An Interoperability Infrastructure for Distributed Feed Networks

Fridolin Wild[1], Steinn E. Sigurðarson[1], Stefan Sobernig[1], Christina Stahl[1], Ahmet Soylu[2], Vahur Rebas[3], Dariusz Górka[4], Anna Danielewska-Tułecka[4], Antonio Tapiador[5]

[1] Vienna University of Economics and Business Administration, Austria
[2] Işik University, Istanbul, Turkey
[3] Tallinn University, Tallinn, Estonia
[4] AGH University of Science and Technology, Kraków, Poland
[5] Universidad Politécnica de Madrid, Spain

Abstract. Blogs have the affordance to become an integral part of teaching and learning processes as a vehicle for knowledge management. Open, flexible systems integrating blogs provide user-friendly, personalised microlearning environments while ensuring ubiquitous access. This paper concentrates on blog interoperability in an open space for learning and collaboration. As blogging is one of the most popular web-based forms of publishing today, there is a plethora of different blogging tools, feed readers, and aggregators, enabling information dissemination, filtering, and retrieval. Several different data structure and interaction standards emerged which make integration a real challenge. This paper aims at creating distributed microlearning environments basing on networks of integrated blogs, discusses problems of such an integration along with possible solutions, and proposes an architecture for loosely-coupled blog integration. The core of this infrastructure consists of a service interface for feed management that allows learners as well as facilitators to automatically set up channel structures for feed syndication, while effectively reducing the management efforts thereof through proper system support.

1 Motivation

The conglomerate of all blogs available online, the so-called ‘blogosphere’, has been certified to show a bursty evolution at least since 2001 [1], where an eruptive rise can be identified not only regarding metrics of scale but also with respect to deepening community structures and higher degrees of connectedness. As with July 2007, for example, Technorati is indexing 90 million blogs [2]: Blogging is obviously an increasingly popular phenomenon, although meta-studies reveal that between one half and two thirds of all blogs are abandoned within only two months after their creation [3]. One of the reasons that blogs became so attractive is their ease of use, removing barriers of technoliteracy from web self-publishing [4]. There is a plethora of web-publishing tools, allowing the user to choose from a large variety of (non-) commercial hosting services. Moreover, users can set up their own web-applications choosing from a rich portfolio of open- and closed-source products.

It is not surprising then, that blogs became vehicles for knowledge management to form an integral part of teaching and learning processes. Blogs can be used to organize lectures, seminars, and discussions both between teachers and students. Herring et al. [5] found in their study on blog genres that 57.5% of the blogs' authors investigated in a random sample of 203 blogs are students on a secondary and tertiary level. However, the major share of 70.4% of the blogs are personal journals rather than commentaries to external content. Similar results are reported in [6].

The reasons why people create and maintain blogs vary also include community building and social networking among the key motivations substantiated through empirical studies [7,8]. However, when deploying blogs in collaboration, many obstacles can be found that have not been solved so far. To facilitate "productive blog conversations" which are necessary in knowledge management and learning, "more carefully tailored socio-technical systems are needed", as De Moor and Efimova [9] claim. They identify the problem of notorious fragmentation of conversations to be responsible for (even the authors') difficulties in reconstructing conversations. Furthermore, they see (initial) response times as a problem that may slow down conversations. Another obstacle to productive conversations in collaboration processes is identified to be the low number of links to blog posts. Only 51.2% of all blogs [5] link to other blogs, only 53.7% link to other websites. 30.5% of all blogs do not link to anything at all (besides badges). The average entry received .3 comments, the majority of entries received none. Multimodality poses yet another problem: replies and comments are often distributed across comment fields, but can also be found in blogs of the repliers. Krause [10] identifies the fuzziness of the audience as a problem that may be responsible for failing discussion in his course experiment.

Looking more closely at why weblogs 'die', the general reasons for their abandonment differ for each phase of a blog's life-cycle. Especially in more mature phases, where revisions of tool choices are not so likely to be found, external causes are prevalent including changes in the social network of the author, e.g., changes in employment or family [3].

De Moor and Efimova [9] call for ways to represent the semantics and pragmatics of the blogosphere (e.g., visualisations) and "methods for gradually developing the required complex socio-technical systems around blogs, such as virtual communities" (i.e., collaborative management methods). However, we are not quite there: Interoperability between weblogs is not given. Interoperability requires the participating information systems not only to integrate on a communication and control layer specifying joint communication interfaces and protocols, not only to mutually share a fair amount of knowledge on their (meta-) data schemes and access options, but also to cooperate regarding application semantics and functionality (see [11]).

