4, [3] p. 121ff., [4]), but the area of process management had nearly been untouched. The resulting needs for research have now been identified and several valuable approaches have been developed [5] [6].

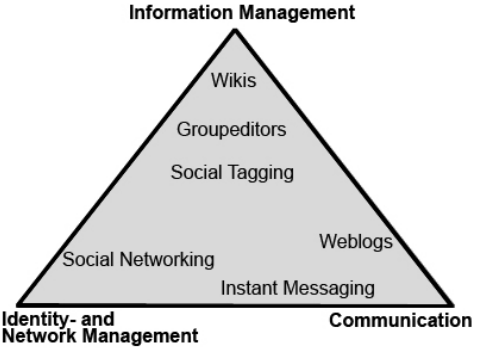

Fig. 1. Social software use cases ([7] p. 45)

A classification of typical social software use cases is given by Koch as presented in Fig. 1 ([7] p. 45, and [4]). Additionally, important usage areas may also be: management of knowledge and information, distribution of knowledge and information, management of experts and contacts, common creation of documents and project management and finally business process management (BPM). While the first four areas have already been covered by Enterprise 2.0 applications, BPM is still dominated by proprietary and custom-tailored systems, even though a few approaches and thoughts aiming to bridge this gap can already be found ([2] p. 645–722). Komus discusses the use of wikis for the collective development of instructions and procedural requirements ([8] p. 38), Neumann and Erol are using wikis to control business processes and develop the organizational requirements of enterprises and the system architecture [6]. Koch and Richter refer to the use of Atom/RSS to disseminate status updates of resources ([9] p. 125). This is where our following

approach and the concept of a feed-based system for business process support start of. Therefore, we will discuss the aspect of IT-based BPM, before we describe the design and implementation of a system using status feeds.

2.1 IT-Based Support for the Execution of Business Processes

Business processes, as associated sequence of enterprise tasks (functions, activities) for the purpose of generating a valuable output ([11] p. 10f.), can be divided in four classes ([12] p. 3ff.). *Production workflows* contain repetitive and highly predictable business processes and implement the core processes of an enterprise. *Administrative workflows* are also predictable and repetitive but of a simple structure and do not touch the core processes. *Collaborative workflows* may include several iterations over one task and are impossible to be predefined. *Ad hoc workflows* have no predefined structure at all, support is limited to documentation and offering communication channels, exceptions are very common. In this work, we focus on the aspect of supporting the execution of not completely predictable business processes (collaborative and ad hoc workflows) which are found in heterogeneous organizations as well as in interorganizational relations [13].

Concerning production and administrative workflows, companies mainly trust on partially adapted workflow management systems (WfMS) ([5] p. 11). They basically provide automatic coordination of business processes while distributing information or tasks to participating resources ([10] p. 50). WfMS are based on formal process models, created by arranging tasks in the correct sequence and connecting them with resources (human or machine). These workflows can then be handled by process engines, which interpret the specification and use it for a case-specific coordination and interaction of the required resources ([10] p. 50). As practical experience shows, WfMS may lead to several problems. These can be imprecise parameters, “overengineering”, a lack of planning, a variety of interfaces, a deficit in information, non-transparent action and interdependencies between instances of one process ([14] p. 2). Many software solutions try to solve these challenges, but there are several problems in the appliance of WfMS which are connected with the behavior of the company’s employees ([14] p. 1f). In particular, WfMS require a behavior of participants that conforms to the process specification ([14] p. 97, [5] p. 3ff). This often leads to a “model-reality-divide” between the assumed model on the one hand and the actual reality on the other hand. This may also involve that innovations and positive developments are inhibited because of strictly predefined processes ([10] S. 55). A lack of responsibility, complicated decision-making procedures, incomplete flows of information, division-oriented thinking, and orientation to one’s function alone may be the consequences that lead to deficits in innovation ([14] p. 2, [5] p. 3ff). A strictly hierarchical organization will empower these tendencies. Considering these difficulties, Benkler places emphasis on decentralization as “conditions under which the actions of many agents cohere and are effective despite the fact that they do not rely on reducing the number of people whose will counts to direct effective action” ([1], p. 62). Therefore, the involved employees and participants of one workflow can be described as a defined common/group, since they are not completely open for everyone (see [1], p. 61).

