Smart-M3 and Geo2Tag Platforms Integration Agent Uses-Cases

Kirill Yudenok
Saint-Petersburg Electrotechnical University
Saint-Petersburg, Russia
kirill.yudenok@gmail.com

***Abstract*—Geo-tagging and smart spaces are two promising directions in modern mobile market. Geo-tagging allows to markup any kind of data by geographical coordinates and time. This is the basis for defining geographical context which can be used in different types of applications e.g. semantic information search, machine-to-machine (M2M) interactions. Smart spaces as the basis for seamless distributed communication field for software services provides semantic level for data processing. Most desired feature of coming software is pro-activeness and context awareness, i.e. services will be able to adapt to the user's needs and situations and be able to manage decisions and behaviors on behalf of the user. The paper is about integration agent (GCSS) architecture and main use-cases.**

I. INTRODUCTION

Nowadays we have two most promising software trends: location based services and pervasive smart environments (smart spaces). Both of them will be a base for user- and machine- oriented proactive services. Smart spaces should provide continuous distributed semantic data and communication field for software services, which is being run on personal devices and autonomous computers and robots. Most desired feature of coming software is pro-activeness and context awareness, i.e. services will be able to adapt to the user's needs and situations and be able to manage decisions and behaviors on behalf of the user [1]. One of the important part of context is location based data. This data is being used for two purposes: for clarifying semantic meaning of queries (when service retrieves the data from smart environment) and for limitation of space of search (usually there is no point to make global search). Geo-coding (or geo-tagging) is the technique of markup real or virtual object by adding geographical coordinates and time. If we consider software, we have only virtual (or digital) objects like media, events, documents etc. So far, smart spaces and geo-tagging systems are being developed mostly separately, there are only few works [2, 3, 4] where software design of smart spaces and geo-tagging integration is discussed.

In this paper focuses on the requirements (use-cases) for Integrated Geo-Coded Smart-Space (GCSS) middleware and also provides some information about the definition of the smart system and examples of its use.

From practical point of view we use Smart-M3[1] [5] and Geo2Tag[2] [6] platform as most developed open source middleware for smart spaces and geo-tagging. Overview of the Smart-M3, Geo2Tag platforms, geo-coding related works and agent integration platforms architecture can be found in article [7].

This paper is organized as follows. Section 2 describes integration agent architecture and main requirements (use cases) of platforms integration agent.

II. GCSS ARCHITECTURE AND USES-CASES

A. GCSS layered architecture

High-level layered design for GCSS is presented on Fig. 1. Each level of the system is responsible for the functions and includes its own interface.

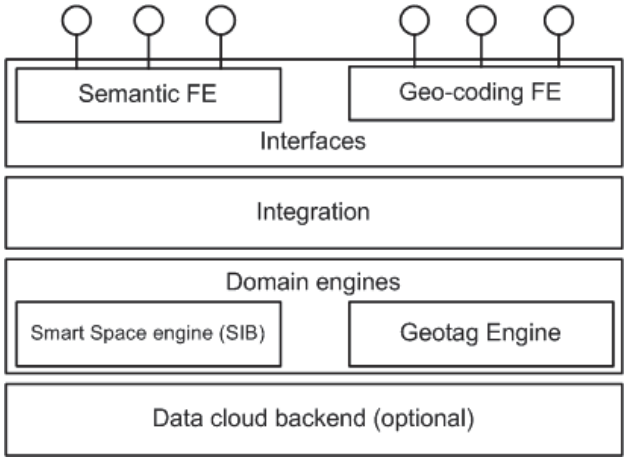

Fig. 1. Layered design of GCSS

There are five base components (levels) [8] that provide basic functioning contour of the system (system life cycle):

- *data acquisition level* – presented by sensors and other receiving information interfaces from the outside world, a person, other systems;

[1] Smart-M3 - http://en.wikipedia.org/wiki/Smart-M3
[2] Geo2Tag - http://www.geo2tag.org

- *data pre-processing level* – data storage and transformation of the primary form to a form suitable for analysis and decision–making;
- *decision-making level* – module responsible for information processing and making decisions to achieve the goals of the system, and support tasks related to self-diagnosis and self-organization;
- *command level* – responsible for making the transformation into control signals own functional components and external systems for the environmental impact implementation;
- *action level* – implementation of information and physical control of external systems, including the task of encoding and transmitting control signals to run-time systems and control command execution.

The main object of the platforms integration is the integration agent or mediator. Its primary task is to provide interaction between Smart-M3 and Geo2Tag platforms and the platforms data conversion into one common format (triplets). Each platform has the necessary programming interface (API).

