On-the-Fly Ontology Matching for Smart M3-based Smart Spaces

Alexander Smirnov, Alexey Kashevnik,
Nikolay Shilov
SPIIRAS
St. Petersburg, Russia
{smir; alexey; nick}@iias.spb.su

Sergey Balandin, Ian Oliver, Sergey Boldyrev
Nokia Research Center
Helsinki, Finland
{sergey.balandin; ian.oliver;
sergey.boldyrev}@nokia.com

***Abstract* — Proper functioning of smart spaces demands semantic interoperability of the knowledge processors connected to it. As a consequence it is required to develop models that would enable knowledge processors to perform on-the-fly translation and interpretation between the internal and smart spaces ontologies. The paper presents our solution to the above stated problem, which has been implemented for Smart-M3 platform.**

Smart spaces; ontology matching; semantic similarity; semantic interoperability; Smart-M3

I. INTRODUCTION

Smart-M3 is an open source software platform [1] that aims to provide "Semantic Web" [2] information sharing infrastructure between software entities and various types of devices. The platform combines ideas of distributed, networked systems and Semantic Web [3]. The major application area for Smart-M3 is the development of smart spaces solutions, where a number of devices can use a shared view of resources and services [4], [5]. Smart spaces can provide better user experience by allowing users to easily bring-in and take-out various electronic devices and seamlessly access all user information in the multi-device system from any of the devices.

The simplified version of the Smart-M3 smart spaces reference model is shown in Figure 1. The Knowledge Processors (KPs) represent different applications that use the smart space. The smart space core is implemented by one or several Service Information Brokers (SIBs) interconnected into the common space. The information exchange is organized through transfer of information units (represented by RDF triples) from KPs to the smart space and back. The information submitted to the smart space becomes available to all KPs participating in the smart space. The KPs can also transfer references to the appropriate files/services into the smart space, since not all information can be presented by RDF triples (e.g., a photo or a PowerPoint presentation). As a result the information is not really transferred but shared between KPs by using smart space as a common ground.

However, real implementation of any smart spaces solution faces a number of problems. Let’s consider a simple case study when a user is having his/her mobile device with KP running on it. Assume that the user KP (UKP) is configured to make some presentation. The LCD projector represented in the smart space by Projector KP (PKP) is a key enabler of this functionality. But in order for the presentation to be shown, the UKP has to share the information about the presentation location (URI) with the PKP. As a result the following conditions have to be fulfilled:

1) The UKP has to know that the PKP is a part of the smart space;
2) The UKP has to share the presentation’s URI in such a way that the PKP can understand it.

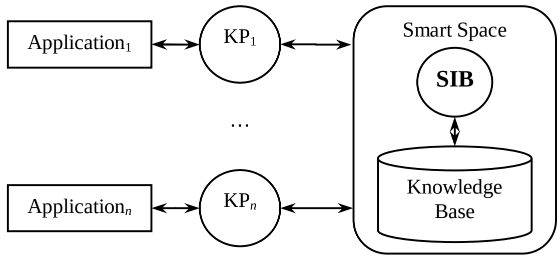

Figure 1. Smart space based on Smart-M3: simplified reference model.

Even from this simple example we can see that proper functioning of the smart space requires semantic interoperability between knowledge processors connected to it. As a consequence it is required to develop a model that would enable knowledge processors to translate on-the-fly between their internal and smart space’s ontology to a certain extent. In this paper, we present our solution for the above stated problem.

II. SIMILARITY IN SMART SPACES

All the similarity metrics in the performed state-of-the-art review are based on the two information retrieval metrics of precision and recall. As it was mentioned earlier, in case of smart spaces it is necessary to avoid false alignments, so the precision should be preferred above the recall. This is achieved via choosing the right threshold value. The possibility of choosing the right threshold value has to be taken into account in the development of the matching models.

Since in smart spaces most of knowledge processors are problem-oriented it should be proposed to utilize reusable ontology patterns for ontology creation. This would enable unification and standardization of the ontologies and significantly simplify ontology matching.

Table 1 summarizes the results of the state-of-the-art.

