Weaving Social Software Features Into Enterprise Resource Planning Systems

Dirk Draheim[1], Michael Felderer[2], and Viktor Pekar[3]

[1] University of Mannheim, 68131 Mannheim, Germany
`draheim@acm.org`
WWW home page: `draheim.formcharts.org`
[2] University of Innsbruck, 6020 Innsbruck, Austria
`michael.felderer@uibk.ac.at`
WWW home page: `http://homepage.uibk.ac.at/~c703409/`
[3] University of Innsbruck, 6020 Innsbruck, Austria,
`viktor.pekar@uibk.ac.at`

Abstract. In this paper we present the Social Weaver platform that enables end users to weave snippets of social software features into the workflows of existing enterprise applications. We discuss the underlying vision from a technological viewpoint, i.e., an end-user development viewpoint, and an organizational viewpoint which is about a certain ubiquitous understanding of enterprise application integration. We present the system's requirements, architecture and realization. The concrete platform is based on the standard web technology stack, which makes sense because the web is the current natural host for enterprise applications, at least for new ones. However, the approach presented in this article is technological-independent with the concrete platform as a concrete instance proving the approach as doable. Conceptually, the realized platform is a key to analyze the current situation and possible future of today's enterprise application landscapes which oscillate between emerging social software metaphors and an ever increasing degree of process automation found in today's organizations.

Key words: Social Software, Web 2.0, Enterprise Content Management, Wikis, Enterprise Resource Planning, Business Process Technology, Workflow Management, Big Data, Aspect-Orientation

1 Introduction

Currently, we see that new social software technologies gain ground in organizations. They come along with new agile and equal approaches to work organization. At the same time, classical enterprise resource planning (ERP) systems are still and always ever growing, simply in terms of the number of processes they support. Many larger organizations still have their own software development departments for realizing process-based applications and it seems that the demand for more and more process automation cannot be satisfied adequately. The two major technological strands of computer-supported cooperative work

(CSCW) [20] on the one hand side and process-based automation [10] on the other hand side co-exist in organization with little to no integration. The same is even truer for the third strand of IT which is about individual office automation and individual ad-hoc IT support. This situation is also the driver for the current big data debate, which is an analytical *a posteriori* approach that massively targets the data facet of this scenario. Here is where the technological and organizational vision of our work starts. We present the Social Weaver platform [29, 32] that enables end users to weave snippets of social software features into the workflows of existing enterprise applications as illustrated by Fig. 1. The technology allows the end-user to enrich existing ERP systems by features of existing social software applications at the level of user dialogs.

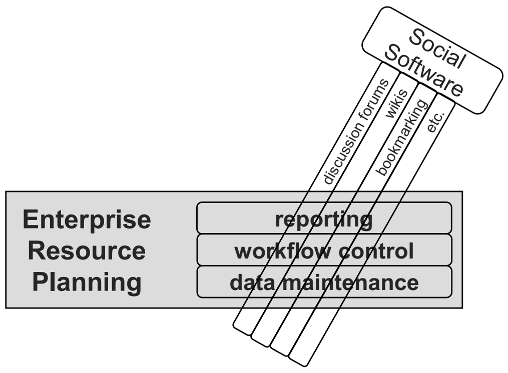


Fig. 1. Weaving social software features into ERP systems.

The approach presented in this article is technological-independent. As a concrete instance proving the approach as doable, the current Social Weaver platform is based on the standard web technology stack. This makes sense because the web is the current natural host for enterprise applications, at least for new ones. The developed platform exists as a stable, working version and has recently been launched as an open source software development project:

`https://github.com/vikpek/SocialWeaver`

The overall vision of this work can be explained in two strands, i.e., a technological and an organizational one, that represent two facets of the same story. We delve into the technological vision in Sect. 2 that starts very concrete with the description of example ERP use cases for the targeted technology. We proceed with an explanation of the organizational vision that in Sect. 3 that is justified by and at the same time transcends the technological vision of Sect. 2. The architecture and implementation of the realized Social Weaver technology are explained conceptually in Sect. 4 and by means of a walkthrough through a concrete example case study based on the Google calendar application in Sect. 5. We report on further work and related work in Sects. 6 and 7, respectively. Finally, we finish the paper with a conclusion in Sect. 8.