According to [12], there are nine different and incompatible versions of RSS. Besides problems of being well-formed, several other incompatible content-model changes have been applied throughout the version history of RSS. Furthermore, switching the abstract data-model between RDF and XML caused further confusion. Additionally, ATOM entered the field in 2003 competing with RSS (cf. [13]). Looking at interaction protocols, a similar picture can be assessed: besides a publish & subscribe pull-mode via HTTP, a set of APIs to implement a push-mode for data exchange causes simple syndication to become suddenly not so simple anymore. Access restriction and protection virtually does not exist, same applies for application functionality and semantics; task support is restricted to simple publishing features.

In Section 2, we are giving a short introduction into theoretic interoperability shortcomings, as can be derived from system integration research. In Section 3, we will carefully review related work on distributed weblog interoperability. In Section 4, we construct the missing link, an interoperability model for management of distributed weblog networks and demonstrate the feasibility with a prototype implementation. In Section 5, we will discuss the effects of the reference model and identify areas of future work.

2 Loosely-Coupling for Blogs

Interoperability is defined to be a property that emerges, when distinctive information systems (subsystems) cooperatively exchange data in such a way that they facilitate the successful accomplishment of an overarching task [14]. Interoperability therefore requires integration on different levels. System integration problems are very complex. Below we list typical weblog integration challenges adopted from Hohpe's and Woolf's [15] general proposition.

1. **Ultimate Holisticity:** The communication to be established is not only between two blogging systems, but also touches the surrounding e-learning systems. This has implications on whole system architecture.
2. **Limited Control:** There is often very limited control over the blogging application, for example when integrating a particular learning environment with an independent, packaged blogging application.
3. **Standards Fragmentation:** There is still the lack of interoperability between standards-compliant blogging applications (due to extensions, different interpretations, limited establishment of standards).
4. **Lack of Shared Meaning:** The data can be represented in XML (the lingua franca of system integration), but it is a rather shallow compatibility – the semantics vary tremendously from system to system.
5. **Speed of Change:** Applications and systems, even standards are constantly changing – maintaining such mixture can be challenging itself.

One solution to these problems can be found in the architectural principle of loosely-coupling. This approach leads to more stable and flexible applications. The principle is "to reduce the assumptions two parties make about each other when they exchange information" [15].

Loosely-coupling often involves specific design patterns, i.e., solution schemes for common problems that can be adopted to the specific situation. Many different kinds of design patterns exist. Following the idea of integration patterns, interoperability can be seen to be achievable on three different **architectural layers** [14]: on the presentation layer through the inclusion of the direct user-interfaces, by remoting in the form of shared functions, or as data integration and dissemination through the replication, retrieval, filtering, etc. of data stemming from other systems.

Blog interoperability can be achieved in two different **interaction styles**: push and pull. In the push case modifications are propagated by the application managing the changed blog. It sends a message with information on the modifications to the servers storing the subscribed blogs. In the pull case the blogging application checks regularly if there was a change in the subscribed blogs and filters for updated data. Pulling requires two steps of communication, while pushing maintains state information and sends data only when necessary: it preserves information about clients' interests and pushes only relevant information. Pull interactions require many queries without effect, as weblogs usually do not change very rapidly. As a consequence, pulling causes a larger communicative network load, especially when there is a larger number of clients. In the push approach, action is performed only when needed. The extensive performance analysis of push versus pull is described by [16]: they were able to show that for small temporal coherency requirements, pulling bears performance disadvantages.

Regarding **authentication and authorization**, the following statements can be derived in accordance with other claims for the social software landscape in general [17,18,19]. First, within an open learning situation, there is no need for explicit authentication, as the idea of public and distributed provision incorporates certain elements of anonymity and pseudonymity. Second, the idea of publicly providing feeds already goes against authorisation in traditional access control mechanisms: a publicly available feed is syndicated, i.e., actively subscribed via pull or push-upon-request mechanisms. There is no need to have distributed authorisation mechanisms that would grant system-external subjects access rights to system-internal objects. Access is only granted in consent and under the control of a system-internal, thus mediating subject. For these settings, capability-based credentials as such (when key- or ticket-bound, not name-bound) regulate access control better and can replace identity-based credentials of centralised authentication and authorisation protocols (cf. [18,20,21]).

3 Related Standards and Their Appliance

There are many attempts to bring interoperability to the blogosphere. These efforts are mainly targeted towards two levels of the interoperability stack: (meta-) data schemes and item collection exchange protocols between different blogging tools. The first kind came to birth with the RSS feeds formats and standards, the second relates to several emerging blogging application programming interfaces (APIs). In the following section, this related work will be investigated in three stages. First, we will analyse the existing standards for weblog (meta-) data and interaction protocols in more depth. Second, we will pragmatically investigate existing tools regarding their interoperability capabilities. Third, we will show that this existing work has several shortcomings if meant to become an interoperability-architecture for distributed blog systems necessary to support collaborative learning settings.

