🏆 *reviewed paper*

Semantic Platform for Building Coherent Net of Smart Services

Alexander Vodyaho, Nataly Zhukova, Maxim Kolchin, Maxim Lapaev

(Prof Alexander Vodyaho, St. Petersburg Electrotechnical University (LETI), 5, Popova str., St. Petersburg, 197376, Russia, aivodyaho@mail.ru aivodyaho@mail.ru)
(Nataly Zhukova, St Petersburg National Research University of Information Technologies, Mechanics and Optics (ITMO), 49, Kronverksky Pr.,St. Petersburg, 197101, Russia, nazhukova@mail.ru)
(Maxim Kolchin, St Petersburg National Research University of Information Technologies, Mechanics and Optics (ITMO), 49, Kronverksky Pr.,St. Petersburg, 197101, Russia, kolchinmax@gmail.com)
(Maxim Lapaev, St Petersburg National Research University of Information Technologies, Mechanics and Optics (ITMO), 49, Kronverksky Pr.,St. Petersburg, 197101, Russia, m.lapaev@corp.ifmo.ru)

1 ABSTRACT

Information infrastrucrure of modern cities has been developing very intensivly and nowadays a lot of new services appear and a problem of their integration has become evident. Existing integration solutions used both in frames of enterprise information system and Internet-oriented solutions are not efficient enough when scaled to city infrastructure. In this paper an integration platform based on semantic technologies is proposed. The services and their peculiar features are described in ontology. Now semantic platforms are effectively used in a number of applied domains for solving a wide range of problems, for example, support of medical and administrative processes in medical centers. Semantic smart services networkes are shown to be an efficient intelligent way of building a data processing chain for diverse input state of structure and varied desired output format or stage in a flexible way. We present one of the examples of using smart services for doctor routine tasks to present the temporal data in a convenient inlelligent way.

2 INTRODUCTION

Information infrastrucrure of modern cities has been developping incredibly fast over last decades. Improvement of all kinds of services has been impatiently demanded by end users in all domains who where forced to keep up-to-date not to lose ground in their spheres of interest. As a result, the majority of services used now are high-quality services that meet the extended requirements of end users. They can be definitely called smart services. The proplem is that there are a lot of services, but they are not compatible with each other. They can hardly be considered as elements of complicated business processes. It leads to creation of new services with duplicating functionality. Observed dynamics of service market development and its short term prediction clearly shows that in near future it will be impossible to satisfy all requests for new services and service infrastructure will become overheated. At the level of enterprises the problem is commonly solved by means of enterprise service bus, at the level of WWW – due to building and overall application of semantic web services. For the level of cities still there are no solutions that allow building complex logical structures based on existing services. The most obvious way for services integration is their unification. Even this simple solution is unimplementable for two reasons. First, it requires huge resources that depend on the total number of services. Second, it can affect the functionality of the services that is inadmissible for end users. So, one can say that at the level of the city integration solutions based on enterprise service bus are too light but Internet oriented solutions such as semantic web servises are too heavy.

In this paper we propose a platform for agile service integration that allows linking services using semantic technologies. The platform does not generate additional requirements to services or imposes any restrictions. It supports linking services and, thus, building a net of services. Furthermore, it can reveal possible links between services that can enrich the service infrastructure. Sematic technologies form the base for integration platform. The services and their peculiar features are described in the platform ontology using OWL language. The OWL description of the services clarifies reasonable cases and ways for services usage. Similar approach is used for describing logic of complex services application. The processes of services interaction are defined in ontologies as well. For logic description BPEL is used.

3 BUISNESS PROCESSES, ONTOLOGIES AND KNOWLEDGE GRAPHS

Semantic service integration, state of the art: R&D in the domain of automatic generation of business processes is being realized for at least 15 years. By the most part web services that are used in the process have semantic description. Such approach is called semantic web services (SWS) approach. Essential results have been achieved in this sphere [1, 2]. OWL-S standard was proposed [3]. Certain results have been received in the domain of semantic grid services [4]. Detailed analysis of the problem of semantic integration of web services one can find in [1]. Described approaches can be classified as Internet-oriented or heavy-wheight solutions. Nowadays it becomes more and more evident that in the nearest future general solution of the problem of ‘on the fly’ generation of business processes on the base of semantic solutions will be strongly claimed.

For solving the problem of autumatic generation of BP on the base of existing services for city information systems more simple solutions can be used. They can be classified as light weight SWS (LW SWS).

Idea of suggested approach: The main idea of suggested approach is usage of ontology based knowledge graph (RG) which describes all possible business processes. Needed business processes are tailored from the KG. So we have semidistributed system. To KB of buisness servises an agile integration platform is required. The platform must meet the following requirenments:

i) the platform must be able to generate business processes on the fly or manually,

ii) a user can see generateg BP and can correct it,

iii) the platform must have agile features, i.e. if context is changed the selected path in KG must be corrected,

iv) the corrections will define changes in the corresponding BP,

iv) the platform must provide possibility to change BP structure inaccordance with results of executed operations,

v) the platform must be easy to use, a user can form BP by defining its goal.