 ISBN: 978-1-61208-100-7

TABLE 1. STATE-OF-THE-ART SUMMARY

Criteria	Possibilities / Requirements
Agent anonymity	Matching approach for different ontologies Matching of different versions of the ontology
Information/ knowledge representation format	Format independent RDF KIF Graph-based formats
Ontologies aligned	Any Lightweight Large-scale
Automation	Semi-automatically Automatically
Algorithm complexity	High complexity Low complexity $O(N^2)$, N – the number of elements in the contexts of the concept to be matched
Precision	Supported N/A
Matching method class(es)	Contextual: distribution-based graph-based structural similarity structural propagation Linguistic: similarity-based Statistical: data type compatibility Combined: pattern-based heuristics rule-based
Usage of synonyms	No synonyms Synonyms supported: Thesaurus-based WordNet-based
Ontology element matching	One-to-one Any to any entities matching One to any entities matching supported
Internet usage	Internet is used Internet is not used Internet can be used

Based on it, the following concluding remarks can be made.

The goal of ontology matching is basically solving the two major problems, namely:

1) Ontology entities which have the same name can have different meaning.

2) Ontology entities which have different names can have the same meaning.

For this purpose a number of techniques are applied in different combinations. These techniques include:

1) Identification of synonyms

2) Similarity Metrics (name similarity, linguistic similarity)

3) Heuristics (for example two nodes are likely to match if nodes in their neighborhood also match)

4) Compare sets of instances of classes instead compare classes

5) Rules: for example, if class A1 related to class B1 (relation R1), A2 related to class B2 (relation R2) and B1 similar to B2, R1 similar to R2 therefore A1 similar to A2.

As a result of matching the following types of elements mapping proximity can be identified:

1) One-to-one mapping between the elements (Associate-Professor to Senior-Lecturer)

2) Between different types of elements (the relation AdvisedBy(Student, Professor) maps to the attribute advisor of the concept Student)

3) Complex type (Name maps to the concatenation of First Name and Last Name)

All methods can be separated into the following four groups.

A. *Linguistic methods*

These methods are focused on determining similarity between entities based on linguistic comparison of these entities (count of the same symbols estimation, estimation of the longest similar parts of words, etc.).

B. *Statistical methods (instance based)*

These methods compare instances of the ontology entities and based on this estimation entities can be compared.

C. *Contextual methods*

The aim of the contextual similarity is to calculate a measure of similarity between entities based on their contexts. For example if parents and children of the ontology classes are the same consequently the classes also the same.

D. *Combined methods*

These methods combine specifics of two or three of the above methods.

In the M3 approach, there is no strict definition of instances and differentiation of them is not an easy tasks. Because of this reason the techniques and methods relying on instances were not considered for further development. Hence, the developed models presented below integrate all of the above techniques (except those dealing with instances) and propose a set of combined methods having features of the linguistic and contextual methods.

III. PRINCIPLES FOR CREATING ONTOLOGICAL DESCRIPTION FOR SMART SPACE KNOWLEDGE PROCESSORS

In this section, seven principles for creating ontological description for smart space knowledge processors are proposed. The correspondences between the principles and the criteria are indicated in Table 2.

A. *Synonyms*

Synonyms of the used in the ontological description terms have to be provided. Synonyms can be provided as additional RDF-triples. For example:

RDF Triple:
(“URI”, “is”, “http://myexample.com/pr1.ppt”)
Synonym1:
(“URL”, “synonym”, “URI”)
Synonym2:
(“location”, “synonym”, “URI”)

TABLE 2. STATE-OF-THE-ART SUMMARY

Criteria	Corresponding Principles
Agent anonymity	Taken into account in the matching approach
Information/knowledge representation format	Connected graph-based RDF-triples (subsec. C) Fullness and consistency of the description of the knowledge processors possible actions (subsec. E) Taken into account in the matching approach
Ontologies aligned	Fullness and consistency of the description of knowledge processors possible actions (subsec. E) Usage of ontology patterns (subsec. G)
Automation	Triples for determining format for values (subsec. F) Taken into account in the matching approach
Algorithm complexity	Taken into account in the matching approach
Precision	Taken into account in the matching approach
Matching method class(es)	Synonyms (contextual methods) (subsec. A) Abbreviations shouldn't be used (linguistic methods) (subsec. B) Connected graph-based RDF-triples (contextual methods) (subsec. C) Combined methods are used in the matching approach
Usage of synonyms	Synonyms (subsec. A) Abbreviations shouldn't be used (subsec. B) Taken into account in the matching approach
Ontology element matching	Homogeneous ontological description of elements (subsec. D) Taken into account in the matching approach
Internet usage	Taken into account in the matching approach

B. *Abbreviations shouldn't be used*

Abbreviations of the ontological description terms complicate matching of the ontologies. They should be avoided in ontological description.