3.1 Feed Standards

Meta-data standards define (XML) languages for describing collections of web content items and the information related to them. Collections are represented as feeds (also called 'channels'). Content items are usually weblog posts, but also images, audio, video, or any resource published in the web. The origins of these formats date back as early as 1995 [13], although today only a few of them are still in use. Although there had been various proposals for meta-data standards for news synchronisation – [13] counts eleven different standards in altogether 30 different versions –, today there are only three main standards: RSS 1.0, RSS 2.0, and Atom.

RSS 1.0 was developed later than the precursors of RSS 2.0 and bases on a different technology: it deploys the resource description framework (RDF) of the W3C [22] and is not a simple XML derivative. It's more difficult to use, however, to the benefits of its expressiveness [13].

RSS 2.0, now courtesy of the Harvard University [23] and frozen since 2002, seems to be more popular, although it is less formally grounded and although the development process has been widely criticized. Some of the drawbacks from the interoperability point of view are that it is not in an XML namespace and its content model does not permit well-formed XML mark-up.

The **Atom Syndication Format** [24] is an XML-based document format, developed by the Atom Publishing Format and Protocol (atompub) IETF working group [23]. It is the result of a standardisation effort, being robust and complete. It was designed as to be most flexible which is accomplished through extensions. An extension that covers threading has been proposed [25], which seems to be appreciated as a "robust specification for how to syndicate comments" [26].

There is no short answer on which of these three alternatives to choose. All of them share a common core. Depending on what is to be achieved, however, there are use-case specific advantages of the one or other format. RSS 1.0 is semantic-web technology and achieves more expressiveness through the separation of document format and content such that it allows for subsequent reasoning in the available (meta-) data. RSS 2.0 offers more simplicity in the specification (through fuzziness) and some advantages for provision of multimedia data, e.g., for podcasting. Atom's development process is more open; the standard itself is also formally and completely defined. Transformations between these formats can be easily executed, although the interpretation of the transmitted data might then be more dependent on the interpreting applications.

Besides mark-up languages for feeds and items (including possibilities to annotate the origins of items), a meta-data format for exporting lists of feeds has been proposed by Winer [27]: the Outline Processing Markup Language (OPML). With these outlines, a list of feeds can be exchanged in a structured format, so-called blogrolls [28].

3.2 Blog Interaction Standards

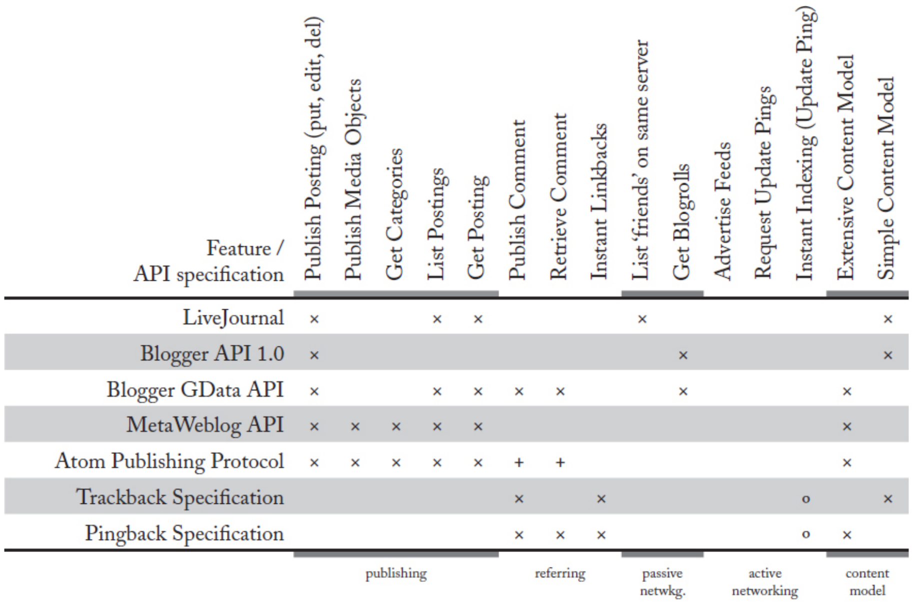
Several specifications for different kinds of blog interaction have been proposed in the recent years. The first xml-rpc specification for publishing blog items was developed in 1998 [29]. Subsequently, Winer was involved in developing another xml-rpc plus soap interface [30].

The first well-known API for publishing was developed by LiveJournal (around 1999). It passed simple key-value pairs for every request and response over HTTP [31]. The next big API was developed for Blogger, also based on xml-rpc. However, its metadata for entries cannot be extended [32]. In response to that problem, UserLand created the MetaWeblog API [34]. Both, the blogger API and the MetaWeblog API are widely used. In 2007, Blogger replaced the blogger API by the generic Google Data API. Sixapart's Moveable Type adds further extensions on top of the blogger and MetaWeblog API: most notably this is the trackback extension proposed in [33]. A similar, but less spam-vulnerable mechanism for informing about citations and replies has been proposed in [35] – the pingback specification. Both pingback and trackback have been re-used (not to say: abused) to support indexing of blog postings in aggregator services on the web (for an overview on pingback-based indexing services see [36,37] for an example on topic-oriented trackback-based indexing services). Starting as EchoAPI in 2003, the Atom Publishing Protocol emerged as an attempt to unify the various APIs and standardise it through the IETF. The AtomPP specification is still being developed today [38].

