Indirect Interaction of Agents in a Smart Space: Operation Efficiency and Fault Tolerance Support

Ivan V. Galov
Petrozavodsk State University (PetrSU)
Petrozavodsk, Russia
galov@cs.karelia.ru

***Abstract*—The Smart-M3 platform allows to create distributed applications due to interaction of agents via shared information space. The important problem is dependability of agent interactions which can be affected by various failures in ubiquitous computing environment. To support agent interactions we suggest improved agent operation processing scheme, notification model for effective agents coordination and fault tolerance mechanisms.**

I. Introduction

Various ubiquitous computing environments become more popular [1]. Smart spaces is a paradigm related to creating such environments. Typical smart space system is targeted for use in a physical spatial-constrained space involving multitude of various devices employing Internet of Things (IoT) technologies. Smart-M3 [2] is an open source prototype for developing and deploying smart spaces. It allows to create distributed multi-agent applications based on publish/subscribe and blackboard models. Applications are constructed due to indirect interaction of agents via shared information space which is maintained by semantic information broker (SIB). Smart-M3 agents are called knowledge processors (KPs).

An important issue in smart spaces is a way of constructing, deploying and running systems in IoT environments. It brings different challenges such as failures in interaction between KPs during processing KP requests or failures in smart space infrastructure [3], [4] because of volatile nature of IoT environment devices and communications. To overcome such problems we consider approaches of agent interactions support on SIB side as well as on KP side.

The rest of the abstract is organized as follows. Section II describes problem of indirect interactions support and related work. Section III suggests operation processing scheme in SIB to increase operation efficiency. Section IV considers notification model to coordinate KP interactions. Section V is related to fault tolerance mechanisms to improve Smart-M3 applications dependability. Section VI concludes the abstract.

II. Agent Interactions Support

Agent interactions are performed due to operation requests processing in SIB. The way the operation processing is performed and supported in SIB affects on operation efficiency during KP interaction. If some operations are not properly processed it results in interaction problems and errors in applications. Thus it is necessary to have reliable and effective operation processing scheme in SIB.

Subscription is the most important operation which implements indirect interactions. As subscription is a resource-consuming operation it is essential to arrange KP interactions in effective way with minimized number of subscriptions. With minimal subscriptions number SIB load decreases and applications become more dependable as less number of subscriptions decreases the chance of subscription connection failure.

Besides dependability of interactions also depends on smart space infrastructure. Failures in infrastructure elements can lead to disruption in interactions and violate the right application operation. Utilizing mechanisms which support infrastructure elements and restore their operability in case of failures allows to increase applications dependability.

Currently there are works related to constructing and supporting of smart space applications and services [5], [6], [7], [8]. But they do not provide definite methods how to support dependable interactions between KPs and deploy dependable smart space infrastructure in IoT environment. We suggest three approaches to support smart space deploying and improve dependability of KP interactions: a) by improving SIB operations processing scheme, b) by introducing notification model to assist programmers in developing effective and dependable agent interactions, and c) by enforcing Smart-M3 software infrastructure with fault tolerance mechanisms.

III. SIB Operations Processing

Current SIB implementations have several drawbacks affecting their dependability [9] thus we suggested plug-in based SIB architecture used in our CuteSIB implementation. Improved architecture allows to configure operation processing flow and extends standard SSAP operations with SIB rules. CuteSIB operation processing scheme is shown in Fig. 1.

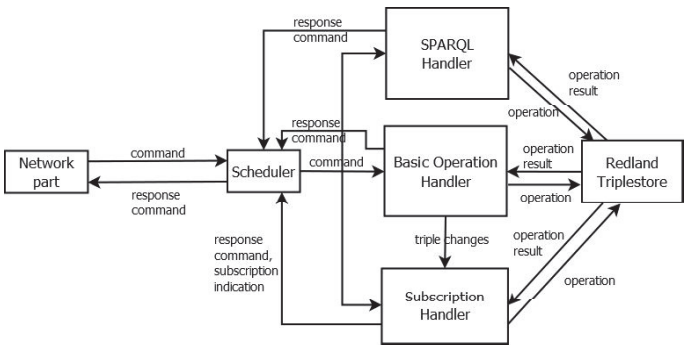

Fig. 1. Operation processing in CuteSIB

ISBN 978-952-68397-0-7 (paperback), ISBN 978-952-68397-1-4 (PDF)

All requests from KPs are represented as commands. Commands are created in network part of SIB and then are sent to scheduler module sequentially. Scheduler identifies command type (type of request) and then dispatches command to corresponding command handler. There are several command handlers: for basic operations (insert, remove, update, query), SPARQL requests and subscription operations. Using different command handlers allows to process different commands simultaneously. Besides improved architecture allows to activate/deactivate different command handlers thus it is possible to configure operation processing in SIB according to environmental needs. Each command handler gains access to triplestore (which is maintained by Redland library), performs corresponding operation and send response command to network part via scheduler.