Instead of using triple:

("START_D", "is", "06+11+2004") – VCal

or

("DTSTART", "is", "20041106") – iCal

The following rule should be used

("start date", "is", "2004-11-06")

If it is not possible avoiding using abbreviations (e.g., when they are standardised like in the examples above) than it is recommended to resolve this potential problem by adding appropriate synonyms for these abbreviations, for example:

("START_D", "synonym", "start date")
("DTSTART", "synonym", "start date")

C. *Connected graph-based RDF-triples*

RDF-triples have to be joined with each other. This allows to process such ontological descriptions as a graph and to apply graph-based methods of ontology matching.

("URI", "is", "http://myexample.com/pr1.ppt")
("slide", "is", "5")

It is important to connect the above triples, by complimenting database by the following additional triples:

("slide", "part_of", "presentation")
("presentation", "property", "URI")

D. *Homogeneous ontological description of elements*

Ontological description elements have to be homogeneous. For example, if an element is a subject or object in one rule it cannot be a predicate in another rule.

Instead of using triple:

("presentation", "URI", "http://example.com/pr1.ppt")

The following two rules should be used:

("URI", "is", "http://example.com/pr1.ppt")
("URI", "part_of", "presentation")

The triples describing synonyms are specific triples and they can contain predicates as subjects and objects.

E. *Fullness and consistency of the description of the knowledge processors possible actions*

Ontological descriptions of the knowledge processors have to include all their possible actions and relations, but at the same time shouldn't include any unnecessary information.

An example of projector ontology RDF triples is presented as a graph in Figure 2 and as a list of corresponding triples under the picture.

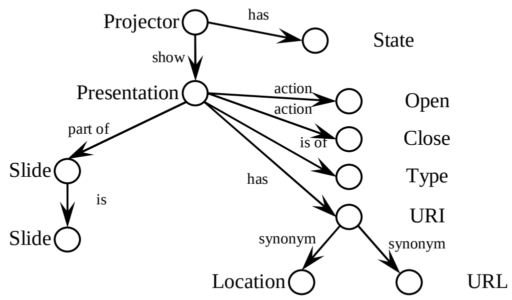

Figure 2. Ontology of the projector.KP.

("projector", "has", "state")
("projector", "show", "presentation")
("presentation", "action", "open")
("presentation", "action", "close")
("slide", "part of", "presentation")
("slide", "is", "Number")
("presentation", "has", "URI")
("presentation", "is of", "Type")
Synonyms:
("URL", "synonym", "URI")
("location", "synonym", "URI")

F. *Triples for determining values format*

In case of complex formats of data values it is needed to add special triples into the ontology, which describe value format. For example, vCal has description:

"Start_T", "value", "07+29+30")

To let the system recognize this time format it is needed to add the following triple:

("Start_T", "format", "hh+mm+ss")

G. *Usage of ontology patterns*

Most of knowledge processors are task-oriented and it is expected that there will be many knowledge processors

 ISBN: 978-1-61208-100-7

performing the same functions (e.g., knowledge processors representing functionality of LCD projectors). As a result utilizing reusable ontology patterns for ontology creation is proposed. This would enable unification and standardization of the ontologies and significantly simplify ontology matching. Such patterns should be problem-oriented as knowledge processors themselves. As an example of such a pattern the projector knowledge processor ontology presented in Figure 2 can be considered.

IV. MULTI-MODEL APPROACH FOR ON-THE-FLY MATCHING KPS AND SIB ONTOLOGY

The below proposed approach allows matching of KPs and SIB ontology for the interoperability purposes and is based on the ontology matching model illustrated by Figure 3. The approach takes into account that the matching procedure has to be done “on-the-fly” by mobile devices with limited resources and remembering the fact that knowledge processors are responsible for performing certain concrete and well-described tasks, which means that the corresponding ontology generally should be small–to–medium size and describe only very limited domains.

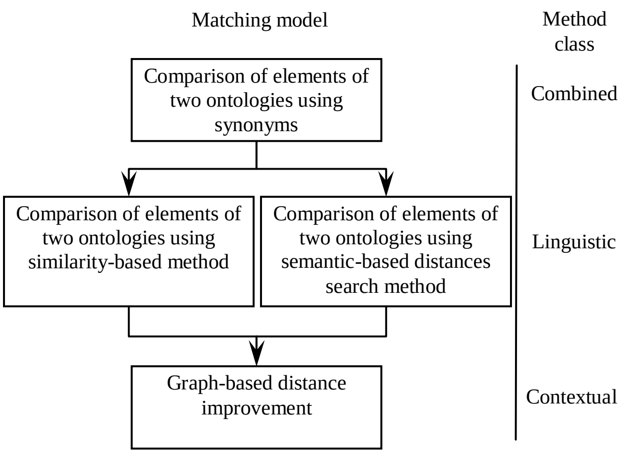

Figure 3. Multi-model approach to on-the-fly matching KP and SIB ontology.