Collaborative and ad hoc workflows have so far mainly been addressed by using groupware software, which does not aim at automation but primarily at supporting the participant's communication ([7] p. 45). They can therefore create and exchange documents or get in contact via instant-messaging. The participants are self-organized; that is, there is no obvious structure or process for spectators to identify. Groupware has been existing before the rise of Web 2.0 which now brings in new opportunities and applications to harness collaborative intelligence and make use of collective action (for examples and studies see [15]), but is still unusual in the enterprise context.

Considering this gap, we aim at an Enterprise 2.0-based system for supporting and coordinating the execution of business processes. This should on the one hand displace groupware-solutions and its deficits. On the other hand it should create space for innovation and out-of-the-box thinking by establishing a structured, open and easy to follow platform for decentralized collaboration and peer production.

2.2 Software Architectures for the Execution of Business Processes

In the everyday use, software is often seen as a black box, though a closer look reveals a system of different involved and interacting components. The structured alignment of these system components as well as the description of their relations is covered by software architecture [16]. This section briefly introduces and compares the concepts of Service Oriented Architecture (SOA) and Event Driven Architecture (EDA).

The idea of the SOA is to connect and coordinate existing systems or components in a company instead of replacing them by a completely new homogenous system ([17] p. 89). Therefore, the SOA uses loose links of existing functions and applications by making them accessible via standardized protocols, independent of the used platform or language (e.g., web services). To provide extendibility, scalability and flexibility, the involved components do not communicate directly with each other but by using a service-bus.

The SOA is suitable for process-oriented workflows that can be described by requests, loops and single working steps ([17] p. 119). In contrast, the EDA provides a structure to support event-oriented processes, where an event can be described as a change of state ([18] p. 1). The EDA is not to be seen as a rival to the SOA but as an extension ([17] p. 121). Its software structure describes the so-called middleware (often message-oriented, MOM) as core element which can also be named as event-processor ([17] p. 121). The involved components send an event of a special type to this middleware where other components have previously registered themselves for special event-types. As soon as an event arrives at the middleware it sends a message to the registered components that will then pick up that event. Besides the middleware as event-processor there are event-publishers and registered event-consumers which will then continue with the next event on their part (event-reaction) [18]. This architecture determines some fundamental characteristics ([17] p. 122f.): The MOM works as mediator. Thus, an event-publisher does not know where and how the event is preceded next. The participating components do not communicate directly but in an

asynchronous manner, while the semantic of the exchanged messages has to be platform independent and standardized to be understandable for everyone. The event-consumers work autonomously and can decide themselves which events to take and what to do with them.

3 Design of a Feed-Based BPM-System

3.1 Requirements and Fundamental Architecture

According to the previous sections, our scientific objective is to develop a system to support the execution of collaborative and ad hoc workflows which provides a high level of flexibility and interoperability by using Web 2.0 elements and works as a platform for coordinating the process by considering the advantages of peer production. Therefore, the following requirements are set up for this artifact: i) According to the principles of Web 2.0: simple concept, low requirements for the technical infrastructure, low barriers for the users to participate, usage of collective intelligence, simple implementation and administration, open source-code. ii) Following the findings in BPM: flexibility against changes and deviations, simple adoption to inter- and intraorganizational conditions, low implementation and operating costs, usage of innovation conducted by employees, space for self-management, extendibility and interfaces for linkage and combining the software with existing systems.

To meet these requirements and achieve the scientific objective, we propose an approach which is inspired by social network's status feeds: it uses these feeds as description of events or transitions. Since we are trying to support the execution of collaborative and ad hoc workflows and to provide space for employees' innovations, ideas and synergetic effects, the approach synthesizes the EDA and the capability- and behavior-oriented process organization [14] along with the concept of peer production [1].

The general system design is based on three fundamental components and their interaction (see Fig. 2). The underlying structure corresponds to the EDA as described in Section 2.2, to meet the requirements as stated above, especially concerning the ad hoc characteristics of the processes that should be supported.

