Mechanism for Robust Dataflow Operation on Smart Spaces

Ilya Paramonov, Andrey Vasilev
P. G. Demidov Yaroslavl State University
Yaroslavl, Russia
Ilya.Paramonov@fruct.org, vamonster@gmail.com

Sergey Balandin
FRUCT Oy
Helsinki, Finland
sergey.balandin@fruct.org

Abstract

Smart Space applications could use different architectures during their operations. The dataflow network pattern allows to modularize the system and provide seamless support for replaceable components. One of information sources for ubiquitous environments are the sensors, whose readings are processed by a number of computational units. The sensors and processing nodes together form the dataflow network. In this paper we describe the implementation of asynchronous dataflow network on top of RDF store. Possible causes of dataflow disruption are discussed. The mechanism for maintaining network operation when data processing unit looses connection with RDF store is proposed. The corresponding modification of node operation is provided.

Index Terms: RDF store, Dataflow network, Smart M3, Node substitution.

I. INTRODUCTION

The ubiquitous computing is a model for interaction with computers, in which computing elements provide information and services when and where desired [1]. There is a number of approaches for creating ubiquitous applications [2]. One of them is the smart space paradigm [3], which provides infrastructure for development of multi-agent systems having shared view on dynamically changing resources and providing different services. Several applications have been built in accordance with the paradigm including multi-blogging system [4], the unified video monitoring system [5], smart-home environments and so on.

Smart space applications consist of several components that interact with each other. Though being oriented to perform in dynamic environments, such systems are not designed to maintain the operation in the situation when one or several agent loose connection with the other. So, the whole application or big part of it should be restarted in order to resume the operation. In accordance with the recent vision of future ubiquitous environments [6], the number of smart objects and complexity of provided services should increase. So, the existing approach can not be used and it is desirable to develop solution for the stated problem to provide continuous service to the end-user.

The issue could be overcome either by changing the architecture of the application or by extending the functionality of communication platform. The first approach requires each application developer to create own issue prevention strategy and apply it to the operation of all application-critical agents. This approach potentially allows to design and implement system specific solution, but this requires resources, as the complexity of the whole system increases. The second approach for sustaining system operation allows every application designer to rely on general mechanism provided by the platform, therefore reducing the system design time. Also, such mechanism could be combined with the first approach if some special properties are required from the operation sustaining mechanism.

We have decided to provide support for maintaining operation of smart space application on the communication platform level. The main idea of the proposed support system is to add additional agents to the smart space, which would replace the original agents of system during their absence or malfunction. This way other agents, which depend on data coming from failing agent could continue their operation on a satisfactory level. One of the key aspects of agent substitution is the context transfer from the failing agent to the substitute. Supplied with the knowledge of a context of the original agent, the substitute should provide reasonable level of service for dependent agents.

The system is designed to work with the open source solution for development of smart space applications Smart-M3 platform [7]. The core of the platform is the Semantic Information Broker (SIB), which provides access to the shared information. The data in the smart space is stored as the Resource Description Framework (RDF) graph and there are no restrictions on data, which can be put into the storage. The application built on top of the platform consists of several agents, which are named Knowledge Processors (KP).

The rest of the paper is structured as following. In Section II the description of used communication framework is given. Section III gives an operational description of a dataflow network. In Section IV the possible cases of service disruption are discussed. Section V presents a description of node substitution mechanism. Section VI summarizes the paper.

II. Smart-M3 platform

Smart-M3 is an information sharing platform, which provides infrastructure for creating ubiquitous applications in accordance with the smart spaces paradigm. As stated in introduction, the heart of the platform is the SIB, which allows application agents - KPs interact by the means of information sharing. The information in SIB is stored according to RDF model, where the base element is a triple. The triple consists out of a subject, a predicate and an object and represents a statement of relationship between the things described by the subject and the object. The collection of triples forms the RDF graph, where subjects and objects represent KPs and predicates describe relationship between them.

