Blogging in the Smart Conference System

Dmitry G. Korzun, Ivan V. Galov
Department of Computer Science, PetrSU
Petrozavodsk, Russia
{dkorzun, galov}@cs.karelia.ru

Alexey M. Kashevnik, Nikolay G. Shilov
Computer-Aided Integrated Systems Lab,
SPIIRAS, St.-Petersburg, Russia
{alexey, nick}@iias.spb.su

Kirill Krinkin, Yury Korolev
Saint-Petersburg Electrotechnical University
St.-Petersburg, Russia
kirill.krinkin@fruct.org, yury.king@gmail.com

Abstract

Smart spaces provide an infrastructure to use a shared view of dynamic resources and context-aware services within a distributed application. There is, however, no standard scheme for integration of several independent applications. In this paper we analyze two particular smart space-based applications: Smart Conference System that assists conferencing process online and SmartScribo that provides advanced access to the blogosphere. For this reference use case we propose a scheme of their integration that employs a smart space agent to build an overlay smart space on top of the origin applications. The scheme is based on an integrated architecture of both applications and on an ontological model for data synchronization.

Index Terms: Smart Spaces, Smart-M3, Ontology, Smart Conference, Blogging, Intelligent Systems.

I. Introduction

Smart spaces provide an abstract infrastructure for a heterogeneous multi-device system to use a shared view of dynamic resources and context-aware services [1]. This abstraction defines a new paradigm of constructing smart applications for ubiquitous computing, i.e., when computers seamlessly integrate into human lives and applications provide right services anywhere and anytime [2]. A multitude of smart systems has been developed during the last decade. Examples include smart home [3], gaming–wellness–music mashup [4], smart meetings and conferences [5]–[7], and social networks [8], [9].

Smart-M3 [10] is an open-source information sharing platform that provides smart space infrastructures for applications. A Smart-M3 application consists of distributed agents (knowledge processors, KP) running on various computers, ranging from mobile and embedded devices up to dedicated servers and high-performance clusters. This paper considers the problem of integrating several Smart-M3 applications when they enhance own functions based on composition of their smart spaces. In particular, we focus on two independent systems: smart conference system (SCS) [7] and smart blogging system (SmartScribo) [9], [11].

SCS intelligently assists complicated conference processes, automating the hazardous work of conference organizers. It maintains online the visual content available for conference session participants: a current presentation slide from the conference projector and up-to-date session program from the conference whiteboard. SCS allows a presenter to change the slides directly from her/his mobile device. For a session chair, SCS supports automatic time management functions and some other useful online control.

SmartScribo provides advanced access to the blogosphere through its shared presentation as a smart space. SmartScribo users participate in multiple blogs at many blog services using their (mobile) devices as clients. Blog and some other knowledge of user's current interest is kept in the smart space, which is shared with other user's devices and applications as well as with other users. In parallel, other bloggers may use traditional clients (e.g., browser-based), accessing the blogosphere directly. Complex processing, like searching context-relevant blogs or blogger rankings, is delegated to dedicated machines.

Our reference use case is an extension of the SCS functionality with SmartScribo blogging features. It supports online discussions of conference events in the before/during/after mode. Participants typically use mobile devices for clients accessing knowledge in the smart spaces. The conference smart space keeps all knowledge related to the conference, including its program. The participants would like to discuss the program, a talk, a slide of a talk, etc. Authors would like to provide answers and further details based on feedback from other participants. For this feature we enhance SCS such that there is a post in the conference blog-service for each talk, forming the conference blog. A post starts a discussion (questions, opinions, answers from the participants including the authors). In the enhancement, SmartScribo shares blog content from the blog-service into the blogosphere smart space. All blog discussions become accessible to the conference participants via their SmartScribo clients. The key problem is coordination between the SCS smart space, the blog service, and the blogosphere smart space.

The main contribution of the paper is the following.

1) Distributed multi-device multi-agent multi-space architecture for integrating SmartScribo blogging features into SCS. It supports online discussions between the conference participants, even if the conference program is changing dynamically. Our solution employs one of SmartScribo KPs to be a mediator between the smart spaces.
2) Ontological model. First, it defines the structure of content from the two origin smart spaces. Second, it includes ontology to structure data flows between the spaces.
3) Integration scheme. It defines the generic algorithmic logic of the mediator KP for information sharing between the smart spaces.

The rest of the paper is organized as follows. Section II overviews existing systems that automate the conference organization processes online. Section III introduces our architecture for extending SCS with blogging features via integrating SmartScribo into SCS. Section IV describes our ontological model of the smart spaces used in this integration. Based on the architecture and the model we propose our integration scheme in Section V. Finally, Section VI summarizes the paper.