For brevity reasons, the following analysis concentrates on an overview of the differences. More details on the interfaces can be found in [39].

As can be seen from the API comparison in Figure 1, the most extensive protocol is the atom publishing protocol that realises almost all identified capabilities needed for publishing and retrieving collections and entries, competing hard with Blogger's new GData API. At the same time, the comparison table shows clear shortcomings: even simple management functionalities that support indexing and distribution of blogs are only supported via external specifications such as pingbacks and trackbacks.

Networking interactions seem to be the major shortcoming of today's APIs. Passive networking interactions are only supported by early APIs and the GData API – and even there only to a limited amount (either listing friends or getting blogrolls of users). There is no support at all for active networking interactions, which are essential for articulation work in collaborative learning processes: only the abuse (i.e., unintended use of the specification in platforms like technorati) of trackbacks and pingbacks allows for instant indexing,proactive feed management (advertising feeds, requesting subscriptions) between users is overally not



Feature / API specification	Publish Posting (put, edit, del)	Publish Media Objects	Get Categories	List Postings	Get Posting	Publish Comment	Retrieve Comment	Instant Linkbacks	List 'friends' on same server	Get Blogrolls	Advertise Feeds	Request Update Pings	Instant Indexing (Update Ping)	Extensive Content Model	Simple Content Model
LiveJournal	×			×	×				×						×
Blogger API 1.0	×									×					×
Blogger GData API	×			×	×	×	×			×				×	
MetaWeblog API	×	×	×	×	×									×	
Atom Publishing Protocol	×	×	×	×	×	+	+							×	
Trackback Specification						×		×					o		×
Pingback Specification						×	×	×					o	×	
	publishing					referring			passive netwkg.		active networking			content model	

Fig. 1. Overview on Differences between the APIs.

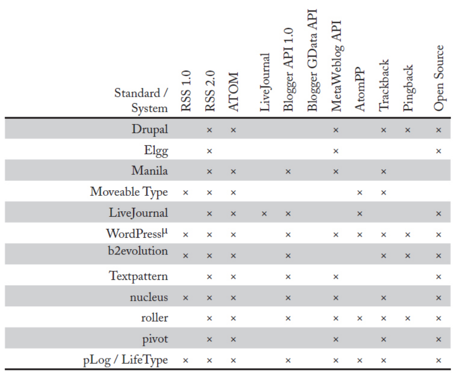
supported. The core publishing process is fully supported by several of the state-of-the-art APIs, whereas comment management in distributed settings can only by achieved by deploying trackbacks and pingbacks in addition to existing APIs.

3.3 Existing Blogging Tools

The number of **blog applications** is huge and still growing. Many of them are provided as free hosting services (like wordpress.com). The collection in [40] lists popular tools for creating blogs, mobile blogs, audio blogs (a.k.a. podcasts), video blogs (a.k.a. vblogs), and aggregators to read blog feeds. Popular blogging systems are: drupal, elgg, manila, MovableType, LiveJournal, WordPress / WordPress MultiUser, b2evolution, textpattern, nucleus, roller, pivot, and pLog (today re-branded to 'LifeType') [42,41]. All of these, except Manila and Moveable Type, are open-source applications.

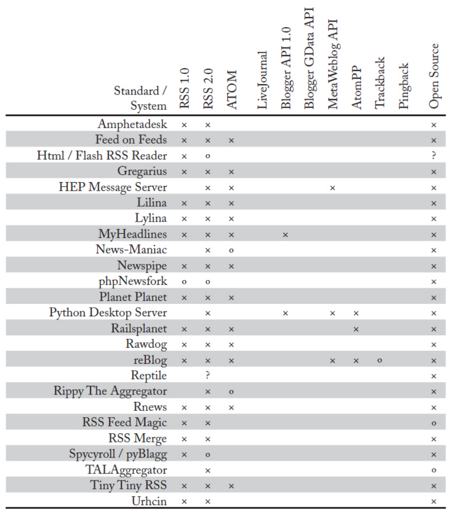
As can be seen from the overview in Table 2, standard support among the blog systems is quite heterogeneous. All systems support RSS 2.0, closely followed by atom, whereas RSS 1.0 feeds are only provided by a limited number of systems. Among the APIs, the blogger and the MetaWeblog API compete closely; support for the Atom Publishing Protocol, however, is growing. Trackbacks are supported by many systems, pingbacks are already offered by several ones.