On the low-level a KP manipulates RDF triples and it may insert, delete, update and query data in the SIB. SIB can notify KP on data modification if KP subscribes to the specified data change. On the higher-level a set of ontologies could be used to describe the information, which is used and produced by application KPs. The ontology not only defines the structure of data, but also provides a semantics for it, therefore KPs may not only use predefined data structures, but to adapt to different information sources, which are currently available. Such information could be used by KPs to provide general-purpose reasoning services as described in [8], [9] and [10]. The ontology is usually not presented inside the smart space.

One of the basic scenarios of agent interaction in Smart-M3 platform is the following: one KP places the description of its state into the smart space and keeps it updated, other can act on this information to modify their behavior and adapt to the current situation, if necessary. The behavior of the KP defines the order of the information modification. The more complex information scenarios are possible, the overview of existing application architectures and KP interaction models can be found in [11].

Smart-M3 is the open source platform, so it allows not only to develop applications using it capabilities, but to adapt it for concrete research and development needs. There have been developed several general enhancements, which provide additional services to the platform. The examples of such enhancements include support for different reasoning techniques and access control mechanism. The access control could be implemented by several ways, the approach implemented in [12] allows to set write protection on RDF data subgraph, so only one KP may write to specified triples.

III. Dataflow network implementation on RDF store

A. *Data flows in future services*

The future vision of ubiquitous services [6] proposes the increase of interaction with environment through actuators and data collection through variety of sensors and sensor networks. In such scenario it is nearly impossible to adapt agents to use different information sources in a variety of concrete environment scenarios. One way to deal with the issue is to provide required information in a standardized way or in a set of standard representations. Such a task can be devoted to special agents, which perform the transformation of incoming data into some intermediate form. The format of output data must not confirm to the required standard, but could have a preliminary format. Then another agent cold perform another transformation and so on, until the initial data is refined into the required representation.

Such data refinement structure could be a part of a smart space, as the smart space paradigm is spatial, i.e. tided with some location. The data refinement process can be seen as a dataflow network. A dataflow network - is a set of concurrently executing processes, which communicate by sending data along FIFO channels [14]. In a dataflow network data coming from sensors will eventually be refined. In such scenario, the dataflow network consisting out of agents should provide a reliable service and become an important component for smart applications functioning.

B. *Dataflow network model*

There is a number of dataflow network models present, which describe networks differ in a number of characteristics [15]. For our study, we defined two requirements for the dataflow model. Firstly, the selected model could be used to describe the vastest class of such networks. Secondly, it should have some properties, which could be useful for implementation of the substitution mechanism.

One of the most valuable properties of dataflow networks for substitution mechanism is the compositionality principle, which states that the behavior of the system could be gained from the behavior of it components. Also this means that two components can be replaced inside the compositional network if they have the same operation semantics. The vast class of networks is the nondeterministic networks using asynchronous channels for communication.

There are several fully abstract models [14], [15] that define asynchronous nondeterministic networks. They are fully abstract to the internal structure of the elements, focusing solely on it behavior. These models differ in the formalism they use.

In order to study the dataflow networks, which are formed on top of RDF store, we need to propose a model of such network. There are several asynchronous data flow network models, which provide description for the same class of network, but only differ in formalism they use.

For our study we have chosen the asynchronous network model presented in [14]. The semantics of dataflow network is given as the function from tuples of infinite sequences of finite words to sets of infinite sequences of finite words. The network model is compositional. The operational semantics of the network is described as the work of automaton, which has several input and output lines. An output line may be connected with an input line, therefore forming the feedback loop. Each communication line is the queue, which allows to put any number of elements to the end and read element one-by-one from the beginning. The example of automaton manipulating three lines of each type is shown on Fig. 1.