High-level layered design and Location based engine for GCSS discussed in article [7].

B. GCSS main use-cases

The main task of the agent - the Smart-M3 and Geo2Tag platforms union, it is expanding the space with new data – geo-data.

The main user interaction with agent is to run it and specify the connecting settings for the Smart-M3 and Geo2Tag platforms and also monitoring and control of its operations. The agent then works independently checking receipt of new geo-data, producing a conversion in triplets and publishes them into space.

The main functional use-cases of the integration agent presented at list below.

List the main functional use-cases of the integration platforms agent:

- smart space Smart-M3 platform management:
  - leave / join platform operations;
    query/insert/delete/update information platforms operations;
    subscribe/unsubscribe platform operations;
- Geo2Tag platform management:
    connection/disconnection;
    obtain platform data;
    search and filtration platform data;
- geo-tags conversion mechanism to space data (triples);
- algorithms for searching and filtering of space data, for example, by means of SparQL queries;
- ranking mechanism of space data (the algorithm of selection the latest objects by location, optional);

The first three use-cases are fulfilled the main features of an agent to increase the space with new information, geo-data, that will be used to determine the location and search for objects in space, of which the first two are available by Smart-M3 and Geo2Tag platforms.

Let us consider the two main agent use-cases in more detail - the geo-tag conversion mechanism and algorithms for searching and filtering of space data and one optional use-case - ranking mechanism.

III. PLATFORMS INTEGRATION AGENT

Now the integration agent responsible for the platforms integration and fills the Smart-M3 space with geo-data by conversion mechanism. Next, the agent will combine the functionality of both platforms (Smart-M3 and Geo2Tag) and will become a sort of common platform within the device to control and manage data between all smart space devices [9].

Agent ontology we may create by using special Smart-M3 ontology generator – SmartSlog [10, 11].

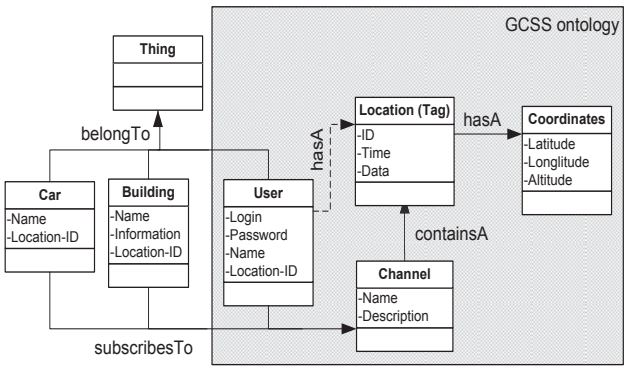

Fig. 2. Overlay ontology used by GCSS

GCSS ontology consists of four classes – User class, Channels class, the Tag itself and its Coordinates. Class User is responsible for a user's of the Geo2Tag platform in space, the tags channel describes a set of tags for a given criterion, the Tag class describes itself data. It should be noted that users can subscribe to an unlimited number of channels, as well as a channel can contain unlimited number of tags. Class User can directly communicate with the tag through the property hasA.

Coordinates are allocated in a separate class for more convenient their representations in an agent ontology.

Each Geo2Tag platform user, if it exists, will be associated with own user in the Smart-M3 space, if not, then will be created a new space user, and it will be automatically attached to the tag location and the channels to which it subscribes. Location may be attracted to any space object after adding new propertie (e.g., Location-ID) in the object class of the space ontology. Class Tag property Data is mainly used for searching and filtering space objects, but it can be also used for its association with the object.

It should be noted that the user location or other space object (not static) can change location with time and in order to remain relevant data necessary to provide handling this situation. Smart-M3 platform provides publish-subscriptions mechanism by subscribing to specific triplets, the object will automatically receive new data after it changing. In our case, these data are the properties of the Coordinates class.

The agent will use the object model of the ontology representation, i.e., have clearly documented ontology classes names and their properties, as well as certain triplets (subscription). Thus, the space agent ontology will look like a list of properties that linked by a predicate. In the first version of the integration agent the space will be filled only with geo-data, which will be linked with their space objects (a person, object, etc.). In the future we plan to expand the space by the addition of the users and channels tags information.

Some details of the work with the agent ontology will be considered in the discussion of the tags conversion mechanism.