A crucial part of our approach is the message-oriented middleware (MOM) in the mediating role which creates and receives events on the one hand and publishes them on the other hand. Hence the MOM works as information platform of the event-consumers (participating components). The MOM itself is not able to manipulate an event, nor knows which event-consumers process what task. To manage the communication with the participating components, a publishing protocol becomes necessary which routes the MOM's information to the event-consumers as fast as possible. Since the consumers may be humans or machines, this protocol needs to be platform-independent which can be achieved by a well-defined syntax and semantic. The third component of the system is a protocol for publishing the executed transitions to the MOM that must be platform-independent for the same reason. Using these standards enables the communication and the execution of procedure-calls in heterogeneous systems.

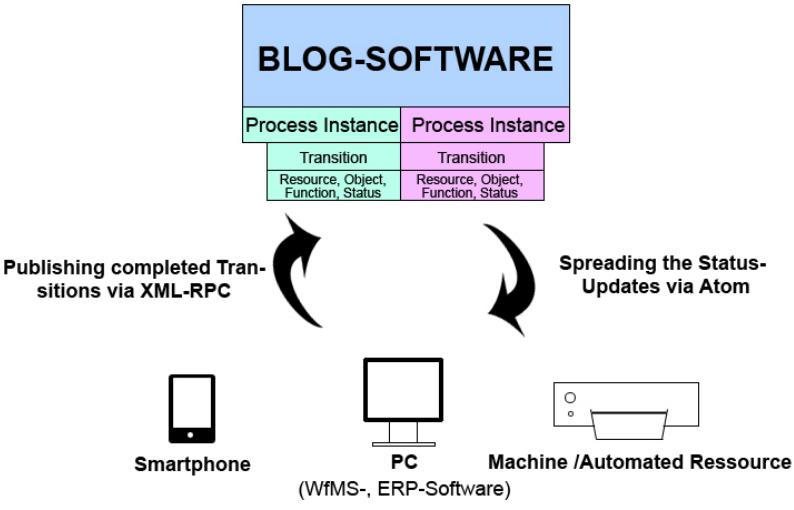

Fig. 2. Composition of the blog- and Atom-based system

3.2 Technical Design

A blog software (e.g., the Wordpress software) is well suited to represent the MOM. A new instance of a process is realized by a post, a new blog entry which can contain information about required steps, descriptions, remarks and a URL-based reference to the considered object. The standardized publishing protocol is realized by the Atom publishing protocol which had been launched in 2003 to establish a copyright-independent, extendible format. It has been developed under special consideration of the characteristics of blog systems ([19] p. 36). In most cases, one feed (a specific information channel of one provider) consists of article-related entries with the following elements: a title, a summary, and the content of one article itself, the link to the website of this article, and eventually some further extensions.

Publishing updates via Atom feeds will be designed in the style of the "Activity Streams" project which aims at a common, open, standardized protocol for status feeds of social networks. Its fundament is the Atom format, extended by a custom namespace [3] whose elements structure a status post into verbs (e.g. "post", "share"), objects ("blog entry" or "photo"), actors, time, and target. The application focus is on private status posts in social networks. However, the general idea of the common standardized grammar might be adapted to the actual context, especially because it enables that computers and machines react to some predefined vocabulary. This approach is related to the "Semantic Web" concept which tries to establish XML-based semantics for information to make it interpretable for machines [20].

For each post processed by the blog software, users are able to post comments. We will use them as notification of transitions or events, published via the blog by the involved event-publishers. Therefore they do not have to load some HTML document into their browser but can publish the notification via remote procedure call (XML-RPC). The typical interaction of the system is as follows: The participating resources for a process register at the MOM by subscribing the Atom feed of one specific post

(representing a new process instance) and become event-consumers. The blog software automatically generates this feed from the published comments for one post (in this case: notifications of transitions or events), so that registered users and devices are informed about new updates. They can then decide how to deal with this event, for example by taking it over (event reaction) and work on it. The resulting transition (changed state of the considered object) can then be published via XML-RPC – a simple implementation of the web service concept. It is basically a remote procedure call, coded in XML and transported via HTTP ([9] S. 214). This enables the communication and procedure-calling between heterogeneous systems. Notifications of a transition contain information about the considered object, the executing resource, its relating function, and its status.