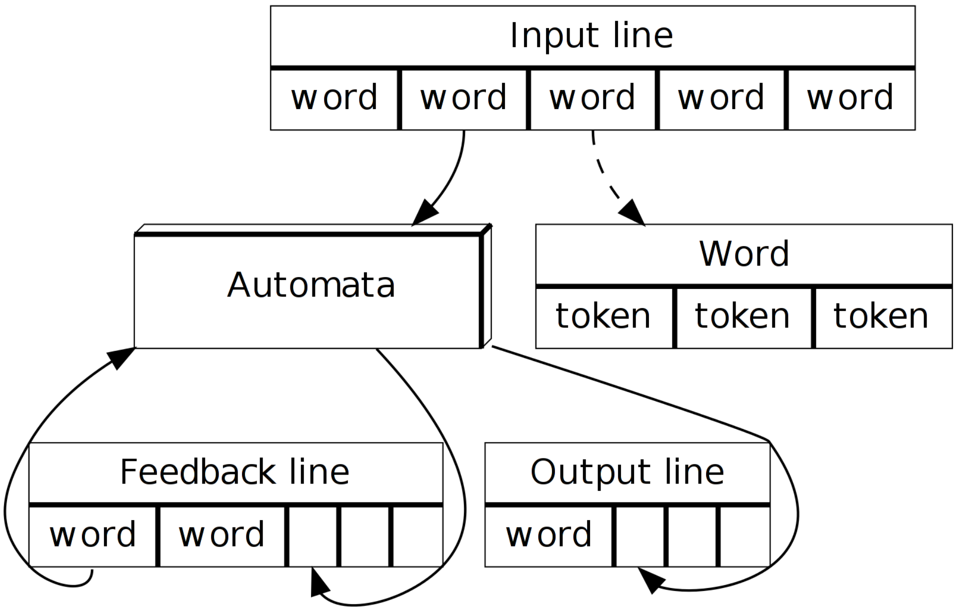

Fig. 1. The example of automata, having one input, one output and one feedback line

Each element on communication line is the word or an empty statement. The word consists out of several tokens. The automaton reads words from input lines and places corresponding values on output lines. The operation of network may be non deterministic, therefore there is a set of possible words that might be places onto the output lines. The notion of word is very important, as it allows to denote the required minimum amount of data for automaton to produce meaningful output result. The order of words on input lines and output lines is determined by the function.

C. KP operational description

Operation of a dataflow network is determined by the functionality of two key components: the computational KP and the communication middleware. The behavior of the computational KP is complex, so three stages of the life cycle are depicted: initialization, operation and disconnection. During each stage the KP operations are determined. As with the original model, we do not place any restrictions on internal structure of the KP, only on it interaction with SIB. The sequence diagrams for each phase is shown on Fig. 2, 3 and 4.

a) Stage 1 - initialization: During initialization phase the KP connects to RDF store and retrieves information via a series of queries. Then it subscribes to the RDF triples, which form the input dataflow for the KP. If there are no such triples available KP may disconnect from the store. Triples, which contain the result of data computation, are named production. Initial value for production triples can be placed into the store later on based on acquired information.

The Smart-M3 platform by default allows every KP to modify arbitrary triples in the store. This policy may allow to replace production values of the KP with inappropriate data. To prevent such uncontrolled behavior the computational KP may put a lock on production triples, so they can only be modified by the KP. Such mechanism can be used also to prevent multiple