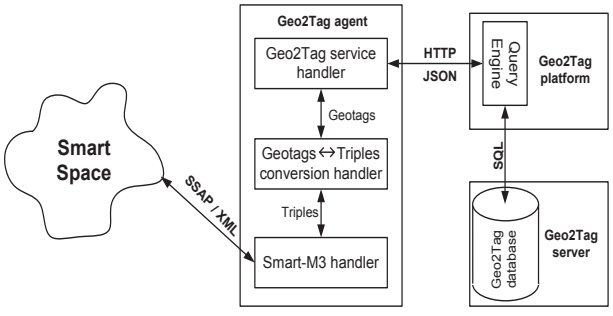

Fig. 3. GCSS architecture

All Geo2Tag platform data are stored in a database on a dedicated server. Geo2Tag platform allow you to record and retrieve data using REST-specific queries in JSON format. There are also a variety number of clients to work with a Geo2Tag platform, mainly for mobile platforms.

Integration with the Smart-M3 platform will be implemented through a special mediator (agent). Its main task is to convert data from one platform format (Geo2Tag, JSON) to another format (Smart-M3, XML). As mentioned above, the Geo2Tag platform transmits data in JSON format, this is a text format, but in a more readable form for humans.

The agent consists of three main components:

- Geo2Tag service handler;
- Geotags – Triples conversion handler;
- Smart-M3 handler.

Geo2Tag service handler is responsible for obtaining geo-data, it connects to the server database and requests data using a special class LoadTagsQuery.

Geotags – Triples conversion handler is required to bring data to a convenient form for the triplets creation. Since the data are returned in JSON format, they need to be parsed and pulling the necessary data, namely, time, location and description of the geo-tag by saving them for later processing.

At the last stage is a connection to the space, then creates triplets according to the ontology which then placed to the Smart-M3 space. What the triplets are created and their number are discussed in "*Geo-tags conversion mechanism*".

After the smart system platform development within the device, each of its mechanism (algorithm) will be subject to thorough analysis by the following criteria: universality, performance, resources, the ability to integrate into embedded devices, memory size, the amount of transmitted traffic, response speed.

The main evaluation criteria's of the Smart System platform will serve – its performance, the ability to integrate into embedded devices, the amount of transmitted traffic and response speed. The analysis should show how the platform behaves in the real conditions and only then take steps to improve its operability.

IV. GCSS USE-CASES

A. Geo-tags conversation mechanism

The conversion mechanism of the LBS Geo2Tag platform data to the Smart-M3 platform data is a main problem for the expansion of the space data.

All Geo2Tag platform geo-data stored in a special database for their retrieval is responsible class *LoadTagsQuery*, which with the help of the request *DataChannels getData()* retrieves all tags data that are stored on the server.

According to the extracted tag structure of the Geo2Tag platform - time, altitude, latitude, longitude,

label, data, user_id, url, channel_id, we can form next triples:

- < User, hasA, Location-ID >
- < Location, hasID, ID >
- < Location, hasTime, Time >
- < Location, hasData, Data >
- < Location, hasLatitude, Latitude >
- < Location, hasLongitude, Longitude >
- < Location, hasAltitude, Altitude >

For a triple creation a Smart-M3 platform interface provides a base class - *Triple* to form a triplet <*subject, predicate, object*> type. Each triplet component is a class *TripleElement*, where the *object* has two types - *URL* and *Literal*. All extracted tags are stored in a list of triplets in the Smart-M3 platform database.

The geo-tags conversion mechanism to the space triplets consists of the following steps:

1) Connect to the Geo2Tag platform by using a Login() query;
2) Point service (database) by setDB() query, where data will be obtained;
3) Sampling nearest tags with a LoadTags() query or Filter(), that queering tags inside the defined geometry figure;
4) Obtaining the necessary tags parameters from received data (JSON format);
5) Formation of the initial triplets for space objects representation by class Triple(S, P, O). A triplet for linking space object with its location, a triplet for location time, coordinates (Latitude, Longitude, Altitude) and data. In general, six triples describe space object location, which presented earlier.
6) Connection and insertion triplets to the space with the help of Smart-M3 API (join() and insert() methods).

After the execution of the algorithm, the space will be filled with latest's tags from the Geo2Tag server database.

Another task is to assign geo-data to space entities (people, things, objects, etc.). For the binding position in the space (coordinates), the tag has a parameter *data*, which describing the location (city, street), thus adding to the ontology person (device, object) an additional property - *location* and associating it with the triplet property *data*, an entity gets a new property - a position in space and time. During the formation of the tag triplet, to the data property is recorded the location-based information. Thus we have the opportunity to make queries by the "*location*" property and find out where is the person, object, device at this time. The agent ontology will be presented in the object-oriented model, where all objects and properties of the ontology are presented in the software objects form (structures, data). The correct connection with any ontology will allow obtaining information about the location of all objects in space and time.