II. Related Work

Information visualization system on distributed displays[1] provides a possibility to control the displaying information during meetings, conferences and workshops. This system allows creating different scripts of meetings, managing a lot of audiovisual system modes in real time and by a pre-arranged scenario, representing numerical data in the business graphic forms. Information from any source in the system (standard audio-visual equipment, specialized software, etc.) can be displayed on any screens. System's software tools can be integrated into an audio-visual complex, providing new opportunities for automated control of its operation modes, conditions and functional interaction of all system hardware and applicable software.

[1]http://www.polymedia.ru/ru/isl/26/

Also, this system has some disadvantages. It is a stationary system, so meetings can be conducted only in a pre-designed room. It uses a large amount of expensive audio and video equipment, and it is difficult for ordinary user to remember and understand all system functions and features because of many complex services.

Multimedia teleconferencing system[2] provides a comprehensive solution for organizing audio, video and web conferencing, collaborative editing of files, drawing different kinds of schemes, etc. It has built-in abilities to synchronize recorded audio conferences and web-conference, and participants can listen discussions and see what data (charts, tables, and slides) are displayed in the session room. It is possible to integrate other applications, such as Microsoft Outlook and Lotus Notes. It has a web interface and support for IP-phones. One of its disadvantages is its strong centralization. All conference materials are stored on the event organizer's devices, hence changes cannot be introduced quickly.

Audio conference system[3] provides a flexible system for audio conferences with up to and sixty simultaneous participants. This system can only work with digital PBX ports having a digital channel ISDN PRI. Software package "Congress" allows managing the conference from a remote workstation (up to 10 pre-arranged conferences with no more than 60 simultaneous participants per one), handling the conference minutes and prepare reports. The system automates collecting the conference participants at the appointed time, etc. The disadvantage is a fixed set of equipment and the inability to connect new devices to the system without complex pre-setting.

Multimedia conference call system[4] focuses on automatic notifications and meeting planning. At the end, the meeting notes/minutes and resolutions are sent to interested parties. The main disadvantage is that the meeting is automated only for its secretary. The meeting notes include: agenda, questions, and reports. The advantage is that meeting minutes are generated automatically.

AM-conference management system[5] is used for support various meetings carried out in factories, offices or workgroups. The main advantage is the ability of a single mode and networking mode. The network mode allows using the system for large structures. The disadvantage is that it is too large and complicated functionality, which requires trainings for users. Also the system requires having a server with high computing capabilities and associated software.

The system of virtual meetings[6] is a tool for organizing different kinds of virtual meetings of geographically remote employees. The main advantage of this system is possibility to keep user information and documents in distributed storage. This makes this system more reliable and stable for network failures. Ontologies of system services provide interoperability for software tools and subsystems. This issue makes events more flexible and with low participation from the human side.

III. Architecture and Scenarios

A smart space can be thought as consisting of devices, services, and knowledge. In Smart-M3, services are implemented as KPs running on the devices. Knowledge is kept in a single RDF triple storage and is accessible via a semantic information broker (SIB). Those KPs that

[2]http://www.ot.ru/facilities_mss.html

[3]http://www.t-service.ru/telephony/ats/conf/conf_k.html

[4]http://www.ot.ru/facilities_mss.html

[5]http://www.auto-management.ru/sector/management/management_21.html

[6]http://www.anbr.ru/page.php?name=sivis&lang=1

run on end-user devices are capable of interacting with people (users). The orchestrated use of integrated physical and computing environment is performed via knowledge sharing in the common smart space.

In this section we describe two Smart-M3 applications of our reference use case: the Smart Conference System (SCS) [6], [7] and smart blogging (SmartScribo) [9], [11]. We introduce architecture of their integration, enhancing SCS with blogging features of SmartScribo. Figure 1 states the architectural vision of both applications and their integration. Our focus is on participating devices and KPs; we shall describe the knowledge structure later in Section IV.

A. Smart Conference

The system uses the Conference Smart Space for keeping its information [6], [7]. The SC smart space contains information about the conference participants, their profiles, information about their presentations, conference schedule, and other technical information.

Smart Conference system consists of the following KPs.

User KP 1) displays information about the conference schedule on the user mobile device, 2) allows to control the user presentation, 3) allows viewing information about another conference participants and their presentations, 4) allows viewing and modifying user profile. Each conference participant needs to install the KP on his/her mobile device or notebook.

Whiteboard KP 1) displays conference schedule on a conference screen, 2) sends notifications to the presenter when the time of his/her presentation is finishing, 3) cancels presentation if the presenter is absent, 3) allows the conference chairman to change the order of participants.

Projector KP displays the current presentation on the conference screen. When the Whiteboard KP changes the current presentation, the Projector KP displays it on the projector.