A **blog aggregator** or a feed reader is the common solution to consuming syndicated content. It can monitor many sites and sources providing near real-time updates to one location. Aggregators can be classified into personal and



Standard / System	RSS 1.0	RSS 2.0	ATOM	LiveJournal	Blogger API 1.0	Blogger GData API	MetaWeblog API	AtomPP	Trackback	Pingback	Open Source
Drupal		×	×				×		×	×	×
Elgg		×					×				×
Manila		×	×		×		×		×		
Moveable Type	×	×	×					×	×		
LiveJournal		×	×	×	×			×			×
WordPress[μ]	×	×	×		×		×	×	×	×	×
b2evolution	×	×	×		×				×	×	×
Textpattern		×	×		×		×				×
nucleus	×	×	×		×		×		×		×
roller		×	×		×		×	×	×	×	×
pivot		×	×				×		×		×
pLog / LifeType	×	×	×		×		×	×	×		×

Fig. 2. Standard Support of Blogging Tools (Overview).



Standard / System	RSS 1.0	RSS 2.0	ATOM	LiveJournal	Blogger API 1.0	Blogger GData API	MetaWeblog API	AtomPP	Trackback	Pingback	Open Source
Amphetadesk	×	×									×
Feed on Feeds	×	×	×								×
Html / Flash RSS Reader	×	o									?
Gregarius	×	×	×								×
HEP Message Server		×	×				×				×
Lilina	×	×	×								×
Lylina	×	×	×								×
MyHeadlines	×	×	×		×						×
News-Maniac		×	o								×
Newspipe	×	×	×								×
phpNewsfork	o	o									×
Planet Planet	×	×	×								×
Python Desktop Server		×			×		×	×			×
Railsplanet	×	×	×					×			×
Rawdog	×	×	×								×
reBlog	×	×	×				×	×	o		×
Reptile		?									×
Rippy The Aggregator		×	o								×
Rnews	×	×	×								×
RSS Feed Magic	×	×									o
RSS Merge	×	×									×
Spycyroll / pyBlagg	×	o									×
TALAggregator		×									o
Tiny Tiny RSS	×	×	×								×
Urhcin	×	×									×

Fig. 3. Overview on Standard Conformance of Aggregators.

community aggregators. Among personal aggregators, we can find web-based tools and desktop tools. Personal aggregators are intended for personal use. A user subscribes to her feeds of interest and manages them in a way similar to a mail client. Analogously to mail clients, the application can either be installed on a web server (then accessed through a browser) or can be provided as a desktop application, independent or integrated with a browser. Desktop aggregators are, for example, Personal Learning Environment (PLEX) and the Firefox derivate Flock. For here, we will focus on web-based feed readers.

Community aggregators gather web feeds relevant to a group of interest using a list of feeds. They aggregate, for example, all feeds from a course. They merge all items in the feeds, can provide this feed again, and usually display all items on a web page. [43] lists 13 different web-based aggregators. Additionally, [44] lists another set of aggregators. No longer existing aggregators were excluded from the analysis, two more aggregators were added.

As can be seen from the overview in Table 3, many of the aggregators systems support all important content standards (12/25 support all). Regarding APIs, however, the overview shows clearly that almost no system supports an API. This is even more astonishing as the APIs contain several synchronisation routines and abuses of trackback and pingback are very popular with commercial online services such as technorati to realise indexing updates. Few systems contained proprietary APIs, that can be used to synchronise the online aggregator with a desktop application. Regarding subscription management and feed advertisement, none of the system provides any support.

3.4 Current Shortcomings

As we have shown through our feed standard analysis above, no clear recommendation can be made in favour of one of the three major feed data standards – RSS 1.0, RSS 2.0, or Atom. All of them have different advantages, all of them are extensible, and all of them can – eventually with interpretation disadvantages on the application sides – be transformed into each other.

By investigating publishing APIs and their enhancements that have been brought forward in recent years, we come to the following conclusion: Standard interfaces exist that support the core publishing process (as necessary for remote controlling a blog with a desktop editor), support for networking interactions is virtually not available. Trackback and pingback, alas, have made the first attempts into that direction, although they were not originally designed for it. We conclude that this major shortcoming in the area of active networking needs to be solved in order to comply with our requirements which we sketched in the scenario section of this contribution.

Looking at state-of-the-art blog applications, we see that many of them support an extensive set of both API and feed data standards, however, naturally reflecting their lack of networking support. Aggregators usually do not support any API standard, which – in case of pingbacks and trackbacks – is especially astonishing. Most of them even lack in support of content standards which prevents them from fulfilling the job they were designed for. Again and even more astonishing, aggregators lack capabilities for subscription management and feed advertisement. To conclude: as our three analysis sections show us, the major shortcomings lie in realising interoperability for feed management.