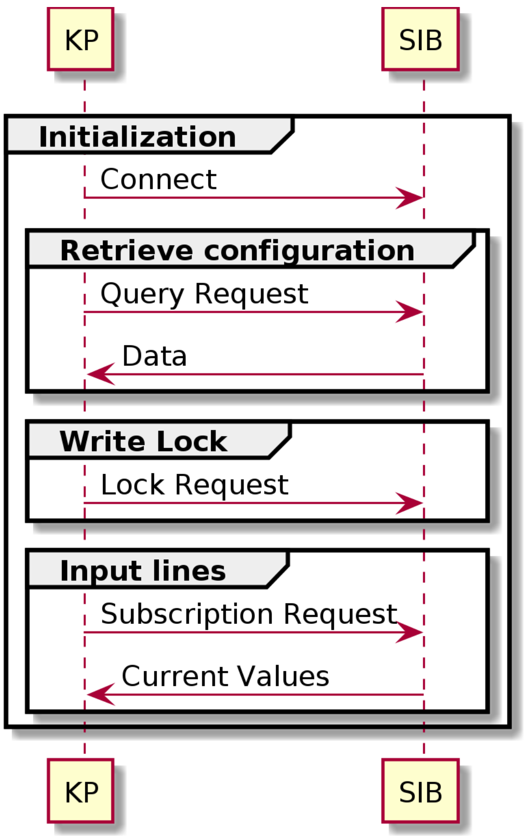

Fig. 2. KP initialization phase

KPs from sending the same information into the smart space. If such overlapping is found, the KP may stop its operation or change output triples.

b) Stage 2 - operation: The operation phase can be named as a production phase, as the computational KP is generating output values. The KP begins its operation on subscription notification. Along with triples, which were received as the subscription payload, the KP may query additional data from the store to produce values. The information received via queries should be served only as a support mechanism, when only the last value of the triple is required for calculation. When the computation is done, the KP may place calculated values.

The KP may update the subscriptions according to the information coming from existing ones. It may either put new subscription to the store or unsubscribe from active one. The computational KP must not act on events that are coming from other sources, e.g. external timers. The information provider KPs may place their values any time they want, but they are not allowed to consume information from other dataflow sources.

c) Stage 3 - disconnection: When the KP disconnects from the dataflow, it removes all active subscriptions. The last output values may either be removed or be replaced with values indicating the interruption of operations. If the output triples were protected by the KP, the corresponding protection is removed. Lastly, the KP disconnects from RDF store.

D. Communication lines and compositional property

As stated previously, the input line of KP is modeled as concrete triple in smart space read by the KP. A set of triples read by the KP form the input lines. The output lines

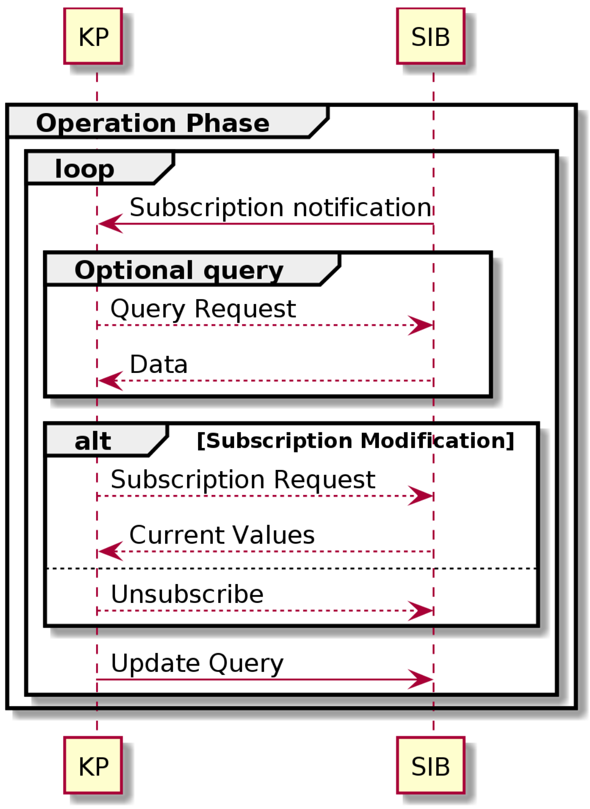

Fig. 3. KP operation phase

are the corresponding write triples. The communication mechanism described in base model is reliable, e.g. all information written to the line must be transmitted to the consuming agent. The SIB does not provide any mechanism to store a sequence of values, therefore KP consuming information must either store it in their internal memory or put it into the smart space in some form and manually manage the storage. One possible way is to store data inside the linked list.

The notion of word in the original model denotes the need for the concrete order of data transfer between the elements of the network. Each producing KP must specify output as a set of packets - the values of triples and the order they are updated. The input of the elements must treat data the same way. Having these constrains applied, the dataflow network is compositional, therefore complex systems may be created out of replaceable components.