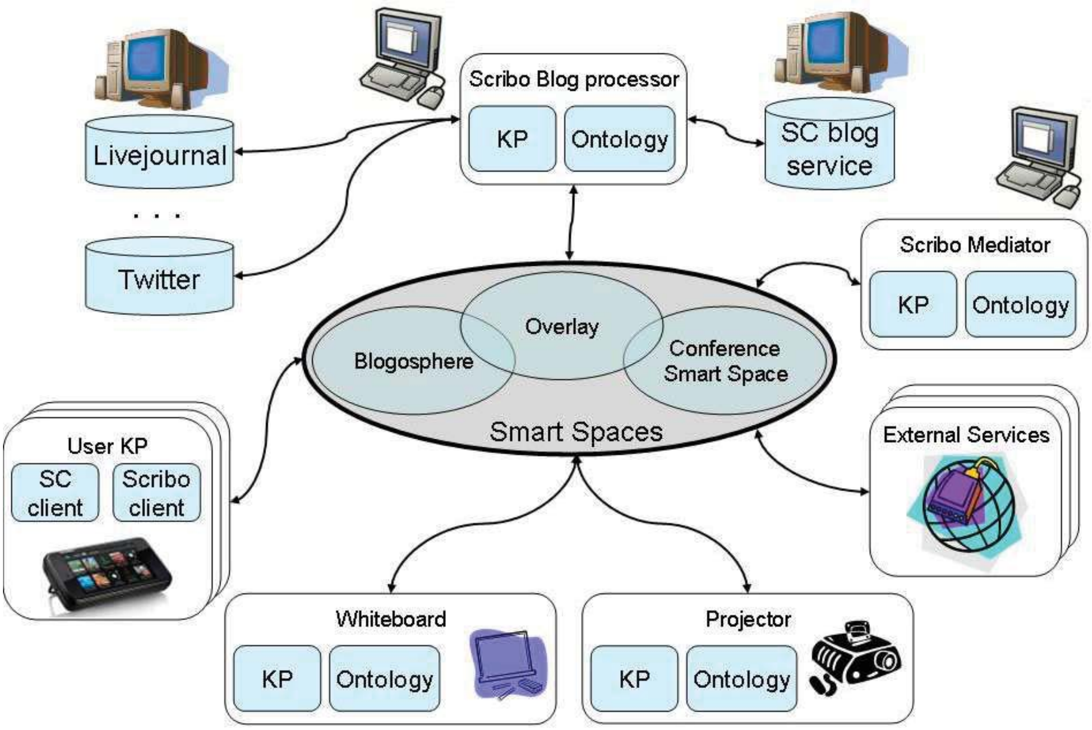

Fig. 1. Distributed multi-device multi-agent architecture of SCS and SmartScribo

It also changes the current slide of the presentation when the User KP changes it in the Conference Smart Space.

External services can be connected to the Smart Conference system if they have appropriate KPs. For example, a translation service can be connected to translate information for users in their native languages.

B. SmartScribo

SmartScribo works in the Blogosphere smart space, which keeps blog discussions interested for the user at the moment. The smart space reflects only a part of information from the Blogosphere; the primary storage is on the blog-service side (LiveJournal, Twitter, etc.). SmartScribo consists of the following KPs.

Blog Processor (BP) is a KP that 1) retrieves blog data available in the blog-service and publishes them into the blogosphere smart space and 2) discovers new data in the space (from clients) and sends them to the blog-service. Each blog service requires a dedicated BP. Since there can be many users, BP should run on a computer with enough capacity.

Blog Client (BC) is a user KP that accesses blog data available in the blogosphere smart space and visualizes them as a tree-like structure. When the user writes new messages the KP publishes them in the smart space. Typically BP runs on user's own mobile computer, e.g., smartphone or Nokia N900.

Blog Mediator (BM) is a KP that produces "meta-information" for blogging. For instance, BM can implement a search or recommendation system, finding blog discussions of the most current interest of the user. The class of corresponding devices depends on the computational complexity of the BM function; a normal PC seems suitable in many cases.

Note that the content of a blog-service may be changed by applications that are not part of SmartScribo, e.g., some bloggers access the service directly with Internet browser.

C. Integration

For extending SCS with blogging SCS deploys its own blog-service, e.g., using the LiveJournal engine. The SC blog-service is a primary storage for all blog discussions related to each conference.

For each conference SCS builds a separate blog in advance. The blog consists of posts, one per a talk/paper. In addition, some other posts can be made for other discussion threads, e.g., for a discussion on the conference organization. In summary, it forms the initial phase when SCS defines the structure for discussions.

After that the conference moves to the regular online phase. Participants start SCS and SmartScribo clients on their user devices. The use of SCS is as before. When the user needs to discuss a conference element she switches to her SmartScribo client.

To connect the conference program and its discussion threads we assign one SmartScribo BP to be a mediator. This SC-BP keeps tracks of the SC program in the SC smart space. Whenever a new element is appearing or an update is happening SC-BP reflects it in the blog-service. This connection is unidirectional; SC-BP reads data from SCS and transforms the program into the blog structure. To diminish improper intervention from external users to the conference blog-service, any post (or comment) that is a part of the structure is published on behalf of SCS (an abstract user with admin rights). All other users, even they access the service directly, may not modify the structure.