It is worth mentioning that the performance optimization was carried out when developing the Geo2Tag platform, where it was found that the bottleneck of the platform is the interaction with the database (WriteTag() and LoadTag() operations) [12] and requires further optimization.

B. Space data filtration mechanism

The filtering mechanism of space data required to obtain relevant information at the moment of the system work, thereby filtering objects by location, we will have a list of the most relevant data to this time. Consider a filtering data mechanism based on their metadata obtained by SparQL3 queries [13].

Each ontology object has a set of metadata, for example, *Id*, *Description*, *Type*, *Time*, *Position*, *Status* (e.g. *Offline, Online, Connecting*). Object metadata used in the filtering process to retrieve only those objects that satisfy the consumer (client) requirements.

The filtering process may be performed in accordance with certain requirements which may be associated with the following factors:

- the scope of object use, that is, for what purpose can be used this object;
- various QoS parameters, such as response time, execution time, availability, security, reputation, cost and others;
- geographic location of the object.

The space object filtering process is carried out by integrating one or more constraints in a SPARQL query. A constraint, expressed by the keyword FILTER, is a restriction on solutions over the whole group of graph patterns in which the filter appears. SPARQL FILTERs can set up restrictions by implementing arithmetic expressions (i.e. =, !=, >, <, <=, >=, etc).

The examples in Table 1 clarify the use of SPARQL FILTER in the objects filtering process.

TABLE I. SPARQL FILTRATION QUERIES EXAMPLES FOR SPACE OBJECTS

Condition	SparQL query

[3] SPARQL 1.1 Query Language – http://www.w3.org/TR/sparql11-query

Filtering space objects by time and position	SELECT ?human ?time ?position
	WHERE {
	?human *belongsTo* ?object .
	?human *time* ?time .
	?human *position* ?position .
	FILTER regex (?*time* = "18.00")
	FILTER regex (?*position*, "Spb")
	}
	ORDER BY ?human

Filtering only "online" space objects	SELECT ?human ?position ?status
	WHERE {
	?human *belongsTo* ?object .
	?human *position* ?position .
	?human *status* ?status .
	FILTER regex (?*position*, "FRUCT 14")
	FILTER regex (?*status*, "Online")
	}
	ORDER BY ?human

In the first example, the filter in SparQL query used to retrieve all objects where object property *human-time* has a time value of "*18.00*" and *human-position* property is a text value «*Saint-Petersburg*». This query displays all objects which are in the «*Saint-Petersburg*» at "*18.00*" hour's local time. The second example displays only online objects at «*FRUCT14*» conference.

As a filtration mechanism can also be used Rt++-tree algorithm based on search tree R-tree, discussed in [14].

C. Space data ranking mechanism

As an extension of the platform integration agent functionality, it would be nice to have a ranking the filtered queries mechanism which are displayed only the relevant data that satisfies the user requirements. For example, issuing a list of space users, which had been at a specified location within a certain period of time. The main ranking criterion for this systems type is the most relevant data of the space object location at a given time.

Location ranking shall produce a ranked list of space objects that are compatible to the user requirements or system queries. Several research works have addressed ranking mechanism in various fields [13], [15]. However, the issue of location ranking, where the term "Location" matches space objects, devices, equipments, etc., is poorly discussed. Indeed, ranking decisions is based on two key elements, the QoS/performance parameters and the nature of the location needed by the user-task [13].

Location metadata are stored in the integration agent ontology and systematically updated using location-based LBS Geo2Tag platform. For instance, if the location is of type space real objects, i.e. *Device* or *Equipment*, these metadata could be the spatial characteristics of the object location, e.g. time, coordinates and description. Thus, once the ranking metadata item is specified, an algorithm should be defined and used to rank and allocate the available objects locations.

The main factors that determine this ranking of space objects location are Object Location and Availability [15]:

- *Current Location:* Ranking mechanism matching objects profile to find the subset of objects which have recently been at a target location.
- *Availability:* Ranking mechanism prioritizes candidate answerers in such a way so as to optimize the possibility that the necessary object will be found. This task involves factors such as prioritizing objects which are the most likely currently online; who located or have located in the same area of the asked query; etc.

Ranking mechanism is built based on the space data filtering mechanism, i.e. for output timely data, by comparing the objects location, uses filtered space data and a ranking parameter or location rating which may be provided by smart space users.

V. CONCLUSION

This article is a speech about architecture and main use cases of an agent integration platform: The geo-data conversion mechanism to the space data, a space data filtering mechanism based on its context using SparQL queries and ranking mechanism.