The implementation of dataflow network on RDF store was designed basing on the original model of the network, so such network should be compositional. The original model describes the semantics of dataflow network operation, focusing solely on input and output operations of the agent. The developed one describes all operations of the KP, including the setup and the tear down operations. The latter ones are required to organize the operation of whole dataflow network and do not interfere with the properties of operation.

E. Data flow network example

The dataflow could be potentially found in most of existing smart space applications, though we will use a custom model. The system under study is the part of smart home environment, responsible for managing the climate in the building. The elements of the system

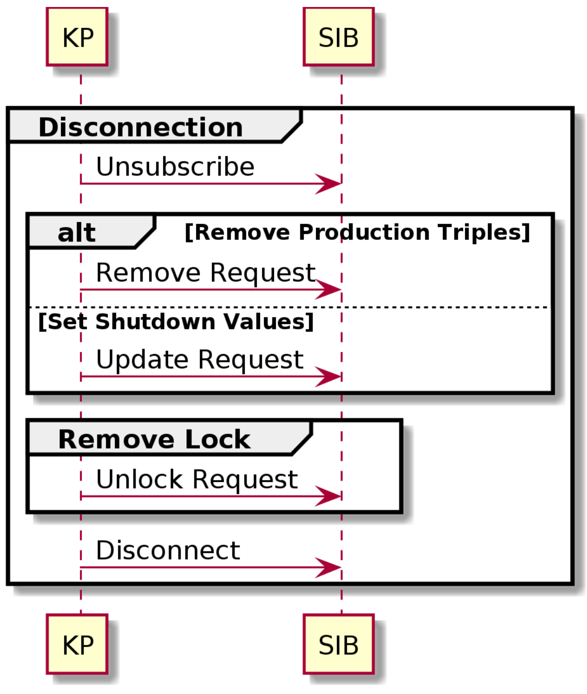

Fig. 4. KP disconnection phase

and the ontology used is shown on Fig. 5. The climate control system in current design is decoupled into several layers: sensors, aggregation service and device control system. In addition, the information about building structure is maintained by another subsystem. The building description includes the description of the rooms, their relative position and the position of the doors and windows.

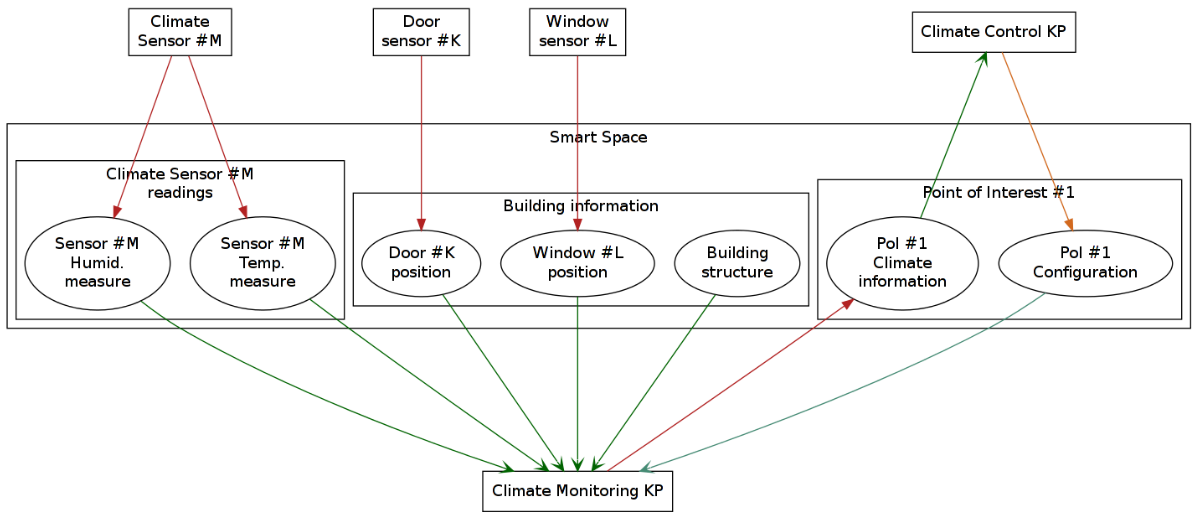

Fig. 5. The example of dataflow network with one computational node depicted - Climate Monitoring KP