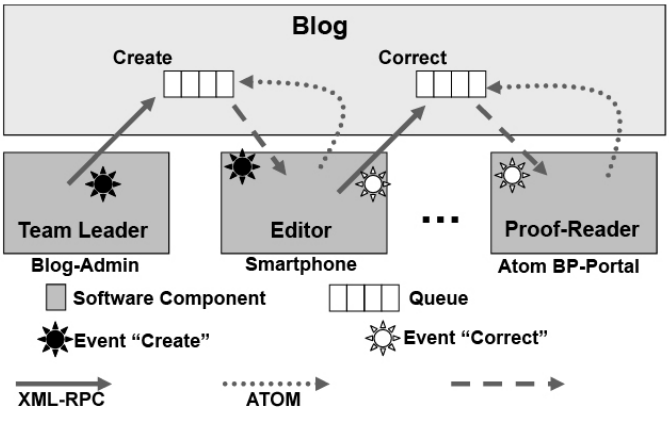

Fig. 3. System from the EDA-perspective (on the basis of [17], p. 122)

By the described arrangement, the system complies with the EDA: the MOM works as mediator and does not know where and how an event is processed. The required common semantic on the technical level is assured by using the Atom protocol and XML-RPC. The participators decide autonomously and are not directed by a central engine. Fig. 3 shows the system from the process-view, based on the EDA. Because Atom, as well as XML-RPC, is based on the platform-independent XML, all participating resources can interact with each other. It is possible to link this system and the conducted process to some WfMS or ERP-software via XML-interfaces. For example, a new sub-process can be started by the responsible WfMS by creating a new post in the blog via XML-RPC. As soon as this sub-process is completed, this information can also be forwarded to the WfMS in XML. The integration of machines could be achieved similarly.

4 Implementation and Case Study

In this section we explain the implementation of our concept by discussing an exemplary use case and describing the developed software artifacts.

4.1 Use Case Scenario

A publishing house's department for a monthly released magazine plans articles for the next issue. The team consists of employed but also freelancing journalists, graphic artists, photographers, editors, and proofreaders. A team leader sets broad guidelines for each issue and its content, in reconciliation with the departments for marketing and controlling. Apparently the team is of a heterogeneous, interorganizational structure without a strict hierarchical order. In this example, the December issue shall (amongst others) feature a detailed article covering the peace-process in the Middle-East. The initial blog post in the category "Articles" is created by the team leader and at first only contains the following information:

Peace-process in the Middle East
By Teammanager:
- *8 pages available*
- *featuring photos, graphs and rich illustration*
- *interviews?!*
- *to be done until 20th November*
- *files to be uploaded and shared in http://[path]*

The Wordpress-based implementation uses two different feeds: The more general feed publishes incoming new posts (here: instances of processes), while post-oriented feeds spread comments that have been entered for specific posts (here: notifications for transitions). The first feed distributes the team leader's post and is modified in a way that makes it deliver the URL of the post-oriented feed (see below). Thus, team members can subscribe to the comment feed without having to open the blog website in a browser.

The recipients of this feed can decide self-responsibly if they will take part in the creation of this article. If an editor is already busy doing research for two other projects, he may just write the subtexts for the photo story. Another editor may instead have enough capacity and is an expert for Middle-East topics. He works in Istanbul in an agency of this publisher and sees the team leader's new post about this article in his modified newsreader. He subscribes to the specific feed and uses the built-in XML-RPC interface for publishing his notification "*started research*". An independent photographer, who already owns pictures for this topic, reads about this article on his smartphone's RSS-reader. Via a web browser he opens the blog website and writes (below the editor's notification): "*started illustration*". The posted status notifications are immediately spread via the specific feed of this article so that other potential participants can see if a task is already executed. As a next step, the editor and the photographer can report their activities as "*finished research*" and "*finished illustration*", so the proofreader is able to start his work ("*started correction*"). As soon as he has posted his notification "*finished correction*", the publishing house's printers will recognize this because of the standardized vocabulary and can immediately start printing or buffering the article until the remaining content of this issue arrives as completed event. In the same way, the printer publishes its status report via XML-RPC.

This example shows how the resources of a creative process which is highly flexible can be coordinated using a modified blog, Atom, and XML-RPC to enable the communication. The underlying simple implementation will be described in the following section.