Subscription is a special operation as subscription handler should interact with basic operations handler. When basic operation (which changes some data in triplestore) is performed, information about changes is sent to subscription handler asynchronously. Subscription handler checks all subscriptions according to changes and sends subscription indication response to network part if it is required.

CuteSIB operations implementation is based on RedSIB [10]. Unlike RedSIB, CuteSIB does not use D-BUS to communicate between network part and triplestore. We measured percent of failed subscription indications (indications which were not received) emulating 100 KPs working at a time. In the experiment parallel working KP processes sent random triples they are subscribed to and counted number of received subscription indications. Then the average subscription loss percent is evaluated. The experiment showed that CuteSIB subscription loss is about 6% and RedSIB subscription loss is 45%. Thus D-BUS elimination leads to decrease of subscription indication loss. on 39%. We also measured operation processing time (time since operation request is received and operation response is ready) on CuteSIB and RedSIB. Operation processing time on both SIBs remained approximately the same as they use the same operation implementations. Thus in comparison with RedSIB operation efficiency in CuteSIB improved due to decreasing of subscription indication loss with preserved operation processing time.

IV. Notification Model

Apart from reliable SIB operations processing it is essential to have an approach for designing an effective interactions between agents. Notification model [11] allows to initiate interaction in situation when changing some particular ontology data is not sufficient for inducing interaction on subscription. Notification is represented as a set of special triples which are sent from one KP to another to initiate interaction (Fig. 2). Notification can transfer to another KP either text message or some part of application data (individuals from ontology).

Usage of notification model by programmers consist in several steps: a) defining number of interactions between every two KPs, b) describing notification properties and its parameters in notification ontology for every interaction, c) interactions programming according to notification ontology. This way programmers can design and describe effective interactions between KPs.

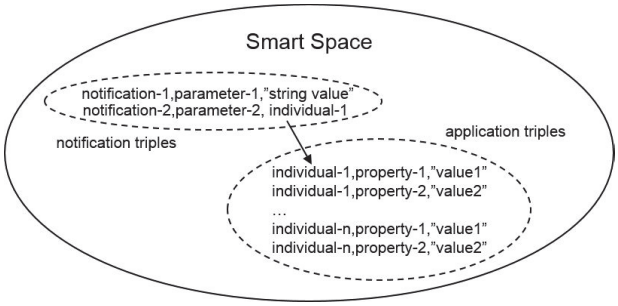

Fig. 2. Relation between notification and application triples

Consider notification model usage in SmartRoom system [12]. There are three base services: conference-service which controls conference runtime, agenda-service which displays conference agenda and presentation-service which displays speaker presentations. When chairman starts the conference — "startConference" notification is sent to conference service and it starts the conference. During the conference start conference-service sends "updateAgenda" notification to agenda-service to update current speaker and "startPresentation" notification to presentation-service to start presentation slides show. At the end of presentation speaker or chairman sends "endPresentation" notification to the conference-service, the next presentation is started and the process repeats. In this case notification model is used to initiate execution of particular actions by service.

Notification model can be used for defining interactions between KPs in Smart-M3 application in unified and effective way. Notification allows to initiate interaction and pass information to another KP due to possibility to transfer either text data or link to smart space data with notification. Such approach allows to reconsider and minimize number of subscriptions needed for KPs in Smart-M3 application.

V. Fault Tolerance Mechanisms

Dependability of KP requests processing can be supported by special fault tolerance mechanisms implemented on SIB or KP side or on software infrastructure level. Such mechanisms extend software infrastructure of Smart-M3 application with restart/reconnect and subscription control functionality and introduce special content-service for persistent storage of volumetric data [13]. Fig. 3 shows enhanced infrastructure for a Smart-M3 application. Every KP implements reconnect function and in case of network problems can reestablish network connection with SIB. If there is a problem with KP application then operating system software can restore KP functioning with restarting this KP. Subscription control mechanism performs regular checks for subscribed data and in case of failure can restore KP's subscriptions. Many Smart-M3 applications can work with volumetric data. Content-service represents a file store which allows other KPs to store volumetric data and share links to this data in a smart space.

Fault tolerance mechanisms considered above can be applied in any Smart-M3 application to increase its dependability. Due to network connection and subscription recovering KPs can interact in ubiquitous computing environments and continue interactions with each other in case of failures. In case

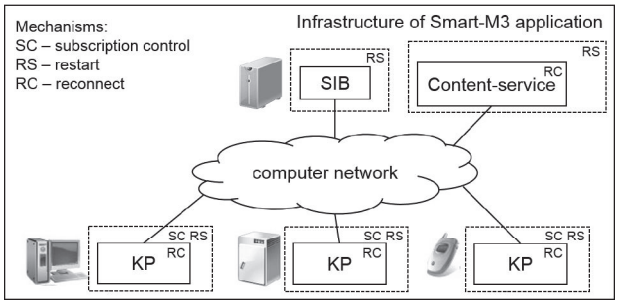

Fig. 3. Enhanced infrastructure for a Smart-M3 application

of failure of full infrastructure content-service allows to keep volumetric data safe.