Raw climate data is coming from temperature and humidity sensors, which are placed all over the space under observation. Sensors may be placed outside the observed place to track the conventional flows coming and leaving the building. Besides measurements, each sensor provides information about its location in the environment, so it is possible to calculate temperature flows.

Aggregation service is a KP, whose aim is to provide climate information in points of interest. Such information includes the values of temperature, humidity and sensible temperature. For a smart house a point of interest could be a room in the house or basement, for a gym it could be several locations near-by and far away from the air conditioning systems.

Based on climate information in points of interests, cooling system could change modify the settings of cooling system to better fit required conditions. Climate information can be provided directly to the end user by visualization KPs, so the user could modify the behavior of the system or choose an appropriate clothing for example.

IV. Dataflow disruption cases

The dataflow network on RDF store consists out of many computational and information providing KPs. These units interact with each other by the means of shared RDF store. The systems of such complexity tend to have many problems. We have picked out four groups of possible dataflow disruption: the RDF store malfunction, information tampering, the KP breaking the communication protocol or stopping its operation. The description of each case with an approach to handle the situation is given further.

a) Case 1 - RDF store malfunction: Presented model of dataflow network depends heavily on RDF store operation, therefore the failure of the store would prevent the operation of whole network. The approaches for maintaining the operations of the store depends on the concrete architecture of the store. For example, the smart space of Smart-M3 platform may consist out of several SIB, so if one of them discontinues the operation, the other may support the operation of running smart application. In this work we consider the possibility of store breakage as a low one.

b) Case 2 - information tampering: The computational KPs interact with each other by the means of triple modification in the store. If a KP accidentally or purposely modifies the value of the production triple of another KP, then the communication protocol is interrupted. The possible ways to overcome the issue is to enable access control either on the KP level or on the data level. In the first case only legitimate agents are allowed to interact with the store. In the second case access control is provided on the level of triples. The support for the latter case is implemented for Smart-M3 platform.

c) Case 3 - communication protocol breakage: The communication protocol between KPs may be interrupted due to possible issues in network packet delivery between the KP and the store. Firstly, the modification transaction could be missed due to packet loss. Secondly, the data could be modified using man-in-the-middle attack. The issues of sustaining the operation due to the protocol inconsistencies is out of the scope of the paper and is a possible direction for the future research.

d) Case 4 - KP loosing connection: The last reason for dataflow disruption is the temporary disconnection of the computational KP from the RDF store. When computational KP looses connection with the RDF store the work of other KPs, which rely on information coming from the failing KP, is compromised. They may wait for the data for a long time until the first KP restores its connection with the store. In the worst case scenario they may wait for data for an infinite time. This case could cause the most trouble as the number of KPs running on an agents with limited power supply should increase in the future.

V. KP SUBSTITUTION MECHANISM

A. *KP substitution approach*

There are two possible was to deal with the KP disconnection issue. The first way is to place some values into output triples of the KP, which are close enough to real production information of the KP until the real one returns into the dataflow network. The other way is to put values, which would indicate the termination of data production, so consumer KPs may update their operation accordingly, find alternative information sources or terminate their operation.

We propose to extend the operation of SIB to provide support for substitution of the computational KP with a fall back one when the connection between the main KP and the store is lost. The fall back KP then sustains the output of the original KP or proceeds with the shutdown sequence. When the computational KP reconnects with the store the operation of the fall back KP is terminated. With such approach implemented, the substitute KP may notify the consumers, so first approach can be implemented too, if required. When implementing such mechanism the precautions for breaking communication protocols during the KP substitution should be taken.