The results of the project:

- integration platforms agent prototype;
- geo-tags conversion mechanism;
- filtering mechanism based on the Geo2Tag platform.

The next step in the development of smart systems device platform is the complete platform components integration, common protocols and interfaces for communicating between all devices.

There are still open questions for future development: overall system performance, effective objects monitoring, integration with media objects.

ACKNOWLEDGMENT

The authors would like to thank Finnish Russian University Cooperation in Telecommunication Program for provided feedback and guaranties.

REFERENCES

[1] C. Perera, A. Zaslavsky, P. Christen, D. Georgakopoulos, "*Context Aware Computing for The Internet of Things: A Survey"*, Communications Surveys Tutorials, IEEE, 1–44, 2013.

[2] N. Nabian, C. Ratti, A. Biderman, G. Grise, "*MIT GEOblog: A platform for digital annotation of space for collective community based digital story telling",* 3rd IEEE International Conference on

Digital Ecosystems and Technologies, Piscataway, N.J.: IEEE: 353-358 (2009)

[3] J. Rishede, T. Man, L. Yiu, "*Effective Caching of Shortest Paths for Location-Based Services*", SIGMOD '12, Scottsdale, Arizona, USA (2012)

[4] K. Kolomvatsos, V. Papataxiarhis, V. Tsetsos, "*Semantic Location Based Services for Smart Spaces*", 2nd International Conference on Metadata and Semantics Research (MTSR), Corfu, Greece (2007)

[5] J. Honkola, H. Laine, R. Brown, I. Oliver, "*Cross-Domain Interoperability: a Case Study"*, Nokia Research Center, Helsinki, Finland (2009)

[6] I. Bezyazychnyy, K. Krinkin, M. Zaslavskiy, S. Balandin, Y. Koucheravy, "*Geo2Tag Implementation for MAEMO*", 7th Conference of Open Innovations Framework Program FRUCT, Saint-Petersburg, Russia (2010)

[7] K. Krinkin, K. Yudenok, "*Geo-coding in Smart Environments: Integration Principles of Smart-M3 and Geo2Tag", In Proceedings of the 13th International Conference, NEW2AN 2013 and 6th Conference, ruSMART 2013*, St. Petersburg, Russia, August 28-30, 2013. Proceedings. Springer 2013 Lecture Notes in Computer Science, p. 107-116. ISBN 978-3-642-40315-6

[8] G. Akhras, "*Smart Materials and Smart Systems for the future"*, Canadian Military. Journal 2000. pp. 25-31, 2000.

[9] D. Korzun, I. Galov, A. Kashevnik, N. Shilov, K. Krinkin, Y. Korolev, "*Integration of Smart-M3 Applications: Blogging in Smart Conference",* Proc. 4th Conf. Smart Spaces (ruSMART 2011), Saint-Petersburg, Russia, 22-23 August 2011, pp. 51-62.

[10] D. Korzun, A. Lomov, P. Vanag, J. Honkola, S. Balandin, "*Generating Modest High-Level Ontology Libraries for Smart-M3",* Proc. 4th Int'l Conf. Mobile Ubiquitous Computing, Systems, Services and Technologies (UBICOMM 2010). N. 103–109.

[11] D. Korzun, A. Lomov, P. Vanag, S. Balandin, J. Honkola, "*Multilingual ontology library generator for Smart-M3 information sharing platform",* International journal on Advances of Intelegent System 4 (3&4), 68-81.

[12] M. Zaslavsky, K. Krinkin, *"Geo2tag Performance Evaluation"*, Proceedings of the 12th Conference of Open Innovations Association FRUCT and Seminar on e-Travel, Oulu, Finland (2012)

[13] E. Nageba, P. Rubel, J. Fayn, "*Semantic agent system for automatic mobilization of distributed and heterogeneous resources,* In *Proceedings of the 3rd International Conference on Web Intelligence, Mining and Semantics* (WIMS '13). ACM, New York, NY, USA, , Article 28 , 9 pages.

[14] P. Th. Eugster, B. Garbinato, A. Holzer, "*Location-based Publish/Subscribe",* In *Proceedings of the Fourth IEEE International Symposium on Network Computing and Applications* (NCA '05). IEEE Computer Society, Washington, DC, USA, 279-282. DOI=10.1109/NCA.2005.29 http://dx.doi.org/10.1109/NCA.2

[15] Y. Liu, T. Alexandrova, T. Nakajima, "*Using stranger as sensors: temporal and geo-sensitive question answering via social media",* In *Proceedings of the 22nd international conference on World Wide Web* (WWW '13). International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 803-814.