VI. Conclusion

In this abstract we suggested three approaches to support interactions between KPs: operation processing scheme, notification model and fault tolerance mechanisms. They allow to define effective interactions between KPs, process operations in more dependable and effective way and recover KP interactions on failures. It increases dependability of the whole Smart-M3 application and software infrastructure.

Acknowledgment

This research is financially supported by the Ministry of Education and Science of Russia within project # 1481 of the basic part of state research assignment for 2014–2016 and project # 14.574.21.0060 (RFMEFI57414X0060) of Federal Target Program "Research and development on priority directions of scientific-technological complex of Russia for 2014–2020". The reported study was supported by the Russian Foundation for Basic Research, research project # 14-07-00252. We would like to thank Dmitry Korzun, Alexey Kashevnik, and Sergey Balandin for their feedback and expertise.

References

[1] J. Augusto, V. Callaghan, D. Cook, A. Kameas, and I. Satoh, "Intelligent environments: a manifesto," *Human-centric Computing and Information Sciences*, vol. 3, no. 1, 2013. [Online]. Available: http://dx.doi.org/10.1186/2192-1962-3-12

[2] J. Honkola, H. Laine, R. Brown, and O. Tyrkkö, "Smart-M3 information sharing platform," in *Proc. IEEE Symp. Computers and Communications (ISCC'10)*. IEEE Computer Society, Jun. 2010, pp. 1041–1046.

[3] P. Pande and A. R. Padwalkar, "Internet of things–a future of internet: A survey," *International Journal of Advance Research in Computer Science and Management Studies*, vol. 2, no. 2, pp. 354–361, 2014.

[4] I. Galov and D. Korzun, "The SmartRoom infrastructure: Service runtime reliability," in *Proc. 14th Conf. of Open Innovations Association FRUCT*, S. Balandin and U. Trifonova, Eds. SUAI, Nov. 2013, pp. 188–189.

[5] D. Korzun, "Service formalism and architectural abstractions for smart space applications," in *Proc. 10th Central & Eastern European Software Engineering Conference in Russia (CEE-SECR 2014)*. ACM, Oct. 2014.

[6] S. Balandin and H. Waris, "Key properties in the development of smart spaces," in *Proc. 5th Int'l Conf. Universal Access in Human-Computer Interaction (UAHCI '09). Part II: Intelligent and Ubiquitous Interaction Environments, LNCS 5615*, C. Stephanidis, Ed. Springer-Verlag, 2009, pp. 3–12.

[7] S. Balandin, I. Oliver, S. Boldyrev, A. Smirnov, A. Kashevnik, and N. Shilov, "Anonymous agents coordination in smart spaces," in *Proc. 4th Int'l Conf. Mobile Ubiquitous Computing, Systems, Services and Technologies (UBICOMM 2010)*, Oct. 2010, pp. 242–246.

[8] I. Paramonov, A. Vasilev, D. Laure, and I. Timofeev, "Agent substitution mechanism for dataflow networks: Case study and implementation in Smart-M3," in *Proc. 13th Int'l Conf. Next Generation Wired/Wireless Networking and 6th Conf. on Internet of Things and Smart Spaces (NEW2AN/ruSMART 2013)*, ser. LNCS 8121, S. Balandin, S. Andreev, and Y. Koucheryavy, Eds. Springer-Verlag, Aug. 2013, pp. 60–71.

[9] I. Galov and D. Korzun, "Design of semantic information broker for localized computing environments in the Internet of Things," in *Proc. 17th Conf. of Open Innovations Association FRUCT*. ITMO Univeristy, Apr. 2015, pp. 36–43.

[10] F. Morandi, L. Roffia, A. D'Elia, F. Vergari, and T. S. Cinotti, "RedSib: a Smart-M3 semantic information broker implementation," in *Proc. 12th Conf. of Open Innovations Association FRUCT and Seminar on e-Tourism*, S. Balandin and A. Ovchinnikov, Eds. SUAI, Nov. 2012, pp. 86–98.

[11] I. Galov and D. Korzun, "A notification model for Smart-M3 applications," in *Proc. 14th Int'l Conf. Next Generation Wired/Wireless Networking and 7th Conf. on Internet of Things and Smart Spaces (NEW2AN/ruSMART 2014), LNCS 8638*, S. Balandin, S. Andreev, and Y. Koucheryavy, Eds. Springer-Verlag, Aug. 2014, pp. 121–132.

[12] D. Korzun, I. Galov, A. Kashevnik, and S. Balandin, "Virtual shared workspace for smart spaces and M3-based case study," in *Proc. 15th Conf. of Open Innovations Association FRUCT*, S. Balandin and U. Trifonova, Eds. ITMO Univeristy, Apr. 2014, pp. 60–68.

[13] I. Galov and D. Korzun, "Fault tolerance support of Smart-M3 application on the software infrastructure level," in *Proc. 16th Conf. of Open Innovations Association FRUCT*. ITMO Univeristy, Oct. 2014, pp. 16–23.