Geo-Coding and Smart Space Platforms Integration Agent Performance Testing and Analysis

Kirill Yudenok

Saint-Petersburg State Electrotechnical University "LETI",
Saint-Petersburg, Russia
kirill.yudenok@gmail.com

Abstract. Internet of Things, Smart Spaces and Geo-coding technologies are fastest growing directions in modern mobile market and urban environments [1, 2]. This is due to the advent of various services that using common technologies, as well as to develop common requirements and architectures for using geo-contextual services in semantic data processing. Location is a mandatory requirement for the Internet of Things and Smart Spaces directions products, because geo-context is a one of the factor to determine the location of subjects in various environments. As a result, it was decided to integrate geo-coding and smart spaces platforms, for the possibility of using geo-context in the semantic space. As an implementation used two open source software platforms – Geo2Tag and Smart-M3. The article discusses an integration agent performance testing and its analysis, provided recommendations for integration mechanisms optimization.

Keywords: Geo-coding, Smart Spaces, Internet of Things, Smart-M3, Geo2Tag, LBS.

1 Introduction

Geo-coding[1] and Smart Space [3] directions are gaining momentum every day. The appearance of various services that adapt to constantly changing conditions, services, that allowing to markup various objects of the world (virtual or real objects). This is the actual day services, that enjoyed worldwide.

Geo-coding systems markup real or virtual objects by adding the geographical coordinates and time. In turn, smart space provides access to distributed semantic information and communication field for software services, which is being run on various type of devices (personal, autonomous computers, robots etc.). These two directions are mutually different from each other, but together complement each other and adds a new features to the overall system such as, pro-activeness, context awareness, machine-to-machine interactions, platforms possibilities [4]. By combining these platforms will be reached the ability to define and discover objects in a described semantic space. Geo-coding capabilities will not only markup the existing objects, but

[1] Geo-coding – http://en.wikipedia.org/wiki/Geocoding

S. Balandin et al. (Eds.): NEW2AN/ruSMART 2014, LNCS 8638, pp. 62–69, 2014.

also expand the space with new data, handle the situations with the new objects, as well as track their location in space and time.

Geo-context is not replaceable component for the Internet of Things and Smart Spaces directions. In our case, access to the geo-context is obtained by the Geo2Tag [5] platform, this is a geographical context marked up on a virtual world map. Access to the geo-context may also be obtained by using positioning technologies or special sensors.

Geo-context use cases in the Internet of Things and Smart Spaces directions, for example, assistance to find a parking space, monitoring energy in the city, assistance in finding electric car charger stations, notification of bus arrival, assistance after a car crash, notification of car traffic jam, location based dating, location based marketing etc [2, 3].

In this article we will talk about testing and performance analysis of the basic mechanisms for the geo-coding and smart space platforms integration agent. The paper is structured as follows: Chapter 2 briefly discusses integration platforms and their main features, in Chapter 3 describes the performance testing methodology of the platforms integration agent, Chapter 4 discusses the performance testing details, in Chapter 5 shows the performance testing results, its brief analysis and provides the optimization recommendations for the basic integration mechanisms, Chapter 6 summarizes the work.

2 Integration Platforms – Geo2Tag and Smart-M3

To support the geo-coding possibilities in the smart space as a geo-coding platform serves a Geo2Tag[2] system, which implement the basic functionality for working with geo-data. Its main features – users and data channels management, basic operations with geo-data (load tags, write tags etc.), multiple geo-data filtering mechanisms (spatial and temporal filtration). Geo2Tag platform includes a full server that handles and stores all geo-data. Platform is implemented using REST API technology, all logic is written on high-level programming language – C++/Qt, Java. Granted API allow to develop services for a variety desktop and mobile platforms (Windows, Linux, Android, J2ME, Web).

Interaction with the smart space provides Smart-M3[3] platform [6, 7, 8]. Its main task is to provide the infrastructure for the exchange of semantic information between different entities (software or hardware). The platform provides a distributed data storage in a special semantic information broker (SIB) and it is processing by means of developed agents – knowledge processors (KP). Programming interfaces allow to develop KP in the following languages – C, C++/Qt, C#, Python, Java, PHP, Javascript.

Common requirements, integration agent architecture and its detailed description has been presented in [9]. The main functional requirements are mechanisms for

[2] Geo2Tag – `http://geo2tag.org`
[3] Smart-M3 – `http://en.wikipedia.org/wiki/Smart-M3`

converting data from one platform format to another, spatial and temporal filtration. The main non-functional requirements are high-performance solution for handling large amounts of data (cloud based massive offline processing and local context indexing/caching) and compatibility with Geo2Tag and Smart-M3 platforms interfaces (i.e SSAP or REST).