4 The Missing Link: The FeedBack Specification

The process of collaborating via blogs can be divided into two independent substeps, i.e., the management of feeds and communication channels including authorisation (the articulation work) and the exchange of items or item collections (the materialisation, the content transmission itself). The following section describes the missing link, a specification for managing feed subscriptions in a distributed setting which complements existing standards as analysed in the previous section.

Therefore, we describe data exchange, interaction steps, and state-transitions between two non-identical blog systems and their users, when communicating management information about particular feeds and specific items in these feeds. This model realises an interface specification for user-centred distribution of feed aggregation activities which is both prerequisite and basic infrastructure for blog-based collaboration.

Aggregation services are already ‘abusing’ the pingback specification in so far as they are using pingback xml-rpc calls to inform about new and updated items and no longer inform about replies to existing blog postings. However, at the same time, there are no standardised options to inform a system about the existence of a feed and about updates which would enable more efficient management in synchronisation.

The proposed specification ‘FeedBack’ deploys a set of xml-rpc calls to transport blog management information from one system to another. It is light-weight in so far, as implementation is made as easy as possible and dependencies on other components are reduced to a minimum. The whole communication process apes human behaviour and shifts control to the user wherever possible.

4.1 Communication Process

There are three important steps when managing feeds (see Figure 4). First, the communication partner has to be informed about the opportunity of subscribing to a blog. Therefore, the feed of the partner is fetched, parsed for the xml-rpc endpoint, and the feedback.offer() remote procedure call is executed.

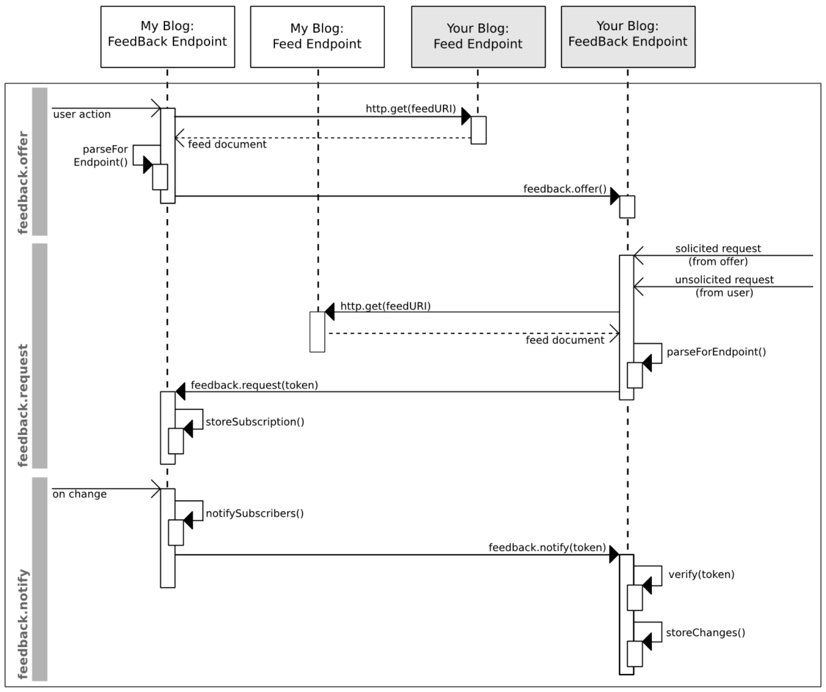
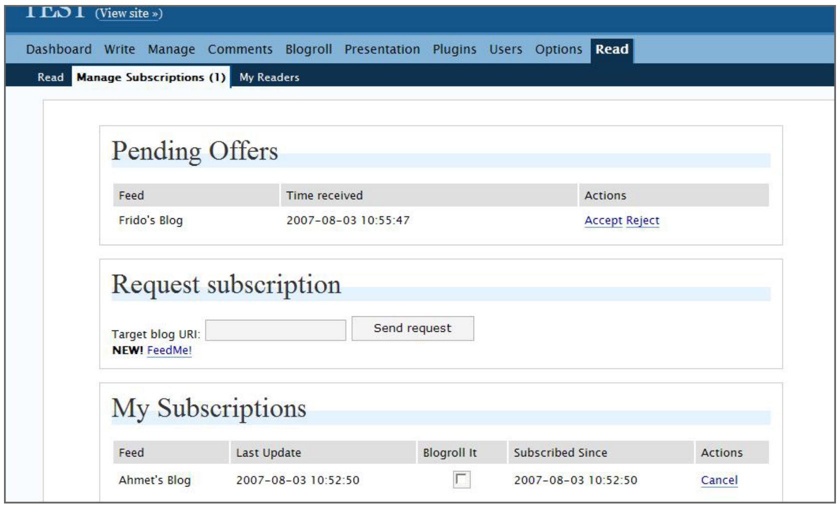


Fig. 4. The Communication Process in ‘FeedBack’.