4.2 Changes in Wordpress

Wordpress has been developed as blog software, thus is not designed for BPM. However, the number of required changes for the concept as described above is surprisingly small. The Wordpress XML-RPC-interface can be activated via its administration-interface ("Dashboard"). The existing simple text field for comments has been changed to two lines – one for the current status of a function, and one for the function itself (like "correction" and "finished"). Because the username of each comment-author is automatically taken over by Wordpress and a comment can only refer to one specific post, a notification accordingly contains the considered object (represented by the post), the operating resource of a task (the author of one comment), its function as well as its status. In the following step, the automatically generated Atom feed that gives an overview of the existing posts has been modified and now also comes with an URL of the related comment-feed for each post. Thereby the participants do not have to load the blog website to subscribe to one specific feed. They immediately receive the required address. This procedure also simplifies the integration of machines and automated resources.

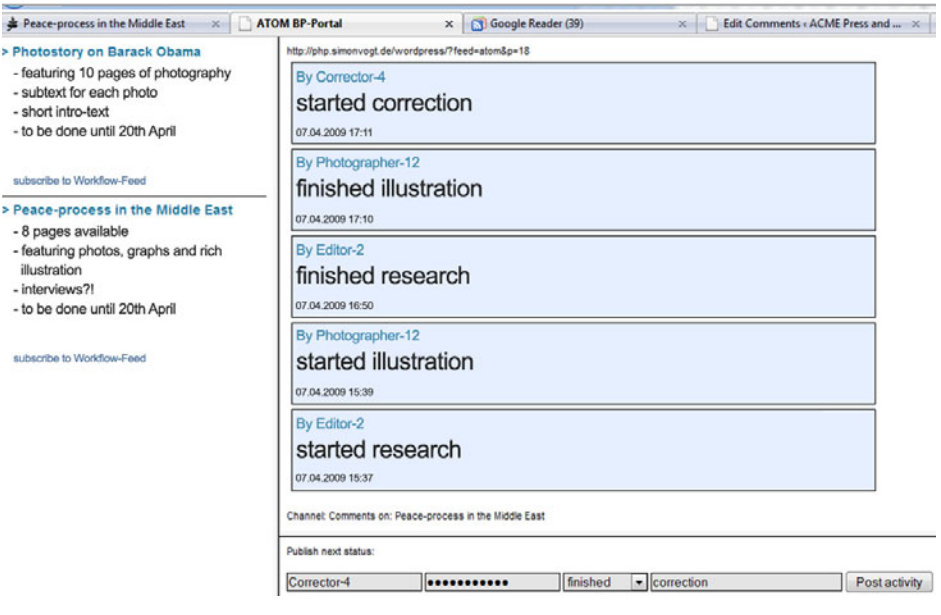

Fig. 4. Atom BP-Portal with selected process-feed

4.3 Browser-Based Atom-/XML-RPC-Client

To ensure that the participants have an overview about the existing and currently relevant feeds and have the opportunity to publish their own updates (without having to open the blog), we prototypically implemented a web-based and thus platform- and location independent user interface using PHP/HTML ("Atom BP-Portal"), as shown in Fig. 4. This application consists of three major parts: At first, an Atom parser for the general Wordpress feed which contains all existing posts (in our case: all existing process instances) and has been modified as described in the previous section. At second, a parser for the specific post that shows the progress of one process, and, at third, a form that enables the publishing of updates to the considered feed directly from this user interface using XML-RPC.