As a result of the platforms integration have been developed a special agent which main tasks are:

1. providing an interface to the semantic and geo-data;
2. distributed storage for semantic and geographical information;
3. interface for the association of semantic objects with geo-data;
4. spatial and temporal filtration (Smart-M3 and Geo2Tag).

Geo-space conceptual model with additional knowledge processors (Cloud-backend, Big Data, context management, sensors) is shown at Fig. 1.

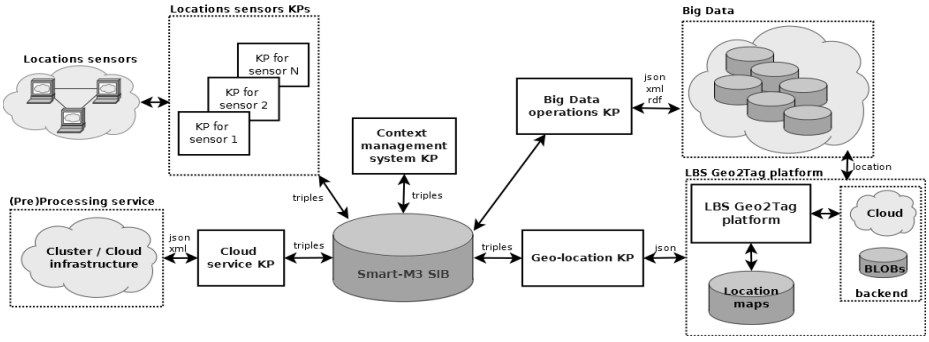

Fig. 1. Geo-space conceptual model

3 Performance Testing Methodology of the Platforms Integration Agent

Geo2Tag and Smart-M3 platforms integration agent was tested on a dedicated virtual machine with installed Smart-M3 platform and below listed characteristics, access to the Geo2Tag platform is performed over the Internet using a HTTP/REST protocol:

- CPU – Intel i7, 3.4 Mhz, 4 cores;
- RAM – 8 Gb;
- OS – Ubuntu 14.04 LTS;
- Geo2Tag – 0.31 version (Qt API 4.8);
- Smart-M3 – 0.9.01 (redland-1.0.16-unibo (Virtuoso[4]), redsibd-0.9.01_time, sib-tcp 0.81, Libwhiteboard Qt API[5]).

[4] Virtuoso – `http://virtuoso.openlinksw.com/`

[5] Libwhiteboard Qt API – `http://sourceforge.net/projects/smart-m3/files/Smart-M3_v0.9.5-beta/`

The testing object is a geo-coding and smart space platforms integration agent and its basic data processing mechanisms. Testing was conducted of two types – functional testing of an integration agent mechanisms and integration agent load and stress performance testing.

For functional testing were developed unit tests that verify the basic mechanisms of the platforms integration such as platforms data conversion mechanisms, data filtering techniques and the basic mechanisms of the geo-coding and smart space platforms, such as login and load data from the Geo2Tag system, query data from the Smart-M3 platform and others.

For performance testing have been prepared tests (scripts) that test system under a certain load. As a system load acts the different scenarios of the system, as a permanent data conversion from one format to another and vice versa, repetitive data filtering methods, query or insert triples. Stress testing was performed for the main agent repetitive mechanisms - conversion, filtering and data querying.

The main integration agent performance metrics are:

1. query (operation) execution time;
2. the number of operations performed in 1 second;
3. the amount of consumed CPU and memory.

4 Platforms Integration Agent Performance Testing

Performance testing was carried out using a specially designed tool for the Geo2Tag platform called *Profiler* [10], whose tasks are:

1. definition and implementation of load tests for any program operation;
2. creation a separate thread for each test with the counting system performance metrics:
   - query (operation) execution time;
   - the number of operations performed in one second.

Listing 1 shows a performance test script example of the filtration tags throw the Smart-M3 platform interface:

```
#!/bin/bash
result_dir="./results_r_`date +'%d_%m_%Y_%H_%M_%S'`/";
if [[ "$#" == "2" ]]
then
  steps_count=$1;
  read_requests_count=$2;
else
  steps_count=100;        # number of iterations
  read_requests_count=500;# number of queries (for Geo2Tag operations)
fi
mkdir "$result_dir"  # results directory
```

```
for (( i=0;  i<=steps_count; i++ )) ;
do
 echo "$i iteration"
 cool_num=`printf "%08d" $i`;
 ./profiler $read_requests_count g2tFilter 2>"$result_dir/$cool_num"
done
```

As a result for each integration mechanism was create a separate test, test selection is performed by a passing name of the operation as a *Profiler* testing tool parameter. For each test was created a separate script that deals with an environment setup and required parameters for the testing tool.