The connection between the SC blog-service and the smart space is implemented as it is in SmartScribo. There is a BP that processes corresponding data flows. This functionality can be

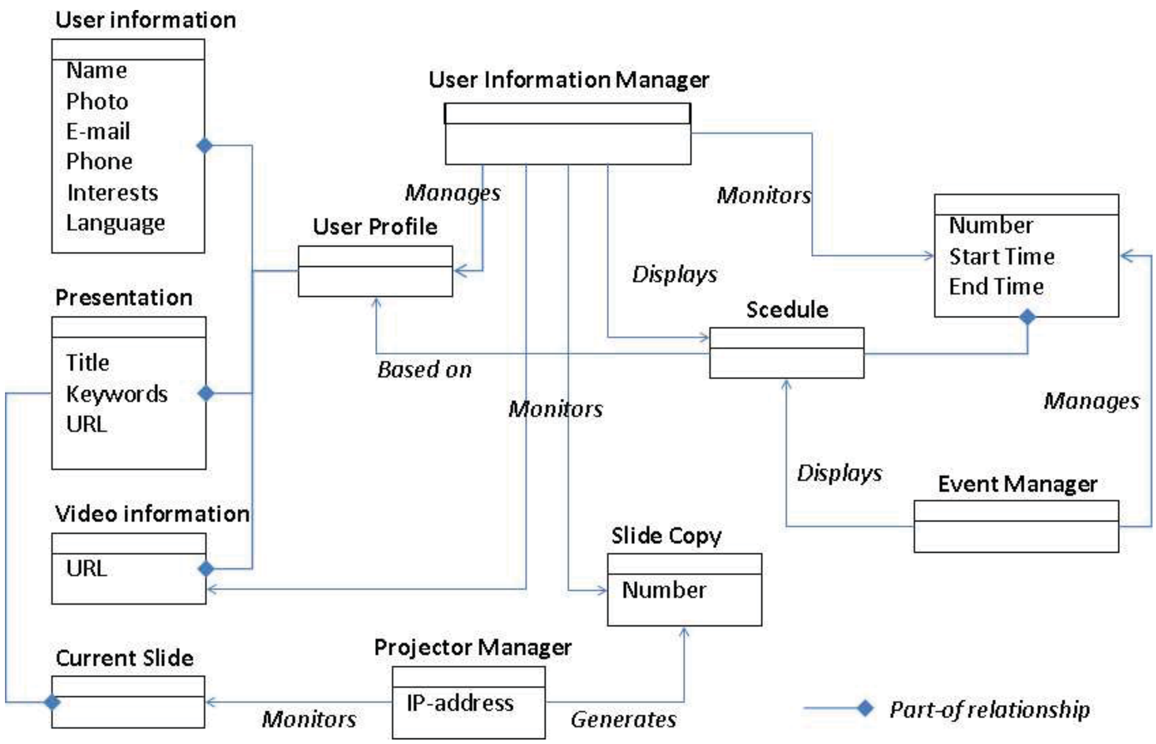

Fig. 2. Ontology of knowledge in the conference smart space

integrated into SC-BP, making the latter the center in the Y-network of SCS, SC blog-service, and SmartScribo. In contrast to the "SCS $\rightarrow$ blog-service" the connection "blog-service $\leftrightarrow$ SmartScribo" is bidirectional. Our current design assumes a common blog account for all participants, in addition to the SC admin account.

The architectural and algorithmic details of SC-BP will be considered in Section V. Note that our solution allows two separate KPs to implement the above two connections. We leave this topic for our future analysis.

IV. Ontology

This section starts with description of the base ontologies inherited from our two origin applications. Although all knowledge is stored in a common RDF triple storage, SCS and SmartScribo use own independent ontologies. Moreover, the knowledge sets are separated on the application level, i.e., the sets of RDF triples are disjoint. Cooperation needs sharing certain knowledge, and we suggest overlay ontology on top of the base ones to server as a bridge for knowledge exchange. Its core is notification ontology for synchronization of changes happening on both sides.

A. Conference smart space

Conference smart space (Figure 2) consists of three main parts: user information manager, event manager, and projector manager. User information manager deals with user profile, displays presenter's video and slide thumbnail, monitors time intervals of the conference, and displays conference schedule. User Profile consists of user information and presentation related information. User information includes: name, photo, e-mail, contact phone, topics of interests, and preferred language.