Second, this communication partner has to actively acknowledge whether she wants to be informed about future updates of the advertised blog, i.e. to confirm the subscription. This can also be done without receiving an offer, i.e., from an unsolicited request. Therefore, a token is generated and a feedback.request(token) call executed. Finally, the communication partner has to be pinged about every update that is available in order to allow for efficient synchronisation – using the feedback.notify(token) call.

4.2 Prototype

We have prototypically tested this specification with implementations for a standard blogging system, WordPress and WordPress MU, and a learning management system, IVA. Currently, we are working on additional implementations, a public release of the WordPress module (see Figure 5) is being prepared.



Pending Offers

Feed	Time received	Actions
Frido's Blog	2007-08-03 10:55:47	Accept Reject

Request subscription

Target blog URI:
NEW! FeedMe!

My Subscriptions

Feed	Last Update	Blogroll It	Subscribed Since	Actions
Ahmet's Blog	2007-08-03 10:52:50	☐	2007-08-03 10:52:50	Cancel

Fig. 5. Snapshot Wordpress with the FeedBack Module.

5 Conclusion

Within this contribution we have extensively analyse the status quo of the technology behind the blogosphere, have identified a major shortcoming from the perspective of its applicability in collaborative learning situations, and have proposed a solution to solve it: FeedBack complements existing standards with a specification on how to support management processes for advertising of, subscription of, and update notifications on existing blog feeds. The prototype of this emerging network shows us that it is feasible. Filling this space with life to turn it into a sustainable place sets yet another challenge for the future.

References

1. Kumar, R.; Raghavan, P.; Novak, J.; Tomkins, A. (2003): On the Bursty Evolution of Blospace, Proceedings of the WWW2003, Budapest, ACM Press, pp. 568–576

2. Technorati (2007): Technorati: About Us, http://www.technorati.com/about/, accessed: July 5th, 2007
3. Gurzick, D.; Lutters, W. (2006): From the Personal to the Profound: Unterstanding the Blog Life Cycle, In: Proceedings of the CHI 2006, Montreal, Canda, ACM Press
4. Tepper, Michele (2003): The Rise of Social Software, In: netWorker **7(3)**, ACM Press, pp. 18–23
5. Herring, S.; Scheidt, L.; Bonus, S.; Wright, E. (2004): Bridging the Gap: A Genre Analysis of Weblogs, In: Proceedings of the 37th Hawaii International Conference on System Sciences (HICSS-37), IEEE Computer Society Press
6. Schmidt, J.; Mayer, F. (2007): Wer nutzt Weblogs für collaborative Lernund Wissensprozesse?, In: Dittler, Kindt, Schwarz (Eds.): Online-Communities als Soziale Systeme, Waxmann, New York/München/Berlin
7. Nardi, B.; Schiano, D.; Gumbrecht, M.; Swartz, L. (2004): Why We Blog, In: Communications of the ACM **47(12)**, ACM Press, pp. 41–46
8. Schiano, Diane; Nardi, Bonnie; Gumbrecht, Michelle; Swartz, Luke (2004): Blogging by the Rest of Us, In: Proceedings of the CHI 2004, Vienna, Austria, ACM Press, pp. 1143–1146
9. De Moor, A.; Efimova, L. (2004): An Argumentation Analysis of Weblog Conversations, In: In: Aakhus, Lind (eds.): Proceedings of the 9th International Working Conference on the Language-Action Perspective on Communication Modelling (LAP 2004), New Brunswick, USA, pp. 197–211
10. Krause, S. (2004): When Blogging Goes Bad: A Cautionary Tale About Blogs, Email Lists, Discussion, and Interaction, http://english.ttu.edu/KAIROS/9.1/praxis/krause/index.html, last access: July 5th, 2007
11. Kosanke, K. (2005): ISO Standards for Interoperability: a Comparison. In: Pre-Proceedings of the INTEROP-ESA 2005, pp. 59–67, Geneva.
12. Pilgrim, M. (2004): The myth of RSS compatibility, http://diveintomark.org/archives/2004/02/04/incompatible-rss, last access: July 5th, 2007
13. Wittenbrink, H. (2005): Newsfeeds mit RSS und Atom, Galileo Press, Bonn
14. Wild, Fridolin; Sobernig, Stefan (2006): Interoperability Framework Draft for the Distributed Open Virtual Learning Environment, Deliverable D3.1, iCamp Project
15. Hohpe, G., Woolf, B., (2003), Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions, Addison-Wesley Professional
16. Deolasee, P., Katkar, A., Panchbudhe, A., Ramamritham, K., Shenoy P., (2001), Adaptive Push-Pull: Disseminating Dynamic Web Data, ACM, http://www10.org/cdrom/papers/pdf/p269.pdf (accessed Jan, 2007)
17. Downes, S. (2005): Authentication and Identification, In: Instructional Technology & Distance Learning, Vol. 2, No. 10, DonEl Learning Inc.
18. Bhatti, R.; Bertino, E.; Ghafoor, A. (2007): An Integrated Approach to Federated Identity and Privilege Management in Open Systems, In: Communications of the ACM 50(2), ACM Press, pp. 81–87
19. Wikipedia (2007): Authorization, http://en.wikipedia.org/wiki/Authorization, last-access: June 25th, 2007
20. Miller, M.; Yee, K.; Shapiro, J. (2003): Capability Myths Demolished, Technical Report SRL2003-02, Systems Research Laboratory, Johns Hopkins University
21. Levy, H. (1984): Capability-Based Computer Systems, Digital Equipment Corporation
22. Herman, I.; Swick, R.; Brickley, D. (2007): Resource Description Framework (RDF), W3C, http://www.w3.org/RDF/, last access: July 5th, 2007
23. Winer, D. (2003): RSS 2.0 at Harvard Law, http://cyber.law.harvard.edu/rss/rss.html, last access: July 5th, 2007