Each test evaluates two system performance metrics – the execution time of the operation (query) and the number of operations performed in one second. Operation execution time metric measures the operation runtime since the beginning of the function execution using the Qt API *msecsTo()* function, the number of operations performed in one second metric counts the number of operations performed during one second of time, based on its runtime according to the formula:

$$number\_of\_operations\_in\_1\_second = 1000 / operation\_time\_in\_milliseconds \quad (1)$$

Each test can be performed a number of times, depending on the *steps_count* parameter of the script passed to the test main loop. In our case, tests are performed hundred times to collect the necessary statistics. After the test, all characteristics recorded to a file with the number of iteration.

As a result of testing were obtained the necessary statistical data that allowing to say how much time was spent on the operation and the approximate number of operations performed in a one second. In order to ensure that the statistics is true, for each operation statistical data calculated the necessary characteristics – the mathematical expectation, variance and standard deviation, which allow to understand the spread of statistical data and their deviation. Also, all the statistical data verified by the "*three sigma*"[6] rule, confirming that all the random variables are normally distributed.

5 Platforms Integration Agent Performance Testing Analysis

The main platforms integration agent performance tests are:

1. loading geo-data from the Geo2Tag platform (basic filter by radius);
2. triples filtering through the Geo2Tag platform;
3. geo-data filtering through the Smart-M3 platform;
4. convertion tags to triples and vice versa;
5. insert data to the Smart-M3 platform;
6. query data from the Smart-M3 platform.

[6] Three sigma rule – `http://en.wikipedia.org/wiki/Standard_deviation`

Table 1. Integration agent performance testing summary results

Test case	Mean value (ms)	Standard deviation (ms)
Load tags from the Geo2Tag platform by radius	538	79.59
Triples filtering through the Geo2Tag platform	133	31.17
Geo-data filtering through the Smart-M3 platform	1621	424.97
Conversion 1000 triples to the tags	31.01	32.72
Conversion 1000 tags to the triples	27.24	15.73
Insert triples to the Smart-M3 platform	3015.4	173.16
Query triples from the Smart-M3 platform	1302.19	90.16

Summary results of the obtained agent integration performance testing characteristics are presented in Table 1.

From the the summary table of obtained characteristics seen that some integration mechanisms takes a long processing time intervals, among them:

1. geo-data filtering through the Smart-M3 platform – 1-2 seconds;
2. insert triples to the Smart-M3 platform – 3-4 seconds;
3. query triples from the Smart-M3 platform – 1-1.5 seconds.

As a result, we analyzed the execution time of the main data integration mechanisms. As an function calls analysis used tool – *callgrind*[7], which is part of the profiler – *valgrind*[8] tool and *kcachegrind*[9] tool.

As a result, it was revealed two types of major problems:

1. Multithreaded data processing in the Smart-M3 platform components.
2. Processing and parsing of obtained results for the basic Smart-M3 operations – insert, update, query (Libwhiteboard Qt API).

In the first case the profiling showed that the Smart-M3 insert and query tests incorrectly handle threads. As a result, the platform integration agent and Smart-M3 platform were subjected to analysis using the Intel Threads Profile, which is a part of Intel Inspector XE 2013[10] tool. Where it was found that the *redsib daemon* and *sib-tcp* Smart-M3 platform components have errors while working in multi-threaded mode when performing basic operations. Smart-M3 *whiteboard daemon* component and platforms integration agent (GCSS) are single-threaded.

The second type of the problem associated with the processing of the basic Smart-M3 operations using libwhiteboard Qt API. At first, each Smart-M3 operation (insert, update, query) generates string results output which is converted into XML-tree

[7] Callgrind – `http://valgrind.org/docs/manual/cl-manual.html`
[8] Valgrind – `http://valgrind.org/`
[9] Kcachegrind – `http://kcachegrind.sourceforge.net`
[10] Intel Inspector XE 2013 toolset – `https://software.intel.com/en-us/intel-inspector-xe`

results of the operation. Thus, the XML-tree composition for a large number of triples and their periodical query lead to loss of performance while performing basic operations. The solution to this problem is to use special data structures or switch to a binary data transfer protocol, such as KSP [11].