2 The Technological Vision

The technological vision of this work is about *end-user application weaving.* Imagine an arbitrary staff-related approval process in an enterprise, e.g., business trip planning, request for leave, or a request in the personnel disposition system. Now, we have that a certain change in the relevant employment law applies and a change to the workflows is needed. Before on the final solution is decided and the appropriate changes to the supporting ERP system is implemented, the chief personnel manager wants to communicate certain hints and orders to the staff that must be obeyed as a workaround to the eventual solution when working with the current processes. Because this phase is temporary, let us say 2-3 weeks, the personnel manager wants to communicate this hints in a lightweight yet robust manner. One option would be to write an Email to the staff. This has certain drawbacks. First, the email will also reach many employees that are not currently interested in, because the relevant workflows do not apply to them in the temporary phase. Second, there is the risk that the important information is overlooked or forgotten once the concrete workflow steps are performed. The same disadvantages apply if the information is posted in an enterprise-wide news system, e.g., based on a social software initiative or the more established enterprise content management system technology. Here, the risk that the information is overlooked might even be higher.

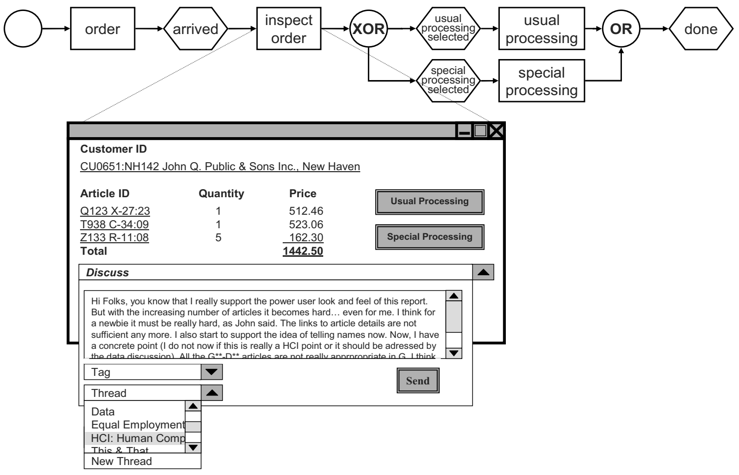
The desirable solution would be to post the necessary hints and orders directly in the relevant ERP system workflow steps. The crucial problem in current settings is that this will result in a change request to the software development department or the extern software service provider which needs to be considered a way too heavyweight solution. The technology realized by the current work offers way out of the dilemma. With our technology the personnel manager or its secretary can immediately drop the necessary hints as comments to the relevant workflow steps. Only minimal to no training is needed to use the technology which is largely self-explanatory. This means our technology follows an end-user development approach in terms of HCI (human-computer interaction) [24], and, actually, a very high-level end-user development approach.

With the current implementation of Social Weaver it is exactly possible to drop a comment via a comment box as an annotation to each element of a web site. The system has an extendable architecture and in principle it is possible to drop any social software feature to a workflow step in our technological concept. Web 2.0 frameworks and portal servers like Alfresco or Liferay offer many ready-to-use features and plugins. For instance, the personnel manager in our little example could enrich a workflow step by a means of gathering feedback from the employee, e.g., by a poll feature or a small feedback form. This way he could gather further information that is needed in the workaround to the new approval workflows.

It is important to realize that our approach does not answer the problem of data integration for the information that emerges by such added features. The entered data will be accessible in the social software application that is intertwined with the ERP system. A certain amount of data integration can be achieved if the social software application feature replays the information if the workflow step is re-visited or displays the information in another workflow step. However, we do not elaborate the pattern for such integration and it depends on the cleverness of the end-user to exploit the social software feature in such a way. Our approach intertwines and integrates application only at the level of user dialogs, the data tiers of the application remain separated silos. However, the approach is a crucial step forward in the direction of rapid, end-user enabled application integration. We wanted to create somehow a sweet spot between usability and rapidness vs. design robustness and maturity. Therefore, the current example of a temporary process workaround is very typical. We feel that the usefulness of such approach is quite evident.

Have a look at a second example in Fig. 2 that we have taken from [9]. In this example a complete discussion forum from a social software framework is woven into the order process of an ERP system. The motivation is a continual improvement process for workflows and their supporting IT systems. We deal with an example of a truly high-repetitive process, i.e., the inspection of orders that is the main tasks of the respective specialized clerks. One of the employees has entered the discussion forum by linking it directly to the workflow steps he wants to discuss. Other employees will find this entry to the discussion forum exactly when they enter this workflow step. The discussion forum is itself a complex application with structured data and some kinds of processes. The example gives an impression of how two complex applications can be linked on the level of their dialogs. Again the integration is shallow, i.e., the data facet is not addressed. It will be further work to elaborate a concept for end-user data integration and our belief it that the way is via eventually solving and overcoming a fundamental problem of the current state-of-the-art in software development and operations, i.e., the lack of deep standardizations of applications and systems as described in [2].