Ontology is represented as RDF triples, consisting of the following ontology elements: subject, predicate, object. Degree of similarity between two ontology elements is in the range [0, 1]. The approach consists of the following steps:

1) Compare ontology elements taking into account synonyms of both ontologies. The degree of similarity between equal elements is set to 1 (maximum value of the degree of similarity).

2) Compare all elements between two ontologies and fill the matrix M using *similarity-based model* described in Section V. Matrix M is of size m to n, where m is the number of elements in the first ontology and n is the number of elements in the second ontology. Each element of this matrix contains the degree of similarity between the string terms of two ontology elements using the fuzzy string comparison method described in Section III-B.

3) For knowledge processors, which can access Internet, e.g., WordNet or Wiktionary, the model of searching semantic distances was developed.

a) Compare all elements of two ontologies and fill the matrix M’. Matrix M’ is of size *m* to *n*, where *m* is the number of elements in the first ontology and *n* is the number of elements in the second ontology. Each element of this matrix represents the degree of similarity between two ontology elements.

b) Update values in matrix M, where each new value of elements of M is the maximum value of (M, M’)

4) Improve distance values in the matrix M using the *graph-based distance improvement model* described in Section VI.

As a result the matrix M contains degrees of similarity between ontology elements of two knowledge processors. This allows determining correspondences between elements by selecting degrees of similarities which are below than the pre-selected threshold value.

The next sections describe major elements of the proposed approach in details.

V. SIMILARITY-BASED MODEL FOR ONTOLOGY MATCHING

The similarity-based model for the ontology matching is presented in Figure 4. It contains a stemming procedure to normalize words, improved fuzzy string comparison procedure, and normalization procedure. The normalization procedure makes it possible to reduce the resulting similarity for its easier interpretation and is not considered here in detail.

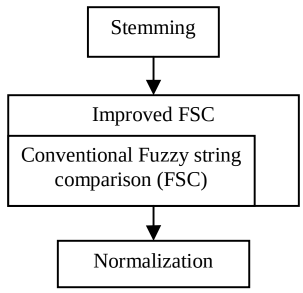

Figure 4. Similarity-based model for the ontology matching.

A. Stemming

To improve the matching quality the application of the stemming procedure is proposed. This operation makes it possible to identify ontology elements even if they are written in different forms. The following conversions can be done: “looking” → “look”, “device” → “devic”, “vertical” → “vertic”, and “horizontal” → “horizont”. This procedure is uniquely tuned for each supported language.

B. Fuzzy string comparison

The basis of the string comparison algorithm is the well-known conventional algorithm that calculates occurrence of substrings from one string in the other string.

1. Perform the comparison based on the above algorithm twice: FC_1 = FuzzyCompare(Element$_1$, Element$_2$) and FC_2 = FuzzyCompare(Element$_2$, Element$_1$).

2. Calculate the result as an aggregation of the above results in accordance with the following formula:

$Re' = n*FC_1+(1-n)*FC_2$, where n is a weight, $n \in [0;1]$.

$n = 0.5$ sets the same weight to the both strings, $n = 0$ searches only Request within Class, and $n = 1$ searches only Class within Request. It is proposed to set $n = 0.5$.

VI. Graph-based Distance Improvement Model

The graph-based improvement model for propagation similarities from one ontology element to another is presented in Figure 5. The main goal of this model is to propagate the degree of similarity between closely matching ontology elements to ontology elements related to them through RDF triples.

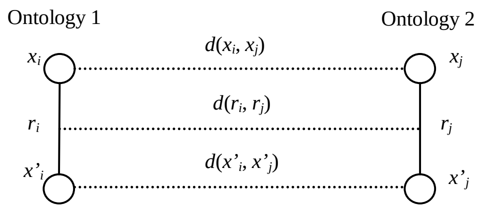

Figure 5. Matching of two ontology model.

Set $X=(x_1, x_2, ..., x_n)$ is the set of subjects and objects in the ontology of two knowledge processors. Set $D_x = (d(x_i, x_j), ...)$ is a degree of similarity between x_i and x_j. Set $R = (r_1, r_2, ..., r_n)$ is a set of predicates in the ontology of two knowledge processors. Set $D_r = (d(r_i, r_j), ...)$ is a set of degrees of similarity between r_i and r_j. Constant Tr is a threshold value that determines whether two ontology elements mapped to each other or not.