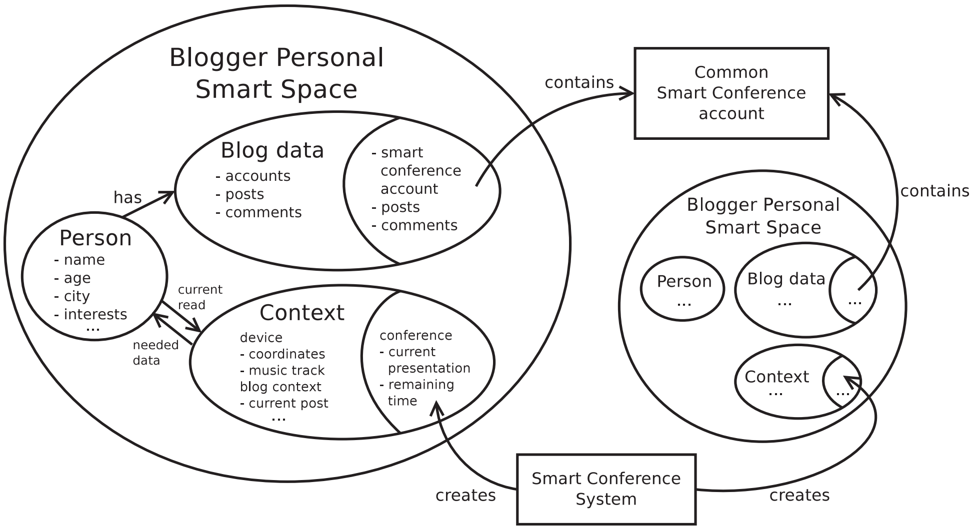

Fig. 3. Structure of blogger's personal smart space in SmartScribo

Presentation related information includes: name of presentation, keywords, and its URI, which will be used for transferring the presentation into the projector at the scheduled time. Also, the presentation information has a section related to video, which can be displayed during the presentation from the projector and in mobile devices (via user information managers). Projector manager generates thumbnail of current slide, which is displayed on mobile devices. Event manager displays the conference schedule and manages time interval. Conference schedule is created based on user profile.

B. Personal smart spaces for blogging

In general case, the Blogosphere smart space is composed of many personal smart spaces of bloggers [11]. Since our current design uses a common blog account for every blogger we deal with an easier case. At the highest level the knowledge in every personal space is divided into the three related subspaces: user profile, context, and blog data (Figure 3). More details can be found in [11]. In this paper we consider only particularities needed in our reference use case of SC blogging.

Profile describes permanent or long-term data about user (name, age, e-mail, interests, etc.). Such data are not blog-specific and can be described with FOAF standard ontology and its extensions. Since we use the only account for every SC participant the profile records contain aggregate information for an abstract participant.

Blog data include service information and actually blog discussions. In the general case, service information is accounts on all blog-services. The SC case needs the only account. Profile is linked with blog data using the FOAF concept “person have account”.

The SC blog consists of many discussions, in accordance with the structure defined by SCS. As in the general case, each discussion is a tree rooted with a post. SC participants add comments to the post or to already existing comments. Each author signs her message inside the body text or leaves the message anonymous.

Context represents current and mutable characteristics related to a person (current location, weather, music track, person's mood, etc.). In the SC case, the context corresponds to an abstract participant, hence describing certain context of the whole conference.

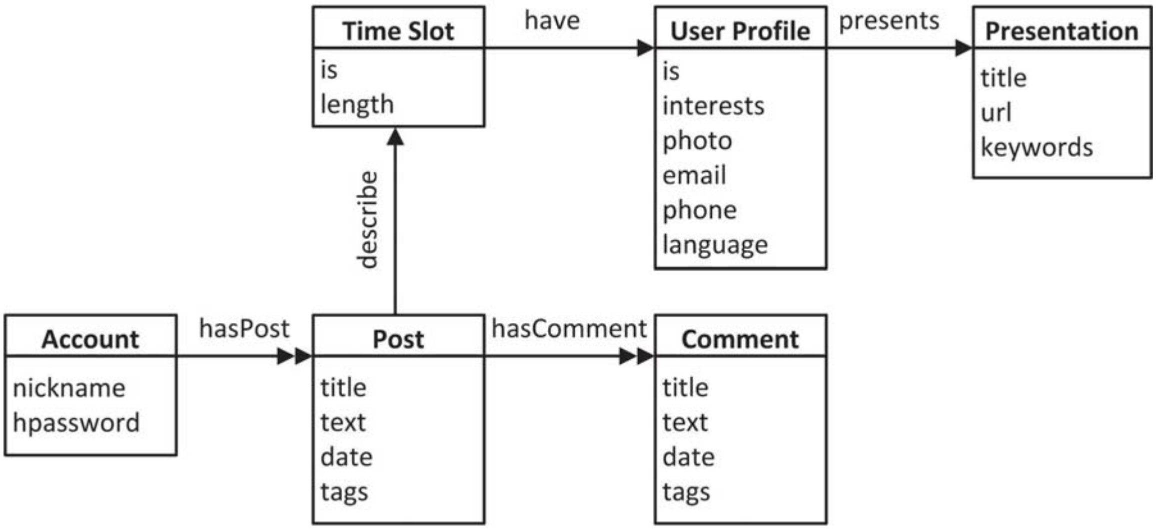

Fig. 4. Overlay ontology used by a blog processor

C. *Notification model overlay ontology*

Notifications initiate KP to execute some actions or to inform KP about execution result. A notification is implemented as a triple or set of them, and KPs may subscribe to the triples. Whenever such a triple is changed the KP is notified. In the SC case, SC-BP operates with two sets of notifications: the first one is for requests from SCS and the second one is for updates from either SmartScribo or the SC blog-service.