Given these two examples as starting point, we can also identify a wide range of interesting concrete application areas in the software engineering life cycle: software testing [18], software quality improvement [19], legacy system refactoring and so forth. For example, in legacy system refactoring it appears natural that the team that re-engineers the software requirements of the legacy system discusses and documents the system along the lines of its dialogs. Again, we feel that the approach is immediately promising but at the same time shows crucial limitations with respect to the data integration facet, in particular, if holistic and integrated viewpoints on software engineering toll support [5, 2] are taken. Here, the problem of loose coupling of the data can be re-phrased in terms of the concept of traceability, which is a well-understood concept in the CASE tool community.



Customer ID
CU0651:NH142 John Q. Public & Sons Inc., New Haven

Article ID	Quantity	Price
Q123 X-27:23	1	512.46
T938 C-34:09	1	523.06
Z133 R-11:08	5	162.30
Total		1442.50

Hi Folks, you know that I really support the power user look and feel of this report. But with the increasing number of articles it becomes hard… even for me. I think for a newbie it must be really hard, as John said. The links to article details are not sufficient any more. I also start to support the idea of telling names now. Now, I have a concrete point (I do not now if this is really a HCI point or it should be adressed by the data discussion). All the G**-D** articles are not really apprpropriate in G. I think

Fig. 2. Example for weaving of a complete discussion forum into a workflow step.

Taken all the examples of this section together, it can be stated that, overall, the proposed platform lays in the realms of enterprise knowledge management [26, 28] and enterprise application integration [21] which will be the topic of the next section.

3 The Organizational Vision

The organizational vision of this work is about *ubiquitous enterprise application integration.*

ERP systems form a center pillar of IT in today's organizations. Their role is not merely some work support that is only justified by some speed-up of work. Their role is crucial, i.e., mission critical in the way they formalize, document and guide work. ERP systems encapsulate major chunks of the enterprises know-how. However, not all kinds of work are amenable for a process-based approach. Some tasks are particularly well-suited for process automation. They are typically highly repetitive and form major chunks of work of often specialized work forces in daily operations. For other tasks, e.g., in the areas of organization, planning, project work or team work, it might be more difficult to analyze their activities and to support them with classic ERP systems. ERP systems are based on structured data and formal processes. In the field of unstructured and semi-structured data and more ad-hoc processes another class of tool emerged over the decades that can be loosely classified as groupware or CSCW (computer-supported cooperative work) tools. A plethora of unstructured data and semi-structured data emerges from individual work and it has been the field of enterprise content management (ECM) [4] to gain control over these data.

There exist strands of informal work in enterprises. Sometimes, it is not rational to formalize certain kind of work, sometimes work arises and is not yet understood well enough to be formalized. In any case, IT support should seek for the sweet spot of bringing some structure into any kind of work found in enterprises. At least, a reasonable versioning and access control management for the artifacts emerging in work always helps. For collaborative work on artifacts, tools for rapid editing have early been proposed, e.g., ZOG [27], and recently gained wide acceptance by the Wiki wave [6]. The usefulness of Wikis to support team work is immediately evident, simply by the features they offer and independent of the equal work philosophy that comes along with their encyclopedic work metaphor. Today's social software products [16] now combine features from enterprise content management systems and collaborative editing tools which seems to be a natural step against the background of an analysis of the needs of organizational IT support. And this is how social software is currently actually exploited in enterprises, the usefulness for the extra genuinely social software features that come along with new products is not yet fully understood. By the way, an early representative of combining ECM with collaborative editing are Hyper-G and its successor Hyperwave [1].

The current big data debate [17], as long as it concerns enterprise data, also shows this duality in structured and semi-structured work and data. The survey in [30], see Fig. 3, shows which data sources are exploited in systematic data analysis efforts in today's organizations.

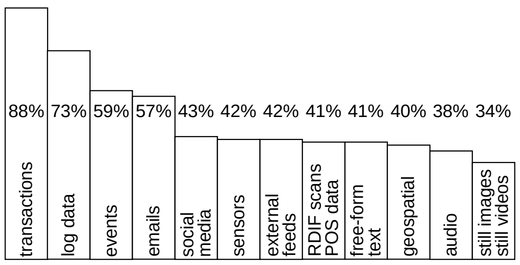


Fig. 3. Current exploitation of data sources in organizational big data initiatives – IBM poll [30] with approx. 500 respondents.

In our *ubiquitous enterprise application integration* vision we think of all the data in the enterprise as one single huge externalized knowledge basis [28] that is exploited by all processes and work activities in the enterprise. In [10] we have discussed this ideal as industrial information integration backbone (IIIB). Whereas a technical realization of such an information and process design currently seems nigh on impossible, it helped us in concrete enterprise application projects [25, 23, 11, 3, 8, 7] to find an optimal solution. Therefore, we think it is the correct metaphor for the case of integrating ERP systems with emerging social software technology.