The following algorithm allows propagating similarity distance to RDF subjects and objects.

d(x$_i$, x$_j$) = maximum(D$_x$)
while (d(x$_i$, x$_j$) > Tr) do
  for each d(x'$_i$, x'$_j$) as x$_i$ r$_m$ x'$_i$ and x$_j$ r$_l$ x'$_j$ do
    if d(r$_m$, r$_l$) > Tr then
      $d(x'_i, x'_j) = \sqrt[2]{d(x_i, x_j)*d(x'_i, x'_j)}$
    endif
  endfor
  Exclude d(x$_i$, x$_j$) from D$_x$
  d(x$_i$, x$_j$) = maximum(D$_x$)
endwhile

The following algorithm allows propagating similarity distance to RDF predicates.

for each d(x$_i$, x$_j$) > Tr do
  for each d(x'$_i$, x'$_j$) > Tr as x$_i$ r$_m$ x'$_i$ and x$_j$ r$_l$ x'$_j$ do
    $d(r_m, r_l) = \sqrt[3]{d(x_i, x_j)*d(x'_i, x'_j)*d(r_m, r_l)}$
  endfor
endfor

VII. Smart-Room Case Study

This is an extended use case scenario originally proposed in one of our previous publications [6]. A meeting takes place in an "intelligent room" that is equipped with LCD projector, whiteboard, and access to Internet-based translation service.

Users that are planning to make presentation have special knowledge processor (called User Knowledge Processor or UKP) installed on Nokia MAEMO device and it implements the required functionality as described below. Upon event of user entering to the smart space meeting room, at least the following information from users' mobile devices become accessible for other smart space UKPs:

- user profile information (name, photo, domain of interests, e-mail, and phone number, etc.);
- presentation information (title, keywords, URI).

Before the meeting starts the agenda is automatically built and shown on the whiteboard including speakers' names, photos, and presentation titles. The current presentation data is highlighted on the screen. All meeting participants can see the detailed agenda on the screens of their personal mobile devices.

Users can change their user profile items. For this purposes the appropriate GUI has been implemented. When the user changes the information about his/her presentation in the profile, UKP changes the appropriate rules in the smart-space. A GUI interface for visualizing the detailed agenda on the screen of participants' MAEMO devices has been implemented as well.

At the scheduled time the appropriate presentation starts automatically, i.e., the LCD projector is switched ON and the appropriate presentation is shown. The user can control the slideshow directly from the mobile device. Five minutes before the presentation ending time and when the presentation time is about to be over the whiteboard reminds the speaker about the time restrictions.

The overall architecture of this case study is presented in Figure 6. It includes the following knowledge processors:

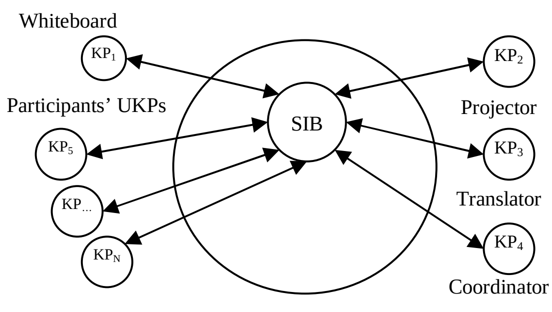

Figure 6. The case study architecture.

- KP1 – whiteboard (PC)
- KP2 – projector (PC)
- KP3 – translator (PC + Internet service)
- KP4 – coordinator (PC)
- KP5… N – UKPs (e.g., Nokia N810)

The application of the matching procedure is shown in Figure 7. Let us consider the following example. The user having mobile device with a knowledge processor (User KP or UKP) running on it and he/she is going to give a lecture. The corresponding ontology is presented in Figure 8.

The following ontology describes the user in respect of giving a lecture:

("user", "gives", "lecture")
("current slide", "part of", "lecture")
("current slide", "is", "Number")
("lecture", "has", "Location")

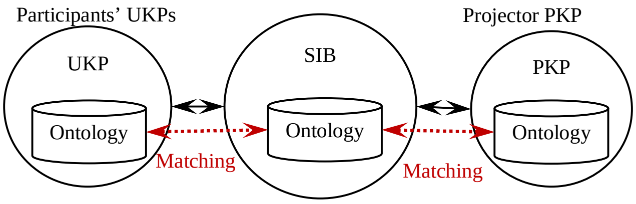

Figure 7. The extended case study architecture.