The following scheme supports proactive service discovery based on the notification model. When one KP (initiator) needs to service from another KP (responder) the former updates the notification triples. Due to subscription mechanism the initiator KP activates the service and then publishes the result in the smart space. Since the initiator KP is subscribed to the triples that contain the expected result, this KP queries the data whenever they are published by the servant KP. Both KP know the set of possible services, expected results, and corresponding triples in advance.

Blogging notifications are divided into the following groups: accounts, posts, and comments. An account notification requests SC-BP to perform some operations using account login and password, e.g., BP logins into the blog-service on behalf of the blog client. It can be represented as a triple “ServiceType—Operation—AccountInfo”, where the object defines a type of the blog service (Live Journal, Twitter, WordPress, etc.), the predicate is the operation to perform (login, refresh, etc.), and subject is an individual that contains account information. Posts and comment notifications are similar with operations send, receive, update, and delete.

The overlay ontology is composed from triples of two source ontologies as shown in Figure 4. There is one-to-one relation between Post class from SmartScribo and TimeSlot class from Smart Conference. We add the property ’describe’ to link these classes. The meaning of this link is that there exists the only blog post for each time slot. This post contains information about a report that has been planned for that time.

V. KP-BASED INTEGRATION

Figure 5 shows the basic scheme that a SC blog processor follows to coordinate in the integration. The blog processor performs the following actions:

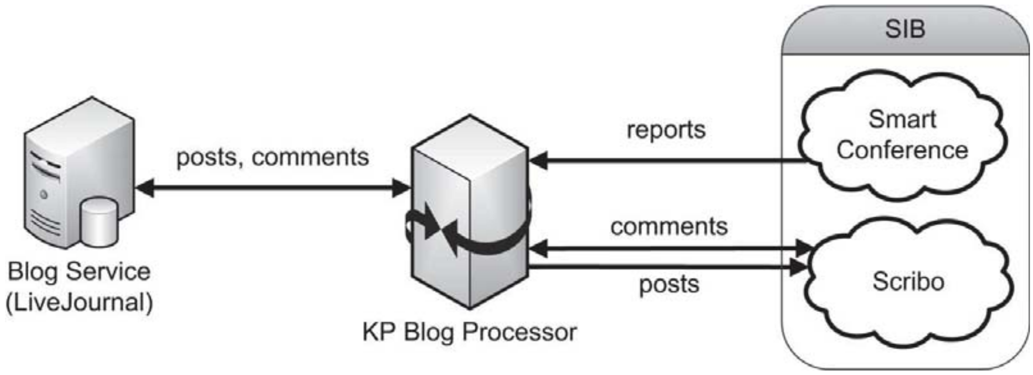

Fig. 5. Data flows of a Smart Conference Blog Processor

- loading information about the conference schedule from the conference smart space;
- adding comments and new posts (report topics) into the blogosphere smart space;
- interacting with the conference blog service (LiveJournal) to keep posts and comment up to date;
- emitting (add/delete) notification records.

Blog processor is implemented in Qt (C++ interface) and contains three main components: LiveJournalHandler, SmartSpaceHandler and Synchronizer. LiveJournalHandler interacts with LiveJournal service by HTTP and flat protocols. It implements authorization and receiving/sending posts and comments. SmartSpaceHandler interacts with SIB to access the smart spaces of Smart Conference and blogosphere. Synchronizer is responsible for synchronization between the blog service and SIB.

Figures 6 and 7 provides our behavioral model of synchronization the data flows.

Post synchronization is initiated by receiving refreshPosts notification. Accordingly, the blog processor loads reports (time slot, author, topic and so on) from the SIB and converts the information into internal structures (in blog service format). The next step is coherence checking and adding absent topics. Synchronization of the blog service and SmartScribo is similar. When the operations have been completed, the blog processor creates refreshPosts notification.

Similarly, refreshComments notification is used for comment synchronization. In this process, the blog processor loads comments from the blogosphere smart space and from the Blog Service. Then BP adds absent pieces of data. Finally, BP writes refreshComments notification.

VI. Conclusion

In this paper we proposed a solution consisting of three key elements: 1) architecture of the integration, 2) ontological model, 3) integration scheme. In total it defines a method that allows constructing an integration of independent smart applications. Based on the reference use case elaborated in this paper, the method is easily generalized to a class of integrations with many smart applications. Information sharing between their smart spaces is coordinated via a dedicated KP that applies an integration ontology on top of the ontologies from participating smart spaces.