4 Architecture of the Social Weaver Platform

The realized system functionality is described best by a generic use case of a user opening a web application and modifying content – see Fig. 4. First, the user opens a web application. Then, the Social Weaver plugin sends a notification to the server with all necessary information like user identifier, time stamp and payload. After the server has received the plugin message, it synchronizes with its current content in the database. The server application responses to the plugin client with content data if such data exist. Then, the plugin uses the content information from the server to insert all social web elements. Now, the user decides to make some changes to the social web content, e.g., adding a comment or creating a new comment box. Again, a notification is sent to the server containing changes. On behalf of that, the server synchronizes the updates and responses. Finally, the plugin redraws the synchronized content.

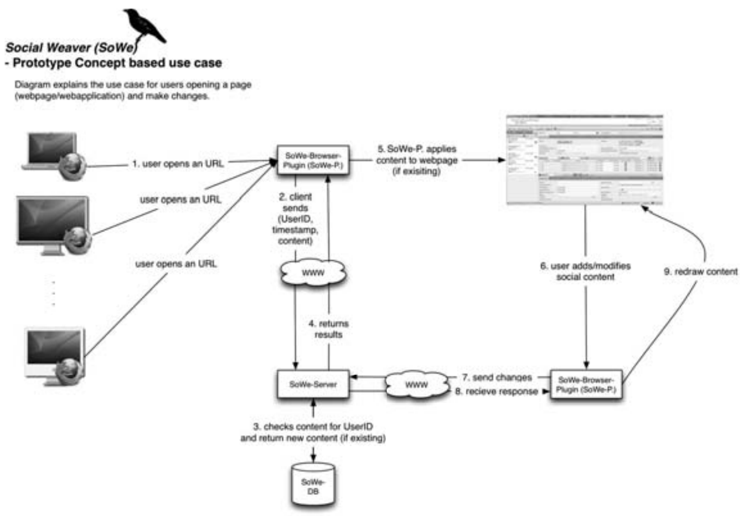

Social Weaver (SoWe)
- Prototype Concept based use case
Diagram explains the use case for users opening a page (webpage/webapplication) and make changes.
1. user opens an URL
user opens an URL
user opens an URL
SoWe-Browser-Plugin (SoWe-P.)
5. SoWe-P. applies content to webpage (if exisiting)
2. client sends (UserID, timestamp, content)
www
4. returns results
6. user adds/modifies social content
9. redraw content
SoWe-Server
www
7. send changes
8. recieve response
SoWe-Browser-Plugin (SoWe-P.)
3. checks content for UserID and return new content (if existing)
SoWe-DB

Fig. 4. Social Weaver prototype use case.

The system consists of a web tier and a server tier. The basic architecture is shown in Fig. 5. The plugin takes control of one or multiple user sessions and draws the additional content into the browser view. The server application synchronizes with each plugin and distributes updates between several clients.

Social Weaver is designed in a modular way, so that it is possible to add more social media features, support multiple platforms and more web applications. In the current implementation, weaving of comment boxes is supported. The realized user experience is that browser displays comment boxes that are related to specific web elements. For example, in an online calendar a user can add a comment box related to an appointment that he wants to discuss in detail. Because it should be possible to add multiple comment boxes to any web element,

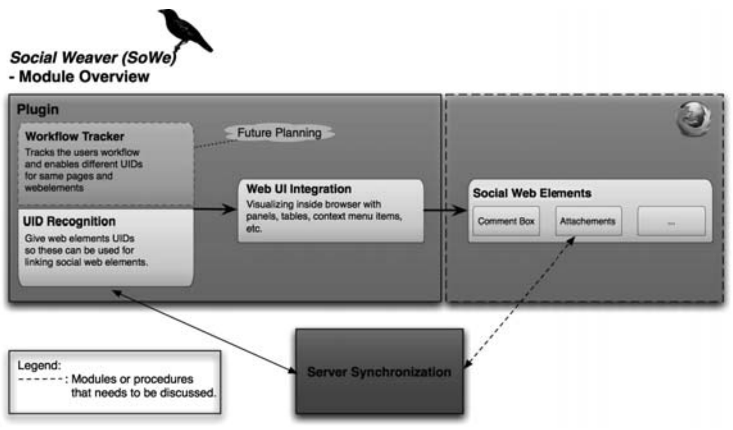

Social Weaver (SoWe)
- Module Overview
Plugin
Workflow Tracker
Tracks the users workflow and enables different UIDs for same pages and webelements
Future Planning
UID Recognition
Give web elements UIDs so these can be used for linking social web elements.
Web UI Integration
Visualizing inside browser with panels, tables, context menu items, etc.
Social Web Elements
Comment Box
Attachements
...
Server Synchronization
Legend:
------: Modules or procedures that needs to be discussed.