B. *Computational KP operation modification*

The behavior of a computational KP is modified to support the operation of the substitute mechanism. The change relates only to the initialization and shutdown phases and does not affect the operation phase of the KP, therefore properties of the network remain the same.

According to new scenario the initialization of the computational KPs starts with the configuration of the KP, e.g. it may query configuration and environment information form the RDF store. Next step is the request for registration as a dataflow KP. Along with request the KP specifies the following information:

- URI of the operational description of the KP.
- Type of the fall back program, which should be executed by substitute KP.
- The program for substitute KP.

The result of the request indicates that the KP is either registered as a new dataflow KP or supposed to resume the interrupted operation.

In the first case, the KP should continue the initialization process as a plain KP in dataflow network, e.g. subscribe to required triples, which form input data stream, put write-protection lock for production triples and continue to operational phase.

In the second case, the KP returns to the dataflow network and takes control back from the fall back KP. With the answer, the SIB returns the list of known subscriptions and the data, which was not processed by the substitute KP. The data transmission is required to prevent the dataflow disruption. The write-protection for production triples is set to accept input from the computation KP.

C. *Substitute KP operation*

There could possibly be several substitute KPs, which can execute programs of different types. The fall back KP begins operation by sending a request to the SIB to register itself as a substitute KP. Along with the request it denotes the type of the program it is able to execute. Then KP goes into idle state and makes no operation at all.

The operation is resumed when it is required to replace the failing dataflow KP. SIB sends the fall back program to execute and the list of active subscriptions along with data, which was not processed by the computational KP. The write-protection for production triples are moved from the failing KP to the substitute KP.

The fall back program in general case should read the configuration information from the smart space via a series of queries, then process the data passed directly from the SIB and move to the operation phase.

When computational KP returns to the dataflow network, SIB notifies substitute KP with stop operation message. Upon receiving such message, fall back KP must shutdown the execution and free all captured resources. It should not send backward notification to SIB.

D. Substitution module

Given the description above, SIB must maintain the list of registered computational KPs and substitute KPs. The computational KPs are described with the unique URI of their operation, the fall back program and the set of active subscriptions. If KP decides to modify the subscriptions, then corresponding changes must be made to the list. The fall back KPs are described with the type of the program they are able to execute and the current state: whether it executes a program of some KP or awaits for job notification.

The core of transition operation is the same for each substitution cases. During the transition SIB module firstly begins to collect information from active subscriptions of KP being replaced, then it transfers write protection locks from one replacing KP to its substitute. When the transfer is complete, the module notifies the new KP with the set of last known subscriptions and not processed data.

The proposed KP substitution mechanism should be implemented as an add-on to the SIB, because it must detect the disconnection of KP from the store and gather all unprocessed data during KP substitution.

VI. Conclusion and future work

In current paper we have presented the definition for dataflow network on RDF store. Such networks could be efficiently used in ubiquitous environments, which include multiple sensors and processing elements. The operation of such network could be compromised in a several ways, we have depicted four possible cases of dataflow disruption. One of them is the loss of connection with RDF store by the computational node. We proposed the creation of special mechanism for maintaining the operation of the dataflow network, which would act as an additional service for the platform. The description of mechanism operation and corresponding KPs operation is given. We are planning to implement such service for Smart-M3 platform and test its operation on existing services and new one.

VII. Acknowledgment

Authors would like to thank FRUCT association for provided support and R&D infrastructure. We would like to thank Vesa Luukkala for providing feedback and guidance.

The paper is a part of the research carried out within the project “Cross-platform future services - technologies of smart space and internet of things” funded by grant of Federal Program “Scientific and scientific-pedagogical personnel of innovative Russia in 2009-2013” contract 14.B37.21.0876.

References

[1] G. Abowd, E. Mynatt, and T. Rodden, The human experience of ubiquitous computing, *Pervasive Computing, IEEE*, vol. 1, no. 1, pp. 4857, 2002. DOI: 10.1109/MPRV.2002. 993144.