Acknowledgment

Authors would like to thank Finnish-Russian University Cooperation in Telecommunications (FRUCT) program for the provided support and R&D infrastructure. We would also like to

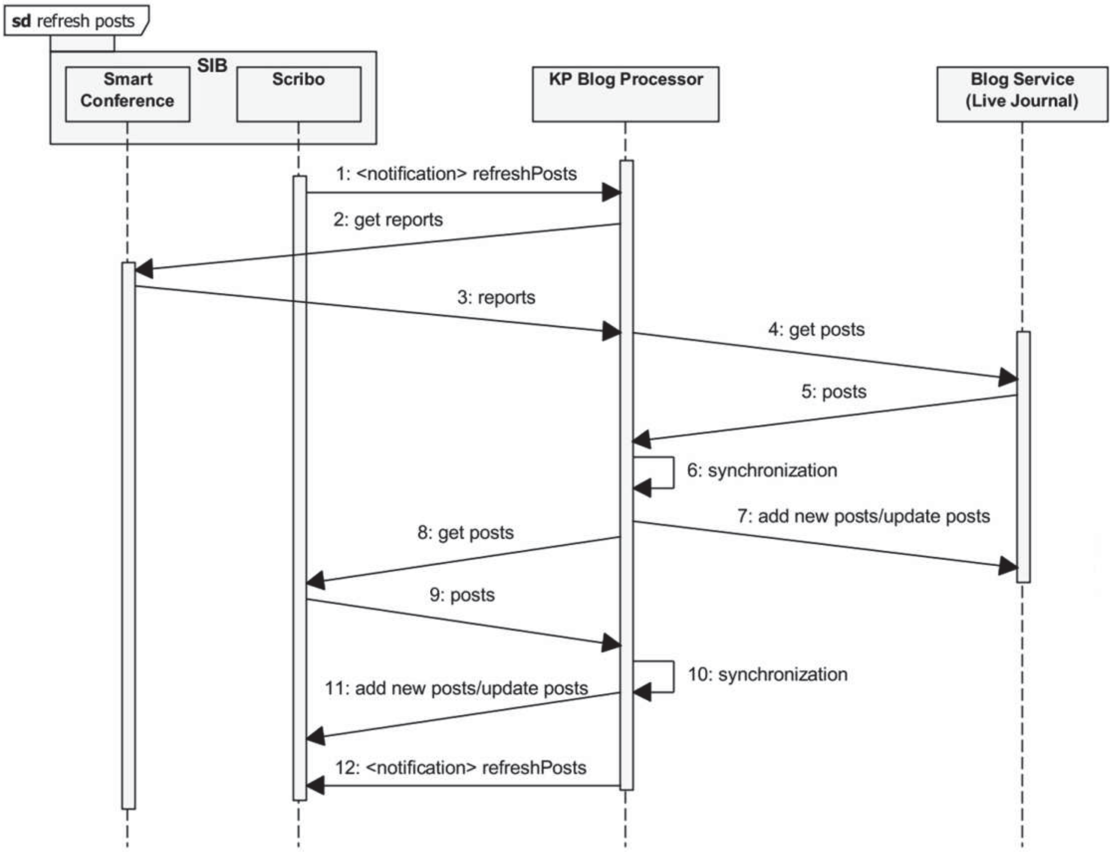

Fig. 6. Synchronizing posts for the Smart Conference blog

thank Sergey Balandin and Jukka Honkola for providing feedback and guidance.

The paper is a part of the research carried out within the project funded by grants 10-07-00368-a of the Russian Foundation for Basic Research, project # 213 of the research program "Intelligent information technologies, mathematical modeling, system analysis and automation" of the Russian Academy of Sciences.

REFERENCES

[1] I. Oliver, "Information spaces as a basis for personalising the semantic web," in *Proc. 11th Int'l Conf. Enterprise Information Systems (ICEIS 2009)*, vol. SAIC, May 2009, pp. 179–184.

[2] M. Weiser, "The computer for the twenty-first century," *Scientific American*, vol. 265, no. 3, pp. 94–104, 1991.

[3] K. Främling, I. Oliver, J. Honkola, and J. Nyman, "Smart spaces for ubiquitously smart buildings," in *Proc. 3rd Int'l Conf. Mobile Ubiquitous Computing, Systems, Services and Technologies (UBICOMM 2009)*. IEEE Computer Society, 2009, pp. 295–300.

[4] J. Honkola, H. Laine, R. Brown, and I. Oliver, "Cross-domain interoperability: A case study," in *Proc. 9th Int'l Conf. Smart Spaces and Next Generation Wired/Wireless Networking (NEW2AN'09) and 2nd Conf. Smart Spaces (ruSMART'09)*, ser. Lecture Notes in Computer Science, vol. 5764. Springer-Verlag, 2009, pp. 22–31.