Fig. 5. Social weaver module overview.

we cannot just drop a box inside the user view, overlapping other interesting parts of the web application. Hence, we have the requirement to make additional content visible to the user, without interfering with the view on the original content. Possibilities would be fold/unfold-windows or just using small icons as references in the original view and outsource additional social content in external windows.

Social Weaver – The Plugin Of course, the plugin needs to be able to communicate to the server application as well. First of all the plugin needs to receive data that it prints to the screen. Secondly, changes made by the user have to be reported to the server. Because we are distributing the information between several users, there is also a need for synchronization. User updates may not overwrite updates made by other users etc. Social weaver exploits a parser framework containing application programming interfaces that creates and parses the content of our tuples. For instance, it is this way easier to add plugins for other browsers. The data in the content-part of our tuple should have a uniform format no matter what web application or browser is in use. The server application does not need to be aware about the environment the plugin runs in - it manages the social web content independently.

Another important point is the interaction with the web application. Most such sites are dynamic and there exists no static and referable URLs. In addition, it is not certain that the same element that two users refer to in their independent sessions has a comparable identifier. This issue definitely needs to be handled specifically for any web application. The good news is that this only affects the plugin. The server application just needs clearly defined identifiers. As a solution for the plugin, we need the possibility to use scripts for identifying elements. This requirement is probably the vastly problematic one because it prevents a general usage of Social Weaver.

Social Weaver – The Server Application The server application's primary task is to synchronize different user sessions on one or multiple web applications. A user session is defined within the plugin, which does not mean a plugin can manage only one session. The server receives messages from different sessions, synchronizes them and distributes the most current state to all sessions. To establish a loss less synchronization every message contains a time stamp. We are assuming that every message contains a user identifier, a time stamp and an unique identifier for an element within the web application. This Anchor is the unique identifier for a single user action. For example, if a user adds a comment to an already existing comment box that is related to an appointment in a calendar, the server receives the users identifier, the time stamp for the modification and an identifier for the appointment in the calendar. With this information, the server can check its database for the comment box and add the new comment. It is important to realize that the server only uses the received data as identifier. All actions are completely independent to the web application.

In addition, we may assume that the received message have the same anchor form as discussed for the plugin. The content part from the anchor needs to be in a uniform format that has been generated by the plugin. So even the browser type does not matter to the server. The server has to be able to parse the content package and to create a new one that can be parsed by our plugins.

Social Weaver – Script Support The support for external scripts is essential for a generic usage of Social Weaver. The reason why script support is extracted into its own section is that it should be decoupled from the server and plugin that were discussed before. The underlying problem is the problematic identification of elements of a web view. There is simply no generic way of identifying elements in the users view across all web sites and applications. For that reason we need an extendable method to support more websites and applications. This could even mean that third parties could support their own systems by just adding the script without the need to modify Social Weaver directly. Let us consider the Google calendar application as example – see also Sect. 5. Assume a case where we want to match the same appointment field across different user sessions, which brings the problem that there is no identifier for the element itself. To the user it is obvious to identify it because of the appointment name, date and time. And those parameters could be just the information we need to extract into our script.

The usage of scripts should be related to one or a set of URLs. This affects mostly the root URL of a server. However, it might be used for sub parts of a web page or application. A set of URLs could be used for scripts that are applicable for many websites. The workflow for using a script when the default matching procedure that comes with the plugin is quite straightforward. Whenever opening a new URL then the plugin should check whether there is a script for that case and depending on the search results proceed with the script or default matching procedure.

5 Case Study Example

This section leads us through a real example where Social Weaver is being used. It is explained which components are used in what situations and how they interact with each other. We use Google Calendar as basis for the scenario. Google Calendar (GCal) [31] is free service for time management or in other words an electronic calendar. In the following context, GCal describes the web application that is accessible with any browser. The particular reason why we use GCal as testing scenario is that it is a freely available web application with shared data across user sessions. Such data can be a single appointment or a entirely shared calendar. Even though the HTML code differs for such data, the equality is clear to the user. The challenges with GCal are the differing HTML code for equal elements across user sessions.

The following explanations are based on a scenario with two users, i.e., Alice and Bob, who both are running the Social Weaver plugin in Firefox and are connected to the same Web Service, which means they share the same Social Weaver session. Additionally they obviously need a shared Google calendar. For accessing the calendar, they use the default web service provided by Google and no alternative client.

The scenario consists of the following steps. First, Alice weaves a comment box to an appointment in the shared calendar Then, Alice uses the comment box to leaves a reminder. Now, Bob logs in and comments Alice's reminder Now, we manually destroy the anchor directly in the database. Then, Alice recognizes this failure and re-links the comment box.

In the following two sub-procedures, update and matching, are explained. The reason why we handle those separately is that we use them more frequently in the whole process. That way we can just refer to them and keep the focus on the actual workflow.