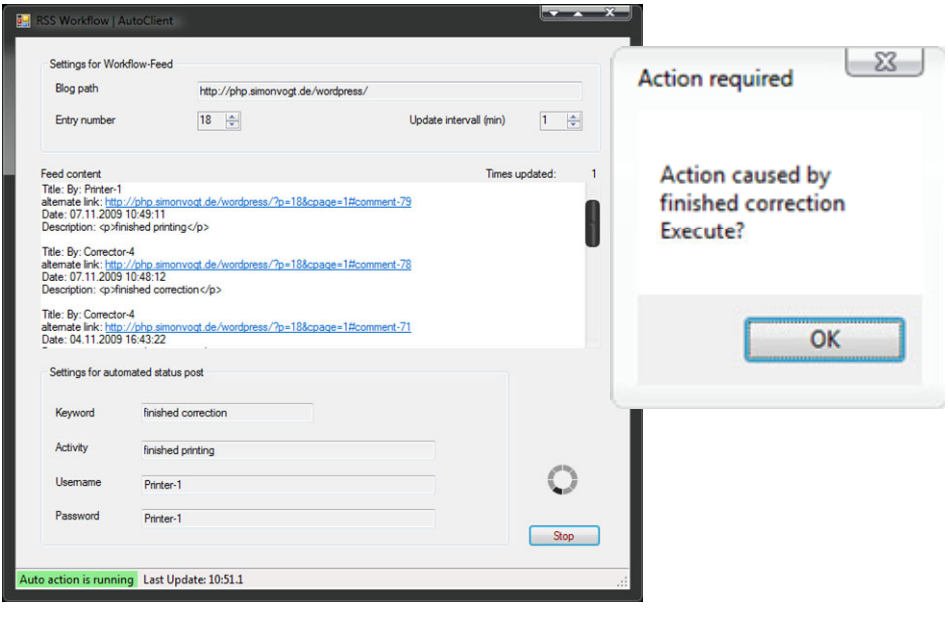

Fig. 5. AutoClient with automatic reaction to key phrase

4.4 Automated Atom-/XML-RPC-Client

The previously described portal application meets the requirements of human resources in a heterogeneous environment. As indicated before, it also is possible to integrate machines and automated resources into the process. That will now be demonstrated by a C#-application ("AutoClient") that simulates a printing unit. It reacts as soon as the proofreader or the team manager publish their final notification, enabled by the standardized vocabulary. The application therefore includes the XML-RPC-functionality and an Atom parser. Furthermore, it has a GUI to set the preferences (see Fig. 5), like the ID of the considered process, the update-interval of the feed, the key phrase that causes the action (e.g. "correction finished") as well as the status notification

that will be published in reaction to that (e.g. “printing finished”). After the Start-button has been clicked, the application periodically reloads the specific feed of the process and as soon as it reads the key phrase it automatically starts its action (represented by a dialog box) and publishes the new status via XML-RPC.

These applications along with the use case example give an impression of the abilities that the combination of a blog, Atom and XML-RPC offer. This system can easily be adjusted to one company’s circumstances by adapting the Wordpress software or custom client software, since all elements are open source and platform independent.

5 Conclusions

„Given what we're trying to do now, what is the simplest thing that could possibly work?” [21]

This question was Ward Cunningham’s guiding credo while developing the Wiki-concept. The Web 2.0 does not consist of complex new developments but of a new composition of existing methods and techniques. So does our system. Blogs and feeds are already common in the enterprise context but in most cases just for simple news-publishing and marketing purposes.

Evaluating the developed software artifact against the scientific objective shows that the requirements as constituted in Section 3.1 are fulfilled: The concept is simple and only few changes to existing software have been made, the infrastructural requirements are low since only a web server is needed; the applications are platform-independent, open source and can be linked to existing WfMS or ERP-software. The barriers for user-participation are kept low as well, since blogs and feeds are a common feature in the everyday internet-use. The collective intelligence is used by letting the employees manage the process themselves. The implementation and customization of the software is easy and cheap, since it is open source and XML-based.

Taking the EDA as architectonical fundament enables full flexibility for the sequences of a process and conforms to the concept of peer production. However, because of the self-management within the EDA concept, complex events that involve multiple resources are difficult to coordinate. In addition the blog does not provide a discussion platform for the participants, since the comment feature is used for status notifications. This might be compensated by an additional forum/wiki. Additionally, monitoring can be ensured by using a Wordpress rating-plug-in. Since the Atom protocol is based on a pull-mechanism, delays may occur in spreading the status notifications to the resources. To minimize this delay, the Pushbutton-Web-concept as well as Pubsubhubbub may be used. These (still experimental) techniques create a hub between publisher and clients to provide a real-time circulation of feeds [22]. The system that has been developed in this article is only one of multiple opportunities to achieve the required functionality and architecture. It is also possible to use other platforms than Wordpress.

A software solution can only be effectively assessed and improved when used in reality. Practical experience enables to draw valid conclusions considering the suitability and the potential. From a research perspective we envision the use of the focus group approach for artifact refinement and evaluation [23]. The future

development of social software in general as well as this particular system in the context of BPM has to be monitored. Hence, this article offers an approach for further development and amends the repertoire of Enterprise 2.0-tools and methods to coordinate business processes.