[5] I. Oliver, E. Nuutila, and S. Törmä, "Context gathering in meetings: Business processes meet the agents and the semantic web," in *The 4th Int'l Workshop on Technologies for Context-Aware Business Process Management (TCoB 2009) within Proc. Joint Workshop on Advanced Technologies and Techniques for Enterprise Information Systems*. INSTICC Press, May 2009.

[6] A. Smirnov, A. Kashnevik, N. Shilov, I. Oliver, S. Balandin, and S. Boldyrev, "Anonymous agent coordination in smart spaces: State-of-the-art," in *Proc. 9th Int'l Conf. Smart Spaces and Next Generation Wired/Wireless Networking (NEW2AN'09) and 2nd Conf. Smart Spaces (ruSMART'09)*, ser. Lecture Notes in Computer Science, vol. 5764. Springer-Verlag, 2009, pp. 42–51.

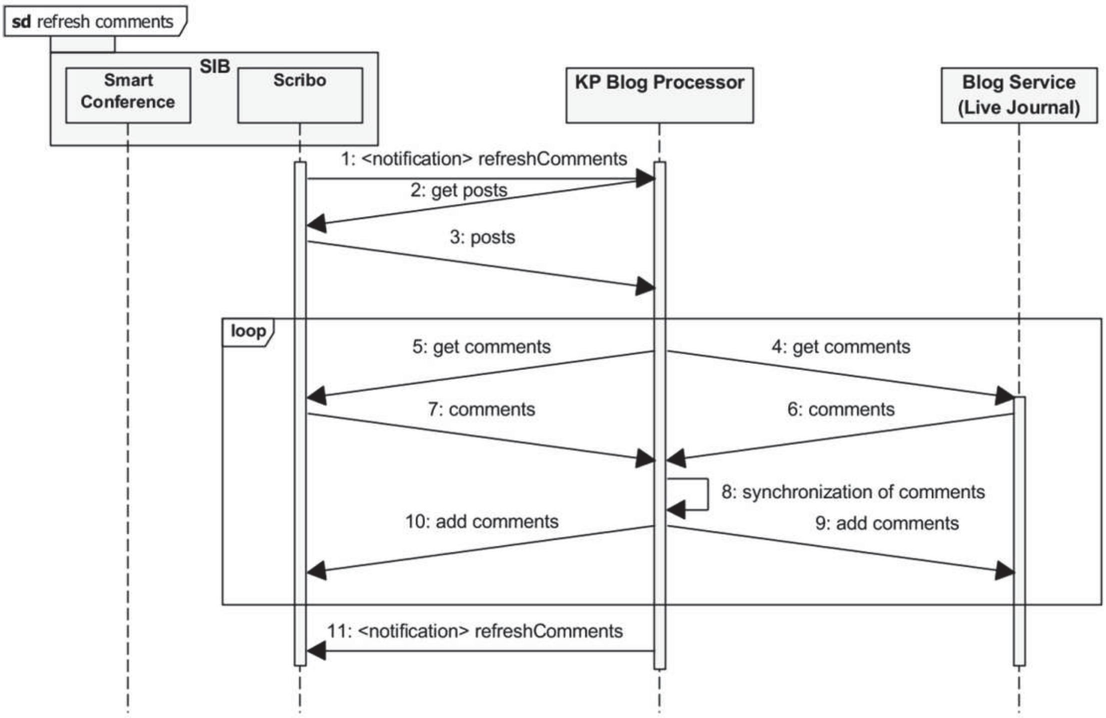

Fig. 7. Synchronizing comments for the Smart Conference blog

[7] A. Kashevnik, Y. Valchenko, M. Sitaev, and N. Shilov, "Smart conference system," *SPIIRAS Proceedings*, vol. 14, pp. 228–245, 2010, in Russian.

[8] S. Balandin, I. Oliver, and S. Boldyrev, "Distributed architecture of a professional social network on top of M3 smart space solution made in PCs and mobile devices friendly manner," in *Proc. 3rd Int'l Conf. Mobile Ubiquitous Computing, Systems, Services and Technologies (UBICOMM 2009)*. IEEE Computer Society, 2009, pp. 318–323.

[9] D. Zaiceva, I. Galov, A. Sannikov, and D. Korzun, "Mobile multi-blogging in Smart-M3: Architecture and scenarios," in *Proc. 8th Conf. of Open Innovations Framework Program FRUCT*, Nov. 2010, pp. 225–233.

[10] J. Honkola, H. Laine, R. Brown, and O. Tyrkkö, "Smart-M3 information sharing platform," in *The 1st Int'l Workshop on Semantic Interoperability for Smart Spaces (SISS 2010) in conjunction with IEEE ISCC 2010*, Jun. 2010.

[11] D. Zaiceva, I. Galov, and D. Korzun, "A blogging application for smart spaces," in *Proc. 9th Conf. of Open Innovations Framework Program FRUCT*, Apr. 2011.