[2] D. Keling, M. DALMAU, and P. ROOSE, A survey of adaptation systems, *International Journal on Internet and Distributed Computing Systems*, vol. 2, pp. 123140, 2012.

[3] I. Oliver, "Information spaces as a basis for personalising the semantic web", *Proc. 11th International Conference Enterprise Information Systems (ICEIS)*, SAIC, May 2009, pp. 179-184.

[4] D. Korzun, I. Galov, A. Kashevnik, N. Shilov, K. Krinkin, Y. Korolev, "Integration of Smart-M3 applications: Blogging in smart conference", *Smart Spaces and Next Generation Wired/Wireless Networking*, June 2011, pp. 51-62.

[5] G. Landi, G. Laura, V. Memeo, P. Pucci, and S. Rapp, A unified smart city environment based on sofias interoperability open platform, *SOFIA Artemis*, 2009.

[6] F. Mattern and C. Floerkemeier, From the internet of computers to the internet of things, *From active data management to event-based systems and more*, pp. 242259, 2010. DOI: 10.1007/978-3-642-17226-7_15.

[7] J. Honkola, H. Laine, R. Brown, and O. Tyrkkö, Smart-m3 information sharing platform, in *Proc. IEEE Symp. Computers and Communications*, ser. ISCC 10, Riccione, Italy: IEEE Computer Society, Jun. 2010, pp. 10411046, ISBN: 978-1-4244-7754-8. DOI: http://dx.doi.org/10.1109/ISCC.2010.5546642.

[8] V. Luukkala, J. Honkola, "Integration of an answer set engine to smart-M3", *Smart Spaces and Next Generation Wired/Wireless Networking*, 2010, pp. 92-101. DOI: 10.1007/978-3-642-14891-0_9.

[9] S. Pantsar-Syvaniemi, E. Ovaska, S. Ferrari, T.S. Cinotti, G. Zamagni, L. Roffia, S. Mattarozzi, V. Nannini, "Case study: Context-aware supervision of a smart maintenance process", *IEEE/IPSJ International Symposium on Applications and the Internet*, IEEE, 2011, pp. 309-314. DOI: 10.1109/SAINT.2011.59.

[10] V. Luukkala and I. Niemel'a, Enhancing a smart space with answer set programming, *Semantic Web Rules*, pp. 89103, 2010. DOI: 10.1007/978-3-642-16289-3_9.

[11] D.G. Korzun, S.I. Balandin, V. Luukkala, P. Liuha, A.V. Gurtov, "Overview of Smart-M3 Principles for Application Development", *Proc. Congress on Information Systems and Technologies (IS&IT'11), Conf. Artificial Intelligence and Systems (AIS'11)*, volume 4, 2011, pp. 64-71.

[12] A. DElia, J. Honkola, D. Manzaroli, and T. Cinotti, Access control at triple level: specification and enforcement of a simple rdf model to support concurrent applications in smart environments, *Smart Spaces and Next Generation Wired/Wireless Networking*, pp. 6374, 2011. DOI: 10.1007/978-3-642-22875-9_6.

[13] C. Bettini, O. Brdiczka, K. Henricksen, J. Indulska, D. Nicklas, A. Ranganathan, and D. Riboni, A survey of context modelling and reasoning techniques, *Pervasive and Mobile Computing*, vol. 6, no. 2, pp. 161180, 2010. DOI: 10.1016/j.pmcj.2009.06.002.

[14] J. Kok, A fully abstract semantics for data flow nets, in *PARLE Parallel Architectures and Languages Europe*, Springer, vol. 259, 1987, pp. 351368. DOI: 10.1007/3-540- 17945-3n 20.

[15] B. Jonsson, "A fully abstract trace model for dataflow and asynchronous networks", *Distributed Computing*, number 4, volume 7, 1994, pp. 197-212. DOI: 10.1007/BF02280834.