References

1. Benkler, Y.: The Wealth of Networks: How Social Production Transforms Markets and Freedom. Yale University Press (2006)
2. Ardagna, D., Mecella, M., Yang, J. (eds.): Business Process Management Workshops. LNBIP, vol. 17. Springer, Heidelberg (2009)
3. Atkins, M., Norris, W., Messina, C., Wilkinson, M., Dolin, R.: Atom Activity Extensions 1.0 (February 13, 2011), `http://activitystrea.ms/spec/1.0/`
4. Hippner, H.: Bedeutung, Anwendung und Einsatzpotentiale von Social Software. HMD – Praxis der Wirtschaftsinformatik 252, 6–16 (2006)
5. Erol, S., Granitzer, M., Happ, S., Jantunen, S., Jennings, B., Johannesson, P., Koschmider, A., Nurcan, S., Rossi, D., Schmidt, R.: Combining BPM and Social Software: Contradiction or Chance? Journal of Software Maintenance and Evolution: Research and Practice 22(6-7), 449–476 (2010)
6. Neumann, G., Erol, S.: From a Social Wiki to a Social Workflow System. In: Ardagna, D., Mecella, M., Yang, J. (eds.) BPM 2008 Workshops. LNBIP, vol. 17, pp. 698–708. Springer, Heidelberg (2009)
7. Koch, M.: Lehren aus der Vergangenheit – Computer-Supported Collaborative Work & Co. In: Buhse, W., Stamer, S. (eds.) Enterprise 2.0 – Die Kunst, loszulassen, pp. 17–35. Rhombos, Berlin (2008)
8. Komus, A.: Social Software als organisatorisches Phänomen – Einsatzmöglichkeiten in Unternehmen. HMD – Praxis der Wirtschaftsinformatik 252, 36–44 (2006)
9. Koch, M., Richter, A.: Enterprise 2.0. Oldenbourg, München (2009)
10. Weske, M.: Business Process Management. Springer, Berlin (2007)
11. Scheer, A.-W.: ARIS – vom Geschäftsprozess zum Anwendungssystem. Springer, Berlin (1999)
12. Alonso, G., Agrawal, D., El Abbadi, A., Mohan, C.: Functionality and Limitations of Current Workflow Management Systems. IEEE Expert 12(5) (1997)
13. Vogt, S., Fink, A.: Status-Feeds zur flexiblen Koordinierung nicht vollständig voraussehbarer Geschäftsprozesse. Working Paper (2011)
14. Bartelheimer, G.: Leistung nicht aus Zufall – Verhaltensorientiertes Prozessmanagement. Tectum, Magdeburg (2009)
15. Swenson, K. (ed.): Mastering the Unpredictable – How Adaptive Case Management Will Revolutionize the Way That Knowledge Workers Get Things Done. Meghan-Kiffer Press, Tampa (2010)
16. Balzert, H.: Lehrbuch der Software-Technik. Spektrum Akademischer Verlag, Heidelberg (2001)
17. Dunkel, J., Eberhart, A., Fischer, S., Kleiner, C., Koschel, A.: Systemarchitekturen für verteilte Anwendungen. Hanser, München (2008)
18. Taylor, H., Yochem, A., Phillips, L., Martinez, F.: Event-Driven Architecture – How SOA Enables the Real-Time Enterprise. Pearson Education, Boston (2009)
19. Wittenbrink, H.: RSS and Atom – Understanding and Implementing Content Feeds and Syndication. Packt Publishing, Birmingham (2005)

20. Berners-Lee, T., Hendler, J., Lassila, O.: The Semantic Web: A New Form of Web Content that is Meaningful to Computers Will Unleash a Revolution of New Possibilities. Scientific American 284(5), 34–43 (2001)
21. Venners, B.: The Simplest Thing that Could Possibly Work – A Conversation with Ward Cunningham, Part V (January 19, 2004), http://www.artima.com/intv/simplest3.html
22. Dash, A.: The Pushbutton Web: Realtime Becomes Real (July 24, 2009), http://dashes.com/anil/2009/07/the-pushbutton-web-realtime-becomes-real.html
23. Tremblay, C.M., Hevner, A.R., Berndt, D.J.: Focus Groups for Artifact Refinement and Evaluation in Design Research. Communications of the Association for Information Systems 26(27) (2010), http://aisel.aisnet.org/cais/vol26/iss1/27