Update Procedure The synchronization for Social Weaver is quite simple. On start up or whenever it is asked, the plugin sends two parameters in a JSON array to the web service using an authenticated POST request. Those parameters are the current URL and the time stamp of the last update. The web service uses that information to determine whether there are new and relevant anchors or not. In the positive case the anchors are returned. In Alice's case, nothing is returned since we have no marked elements.

What happens at the server with those data in detail? We use the time stamp of the last update and the current URL to create a query that receives only the corresponding anchors. Through a simple HTTP header authentication we know which user is getting access to the anchor data. It would be more an issue when having multiple users in different session in one Social Weaver context. However, such scenarios aren't covered in this thesis.

Matching Procedure When we use the term matching procedure it means that existing anchors are visualized to the user. Before every matching procedure, we assume that an update is triggered to ensure that the newest data is being used. Beyond the update procedure there is no need communicate with the server. When the user opens a new URL, it triggers the matching procedure. The plugin searches its local content whether there are some anchors for this URL. In a positive case the content is visualized to the browser view. At this point Alice would receive nothing from the plugin since no anchors exist for `www.cal.google.com`. The way in which anchors are retrieved from the plugin is quite similar how it is done at the back end. The major difference is that we do not use any time stamps at this time. There is no need for that since we assume everything is up to date after the update procedure.

Scenario Execution Now that we learned about the two sub-procedures, we are able to start with the actual scenario. First step is going to be that Alice weaves a comment box to an appointment. Alice enters `www.cal.google.com` which first of all triggers an update. Afterwards potentially new anchors would be displayed in the browser - which is not the case right now. Now Alice is able to mark an appointment – see Figure 6. By clicking an element, she appends a comment box. In the background, the plugin runs the script or scripts that are related to the URL to define an identifier for the element. Using this identifier, the content-data for the comment box and the current URL the plugin creates a message in JSON format and passes it to the server.

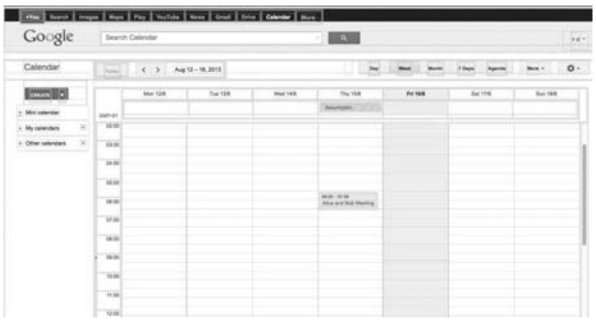


Fig. 6. Activated marking mode.

The content for the comment box in this case is just a link. We use an external comment system that is injected as HTML code. The message is passed as an authenticated POST REST request. The web service performs some checks before the anchor is persisted. For instance, it could be the case that there is already an anchor for exactly this element (because another user created one in the meantime or the element identifier is not unique in this context). In our scene, everything works out fine and the web service stores the new anchor in the PostgreSQL database. The web service returns a positive status code to the plugin. This again triggers the previously discussed update and matching procedure. Alice sees her comment box attached to the appointment after it is persisted on the server. It is not possible that the plugin creates locally new anchors that do not exist on the server. Finally, it is possible for Alice to use the comment box. This step is very simple. As we already mentioned the comment box is an external service that is only injected by a link into our system. Therefore, Alice can add a comment without any consequences to our system at all.

Now it is Bobs turn. This process is quite similar and partially redundant to what happened when Alice created the anchor. For that reason we do not go into detail like we did for the first step. Bob opens the Google Calendar website, which triggers the update and matching procedure for this URL. Since there is an existing anchor now - Bob's plugin receives the data for displaying the comment box entered by Alice – see Figure 7.

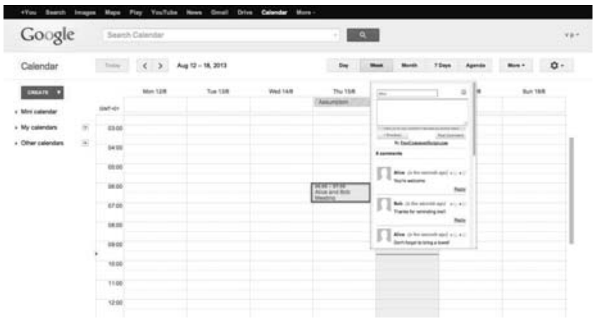


Fig. 7. Comment box woven into Google Calendar.

The last two steps are getting more interesting again. We simulate a evolution of a website that destroys our anchor mechanism. That can happen very easily depending on the type of script we are using or how fast the webpage evolves. What we do is to modify the element identifier directly in the database. This way it becomes impossible to match this anchor for the given URL. So Alice visits her calendar to check whether Bob has reacted to her reminder. Again an update and matching procedure is started. The update works seamlessly, but an error occurs while the matching progresses. The plugin runs the script to determine the element that belongs to the element id - but with no success.