24. Nottingham, M.; Sayre, R. (2005): The Atom Syndication Format, http://www.ietf.org/rfc/rfc4287.txt (accessed Jan, 2007)
25. Snell, J. (2006): Atom Threading Extensions, http://www.ietf.org/rfc/rfc4685. txt (accessed Jan, 2007)
26. Reese, B. (2006): Standardizing the Atom Thread Extension, http://www. majordojo.com/atom/standardizing_the_atom_thread_extension.php (accessed Jan, 2007)
27. Winer, D. (2000): OPML 1.0 Specification, http://www.opml.org/spec, last access: June 26th, 2007
28. Kalz, M.; Specht, M.; Klamma, R.; Chatti, M.; Koper, R. (2007): Kompetenzentwicklung in Lernnetzwerken für das lebenslange Lernen, In: Dittler, Kindt, Schwarz (Eds.): Online-Communities als Soziale Systeme, Waxmann, New York/München/Berlin
29. Winer, D. (1999): XML-RPC Home Page, http://www.xmlrpc.com/, last access: July 5th, 2007
30. Radke, A.; Simmons, B.; Winer, D. (1999): Manila-RPC interface, http://www.xmlrpc.com/manilaRpc, last access: July 5th, 2007
31. Livejournal (2007): Client/Server Protocol, http://www.livejournal.com/doc/server/ljp.csp.protocol.html, last access: July 5th, 2007
32. Williams, E. (2001): Blogger API, http://www.blogger.com/developers/api/1_docs/, last access: June 25th, 2007
33. Trott, B.; Trott, M. (2002): TrackBack Technical Specification, Sixapart, http://www.sixapart.com/pronet/docs/trackback_spec, last access: July 5th, 2007
34. Winer D. (2002): RFC: MetaWeblog API, http://www.xmlrpc.com/metaWeblogApi, last access: August 3rd, 2007
35. Langridge, S.; Hickson, I. (2002): Pingback 1.0, http://www.hixie.ch/specs/pingback/pingback, last access: July 5th, 2007
36. Pchere (2005): One Click Multiple Blog Services Pinging Tools, http://www. quickonlinetips.com/archives/2005/09/one-click-multiple-blog-servicespinging/, last access: June 27th, 2007
37. Paquet, S.; Pearson, P. (2004): A Topic Sharing Infrastructure for Weblog Networks, Proceedings of the Second Annual Conference on Communication Networks and Services Research (CNSRŠ04), IEEE Computer Society Press
38. Gregorio, J.C.; de hOra, B. (2007): The Atom Publishing Protocol, http://ietfreport.isoc.org/all-ids/draft-ietf-atompubprotocol-15.txt, IETF
39. Wild, F. (2007): An Interoperability Infrastructure for Distributed Feed Networks, Deliverable D3.3, iCamp Consortium
40. eLearning Centre (2006): Products & Services: Blogging Tools, eLearning Centre, Learning Light, http://www.e-learningcentre.co.uk/eclipse/vendors/ weblogging.htm, last access: July 5th, 2007
41. Bauer, E. (2004): An Overview of the Weblog Tools Market, http://www. elise.com/ web/a/an_overview_of_the_weblog_tools_market.php, last access: July 5th, 2007
42. Farmer, J. (2005), Centred Communication: Weblogs and aggregation in the organisation, Blogtalk Downunder, May 19-22 Sydney, http://incsub.org/blogtalk/?page_id=54 (accessed Jan, 2007)
43. News on Feeds (2007): News Aggregators, http://www.newsonfeeds.com/faq/ aggregators, last access: July 27th, 2007
44. Hebig, H. (2005): RSS Feed Reader / News Aggregators Directory, http://hebig.org/blogs/archives/main/000877.php#scripts, last access: July 27th, 2007