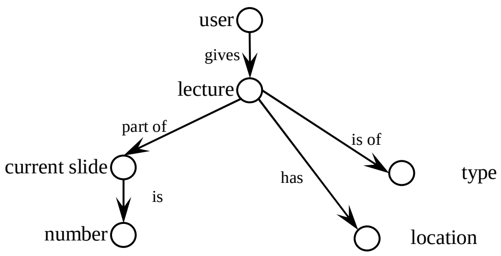

Figure 8. Ontology of user KP.

("lecture", "is of", "Type")

The LCD projector represented in the smart space by Projector KP (PKP) is capable of this function, as it is shown by ontology of the projector KP in Figure 9. This ontology currently located in the smart space. In order for the presentation to be shown, the UKP has to share the information about the presentation location (URI) with PKP.

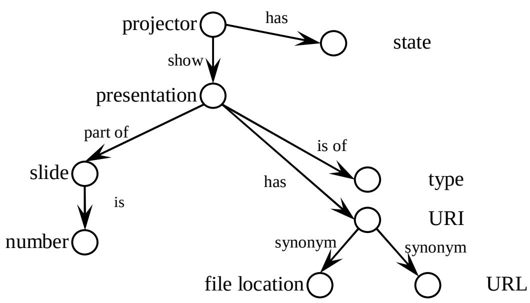

Figure 9. Ontology of projector KP.

The description of corresponding projector ontology is presented below:

("projector", "has", "state")
("projector", "show", "presentation")
("slide", "part of", "presentation")
("slide", "is", "Number")
("presentation", "has", "URI")
("presentation", "is of", "Type")
("URL", "synonym", "URI")
("file location", "synonym", "URI")

For this purposes the UKP and PKP ontology have to be merged. The element "*lecture*" from UKP is merged with the "*presentation*" from PKP as a result of the semantic-based distances search method. Distance between these elements is 0.3. The element "*location*" from UKP is merged with the "*URI*" from PKP as a result of checking the synonyms rules of the PKP ontology.

The element "*Current slide*" is merged with the element "*Slide*". The degree of similarity between these elements estimated via the fuzzy string comparison method is 0.58.

As a result of this matching UKP knows that the following rule has to be added to the smart space to start the lecture:

("URI", "is", "http://example.com/presentation1.ppt")

VIII. CONCLUSION

The paper proposes the multi-model approach to on-the-fly ontology matching in smart spaces. The approach has been developed by integrating the most efficient techniques applicable to Smart-M3 and based on analysis of the state of art. It takes into account that the matching has to be done "on-the-fly" by mobile devices with limited capacities and uses the fact that knowledge processors are responsible for performing certain concrete and well-described tasks. The experiments showed that the matching procedure based on the proposed algorithm takes less than a second on Nokia N810 mobile device.

In this study we could not find fully automated solution for defining threshold value, but we are continuing to think in this direction and your ideas on how it can be solved are very welcome.

ACKNOWLEDGEMENT

The presented work is a result of the joint project between SPIIRAS and Nokia Research Center. Part of the implementation work has been also supported by Open Innovations Framework Program FRUCT – www.fruct.org.

REFERENCES

[1] SourceForge, Smart-M3, http://sourceforge.net/projects/ smart-m3. Retrieved: 10.5.2010.
[2] Semantic Web, www.semanticweb.org. Retrieved: 10.5.2010.
[3] Smart-M3 Wikipedia page, http://en.wikipedia.org/wiki/Smart-M3. Retrieved: 10.4.2010.
[4] Oliver, I and Honkola, J. Personal Semantic Web Through a Space Based Computing Environment. Middleware for Semantic Web 08 at ICSC'08, Santa Clara, CA, USA (2008).
[5] Oliver, I, Honkola, J and Ziegler, J. Dynamic, Localized Space Based Semantic Webs. WWW/Internet Conference, Freiburg, Germany (2008).
[6] Smirnov, A., Kashevnik, A., Shilov, N., Oliver, I., Lappetelainen, A., Boldyrev, S. Anonymous Agent Coordination in Smart Spaces: State-of-the-Art. Smart Spaces and Next Generation Wired/Wireless Networking (ruSmart 2009) conference, Springer, LNCS 5764, pp. 42-51. St-Petersburg, Russia (2009).

 ISBN: 978-1-61208-100-7