For that case, the plugin enables Alice to re-link the content to the correct element or as in this case - appointment. The plugin performs the two following steps. First, a new anchor element is created which is basically a copy to the old one but with a correct element-id. This step is identical to the above-described procedure when Alice weaved the comment box into an appointment the first time. Then, we need to remove the broken anchor from the server. This is done by sending a remove REST request to the server including the old element identifier. After these steps are finished, it is necessary to run the update and matching procedures again. Now Alice and Bob are able once again to use the comment box. Re-linking an anchor does not necessarily has to be due to an error or web page evolution. For instance, if Bob changed the time of the appointment - the anchor would not work either. In this case, a re-link would solve the problem as well.

6 Further Work

Social Weaving can be extended with a workflow support. Since some problems with user interfaces appear only in certain situations that depend on the workflow context it would be a helpful feature to keep track about this information. Our intended approach is to reuse the code of the capture-and-replay component of an open-source test tool for this purpose. In addition, it would be interesting to go deeper into the idea of an automatic script generator. Instead of the user thinking about the architecture of the web environment, it is be possible to determine the needed information automatically. This would be a great acquisition, since no manual configuration for new environments would be necessary.

7 Related Work

Aspect-oriented programming [22] is about maintaining crosscutting concerns as capsules and weaving them into otherwise exsting code. Our weaving of software feature directly corresponds to weaving aspects in aspect-oriented programming. In our approach, the features are distributed by the end-user, not the programmer, because we follow an end-user approach.

The reverse engineering tool Revangie [12] [14] inspects the web site generated code, i.e., it is source code independent. Furthermore, not only static web sites are maintained but also dynamic ones. Revangie has a complex way of analyzing web sites. It is using the form-oriented user interface model [13], which is a graph that contains all information about the pages and additionally relationships between server-side actions and pages. Revangie addresses many problems that Social Weaving has to deal with as well, e.g., issues like screen classification or targets identification. The main difference to Social Weaving is that the whole web site code is analyzed.

The platform Platform for Education of Actual Software Engineering (PEASE) [15] is a domain-specific example of an application design that thoroughly combines process automation with social software features.

8 Conclusion

We have presented an approach of enterprise application integration by means of a concrete technological vision. The presented technology, the Social Weaver platform, enables the end-user to weave social software features into existing ERP system at the outermost tier of user dialogs. This way the technology yields a yet missing piece in the enterprise application integration puzzle. We have explained the architecture and realization of the platform and have discussed its usefulness in terms of concrete use cases.

References

1. K. Andrews, F. Kappe, H.A. Maurer. Hyper-G and Harmony – towards the Next Generation of Networked Information Technology. In Proceedings of CHI'95 – the 6^{th} Conference on Human Factors in Computing Systems, Conference Companion – Mosaic of Creativity, ACM Press, 1995.
2. C. Atkinson and D. Draheim. Cloud Aided-Software Engineering – Evolving Viable Software Systems through a Web of Views. In: Software Engineering Frameworks for the Cloud Computing Paradigm. Springer, 2013.
3. D. Auer, D. Draheim, V. Geist, T. Kopetzky, J. Kng. Towards a Framework and Platform for Mobile, Distributed Workflow Enactment Services – On a Possible Future of ERP Infrastructure. In: Innovation and Future of Enterprise Information Systems. Lecture Notes in Information Systems and Organisation, no. 4, Springer, 2013.
4. T. Bell, K.M. Shegda, M.R. Gilbert, K. Chin. Magic Quadrant for Enterprise Content Management. Gartner RAS Core Research Note G00206900. Gartner, November 2010.
5. R. Breu, B. Agreiter, M. Farwick, M. Felderer, M. Hafner, F. Innerhofer-Oberperfler. Living Models – Ten Principles for Change-Driven Software Engineering. In: International Journal Software and Informatics, vol. 5, no. 1–2, 2011.
6. B. Leuf, W. Cunningham. The Wiki Way – Quick Collaboration on the Web. Addison-Wesley, April 2001.
7. D. Draheim, T. Kopetzky, J. Küng. How to Make Mobile BPM Robust and Intelligent. In (Layna Fischer, Editor): Intelligent BPM - 2013 BPM and Workflow Handbook. Future Strategies, Workflow Management Coalition, 2013.
8. D. Draheim. Towards Total Budgeting and the Interactive Budget Warehouse. In: Innovation and Future of Enterprise Information Systems. Lecture Notes in Information Systems and Organisation, no. 4, Springer, 2013.
9. D. Draheim. Smart Business Process Management. In (Layna Fischer, Editor): 2011 BPM and Workflow Handbook, Digital Edition. Future Strategies, Workflow Management Coalition, February 2012.
10. D. Draheim. Business Process Technology – A Unified View on Business Processes, Workflows and Enterprise Applications. Springer, September 2010.
11. D. Draheim, O. Mangisengi. Integrated Business and Production Process Warehousing. In (David Taniar, Editor): Progressive Methods in Data Warehousing and Business Intelligence - Concepts and Competitive Analytics. IGI Global publication, 2009.
12. D. Draheim, C. Lutteroth, and G. Weber. A Source Code Independent Reverse Engineering Tool for Dynamic Web Sites. In: Proceedings of CSMR 2005 – 9^{th} European Conference on Software Maintenance and Reengineering. IEEE Press, March 2005.
13. D. Draheim, G. Weber. Form-Oriented Analysis - A New Methodology to Model Form-Based Applications. Springer, 2005
14. D. Draheim, C. Lutteroth, and G. Weber. Generator Code Opaque Recovery of Form-Oriented Web Site Models. In: Proceedings of WCRE 2004 – The 11^{th} IEEE Working Conference on Reverse Engineering. IEEE Press, 2004.
15. D. Draheim. A CSCW and Project Management Tool for Learning Software Engineering. In: Proceedings of FIE 2003 – Frontiers in Education: Engineering as a Human Endeavor. IEEE Press, 2003.