Also have been fixed error in the tags–triples conversion mechanism while removing duplicates triples, operation time reached ~ 50 ms. It should be noted that the usage of the Smart-M3 platform together with Virtuoso significantly increase performance in the basic triples processing operations. Performance boost when performing basic operations reached approximately 50%, but the libwhiteboard Qt API insert and query operations are fairly slower than in Python API.

According to the above analysis we may suggest the following recommendations for the optimization integration mechanisms:

1. multithreading errors correction for the Smart-M3 platform components – *redsibd*, *sib-tcp*;
2. replacing the current Smart-M3 platform SSAP protocol to the binary protocol, for example, KSP;
3. use Smart-M3 platform with Virtuoso (increase productivity of Smart-M3 platform operations ~ 50%);
4. use Smart-M3 Python API or optimization of the basic platform operations (insert, update, query) for the Libwhiteboard Qt API;
5. use SparQL[11] queries instead of the usual query (Libwhiteboard Qt API does not support SparQL queries);

6 Conclusion

This article presented the performance testing results and its analysis of the basic geo-coding and smart spaces platforms integration mechanisms. The testing revealed that some of the agent integration mechanisms are need to be improved, and some one depends on the platform, protocol and API. Profiling showed that the integration agent has the following problems – Smart-M3 platform miltithreading problem, unsuitable protocol for data exchange, outdated Qt API. The first two problems are mandatory, because they determine the overall platform performance. Recommendations for the mechanisms optimization will help to increase the agent performance.

Still open questions for further development – performance of the whole system (multithreading errors correction, binary data exchange protocol for the Smart-M3 platform, optimization of basic operations in Qt API), cloud computing.

References

[1] Vermesan, O., Friess, P.: Internet of Things: Converging Technologies for Smart Environments and Integrated Ecosystems, Aalborg. River publishers series in communications (2013) ISBN: 978-87-92982-73-5

[11] SparQL – `http://en.wikipedia.org/wiki/SPARQL`

[2] van der Zee, E., Scholten, H.: Application of geographical concepts and spatial technology to the Internet of Things. Vrije University, Faculty of Economics and Business Administration, Amsterdam, Research Memorandum 33 (2013)
[3] D2.2 Requirements, Specifications and Localization and Context-acquisition for IoT Context-Aware Networks. uBiquitous, secUre inTernet-of-things with Location and contExt-awaReness (BUTLER), Jacobs University Bremen gGmbH (JUB), Project number 287901 (October 2012)
[4] Balandin, S., Waris, H.: Key properties in the development of smart spaces. In: Stephanidis, C. (ed.) Universal Access in HCI 2009, Part II. LNCS, vol. 5615, pp. 3–12. Springer, Heidelberg (2009)
[5] Perera, C., Zaslavsky, A., Christen, P., Georgakopoulos, D.: Context Aware Computing for The Internet of Things: A Survey. IEEE Communications Surveys and Tutorials PP(99), 1–44 (2013)
[6] Bezyazychnyy, I., Krinkin, K., Zaslavskiy, M., Balandin, S., Koucheravy, Y.: Geo2Tag Implementation for MAEMO. In: 7th Conference of Open Innovations Framework Program FRUCT, Saint-Petersburg, Russia (2010)
[7] Honkola, J., Laine, H., Brown, R., Tyrkkö, O.: Smart-M3 Information Sharing Platform. In: 1st Workshop on Semantic Interoperability in Smart Spaces (2010)
[8] Honkola, J., Laine, H., Brown, R., Oliver, I.: Cross-Domain Interoperability: A Case Study. Nokia Research Center, Helsinki (2009)
[9] Korzun, D., Balandin, S., Luukkala, V., Liuha, P., Gurtov, A.: Overview of Smart-M3 Principles for Application Development, AIS (2011)
[10] Krinkin, K., Yudenok, K.: Geo-coding in Smart Environments: Integration Principles of Smart-M3 and Geo2Tag. In: Balandin, S., Andreev, S., Koucheryavy, Y. (eds.) NEW2AN 2013 and ruSMART 2013. LNCS, vol. 8121, pp. 107–116. Springer, Heidelberg (2013)
[11] Zaslavsky, M., Krinkin, K.: Geo2tag Performance Evaluation. In: Proceedings of the 12th Conference of Open Innovations Association FRUCT and Seminar on e-Travel, Oulu, Finland (2012)
[12] Kiljander, J., Morandi, F., Soinen, J.-P.: Knowledge Sharing Protocol for Smart Spaces. International Journal of Advanced Computer Science and Applications 3(9) (2012)