16. N. Drakos, J. Mann, A, Sarner. Magic Quadrant for Social Software in the Workplace, report no. ID:G00236025. Gartner Group, September 2012.
17. The Economist – Special Report. Data, Data Everywhere – A Special Report on Managing Information. Reprint, The Economist Newspaper Ltd., 27^{th} February 2010, pp. 2–13.
18. M. Felderer, R. Ramler. Experiences and Challenges of Introducing Risk-Based Testing in an Industrial Project. In: Proceedings of SWQD'13 – the 5^{th} International Conference on Software Quality – Increasing Value in Software and Systems Development, LNBIP, Springer, 2013.
19. M. Felderer, A. Beer. Using Defect Taxonomies to Improve the Maturity of the System Test Process – Results from an Industrial Case Study. In: Proceedings of RE'13 – the 21^{st} IEEE International Requirements Engineering Conference, IEEE, 2013.
20. J. Grudin. Computer-Supported Cooperative Work – History and Focus. Computer, vol. 27, no. 5, IEEE Press, 1994.
21. L. Haas. Building an Information Infrastructure for Enterprise Applications. In (D. Draheim, G. Weber, Editors): Trends in Enterprise Application Architecture, LNCS 3888, 2006.
22. G. Kiczales et. al. Aspect-Oriented Programming. In: Proceedings of ECOOP'97 – the 11^{th} European Conference on Object-Oriented Programming, LNCS 1241, Springer, June 1997.
23. C. Lettner, C. Hawel, T. Steinmaurer, D. Draheim. Complex Event Processing for Sensor Based Data Auditing. In: Proceedings of ICEIS'2008 – the 10^{th} Intl. Conf. on Enterprise Information Systems, Springer, 2008, pp. 485–491.
24. H. Lieberman, F. Paterno, and V. Wulf (Eds.). End-User Development. Human Computer Interaction Series, Springer, 2006.
25. O. Mangisengi, M. Pichler, D. Auer, D. Draheim, and H. Rumetshofer. Activity Warehouse – Data Management for Business Activity Monitoring. In: Proceedings of ICEIS 2007 – the 9^{th} Intl. Conf. on Enterprise Information Systems, June 2007.
26. R. Maier, T. Hädrich, R. Peinl. Enterprise Knowledge Infrastructure. Springer, 2005.
27. D. McCracken, A. Newell. The ZOG Human Computer-Inferface System – A Renewal Proposal to the Office of Naval Research for the period 1st March 1983 to 1st October 1984, Renewal of Grant N00014-76-0874: ZOG: An Interactive Programming Environment Using a Graph-Structured, Rapid-Response Guidance System. Carnegie-Mellon University, May 1983.
28. I. Nonaka, H. Takeuch. The Knowledge-Creating Company – how Japanese Companies Create the Dynamics of Innovation. Oxford University Press, 1995.
29. V. Pekar. Social Weaver – A Platform for Weaving Web 2.0 Features into Webbeased Applications. Master Thesis, Faculty of Computer Science, Universtiy of Innsbruck, September 2013.
30. M. Schroeck. Analytics – The Real World Use of Big Data – How Innovative Enterprises Extract Value from Uncertain Data. IBM Global Services Business Analytics and Optimization Executive Report, IBM Institute for Business Value, Said Business School, University of Oxford, 2012.
31. `http://google.com/calendar`
32. `https://github.com/vikpek/SocialWeaver`