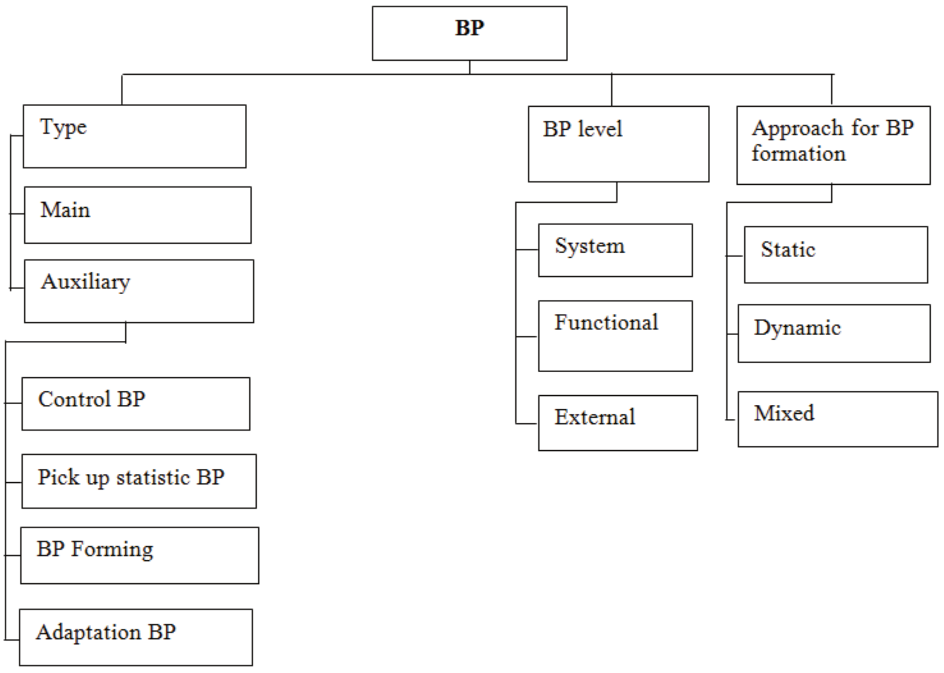


Fig. 1. Approaches to BP implementation

4 AGILE BUSINESS PROCESSES AND SEMANTIC SERVICES

In order to describe suggested approach to building BP on the base of knowledge graphs the following main terms are used: technology, pattern, service and business process (BP).

Technology defines a sequence of steps to be done to solve the problem. Steps are described in general, they may be described even in business terms. Technology can also be defined as not ordered or partially ordered set of method, algorithms, patterns, libraries, services or scripts and a set of business rules and restrictions concerned to business rules. Technology is built on the base of ontological models.

Pattern (business process pattern or abstract business process) is a description of steps to be done to solve the problem described in terms of concrete technology. One can build a pattern in statics in run time mode. It is described by the oriented graph. Pattern is associated with the target. A target-oriented pattern can be conceded as a part of a technology. Pattern is to be used for creation of an executable model of buisness process, i. e. UML model. Then using proper tool it is possible to receive executable code in any form (application, library, service, script, bpel, set of business rules etc).

Business processes: BP (Fig. 1) is a process of invokation of BP of lower level in defined order. BP can be invoked in sequence, parallel or mixed mode. BP can be planned in static or in dynamic mode. Static BP as a rule is realized with the help of bpel or other script language. Dynamic BP can be realized by means of business rulus or sematic web services. Agile BP can be defined as BP which can be adapted to context. Different mechanisms can be used for adaptivity realization. The main one is formation of BP in run time.

Service view point on buisness processes (SVP). SVP can be concided as one of key archtectural view points on buisness processes. In SVP BP are described as triples: **SVP ={S, BP, M},** where S is a set of services, **BP** is a set of **BP** and **M** is a set of BP control stratagies. BP can ivoke services. It also can be ivoked as a service. A service can be realized as BP.

As a rule servises are correlated with levels such as infrastructure servises, system services, busines services and external services.

Infrastructure servises are a low level services, for example, naming service, life cycle services. They are used as elements for building higher level services. System services are composite service realized as business procecces. Their opration is described in terms of subject domain.

Business servises are high level services, their opration is described in business terms. Business services are formed as a composition of system services.

External services are high level services which are to be used for realization of B2B interaction. As a rule they are also business services.

Types of services. From the point of view of tasks to be solved by a service, services can be classified in the following way: functional services, data, information, knowledge (DIK) services, interpreters (engines) and auxiliary services.

Functional services realize procedures of DIK processing. The main types of functional service are transformation services, harmonization services and integration and fusion services.

DIK servises can be divided into 3 groups: services that realize get-set functions for DIK, services for DIK search and mining (data mining, process mining).

Interpreters (engines) are used as scripts interpreters. It may be bpel-engine, business rules engine, sws-engine, etc.

Auxiliary services are used for realization such infrastructure functions as life cycle support, security, etc.

BP management strategies. One can use following strategy: directive strategy, data flow driven strategy, demand driven strategy and resource driven strategy.

Directive strategy – the order of services invocation is defined on the stage of BP forming. This strategy can be realized with the help of i.e. bpel.

Data flow driven strategy – the single precondition of the strategy is that input data for service invoke is available. Implementation of this approach needs special script languages.

Demand driven strategy - the precondition for service invocation is request for results of service operation. For the strategy implementation as a rule 2 passes are used. First step is creation of the graph of requests (BP graph); on the second pass the BP is realized. This approach is used for BP formation in dynamic mode. This 2 passes (stages) can be overlapped.

Resource driven strategy: as a rule is used as an auxiliary one when there is not enough resources to realize all required tasks in parallel.

5 AGILE INTEGRATION PLATFORM

Agile software development approach supposes a series of interactive methods and techniques based on dynamically changing technical requirements to provide their implementation as a result of intercommunication and contribution of diversely qualified engineers and specialists. We propose agile approach for the platform (Fig. 2) for the reason that all kinds of personell is desirable provided that the framework is a multi-layer system at the intersection point of a number of domains causing an unability to obtain developers of same qualification within all of the fields.

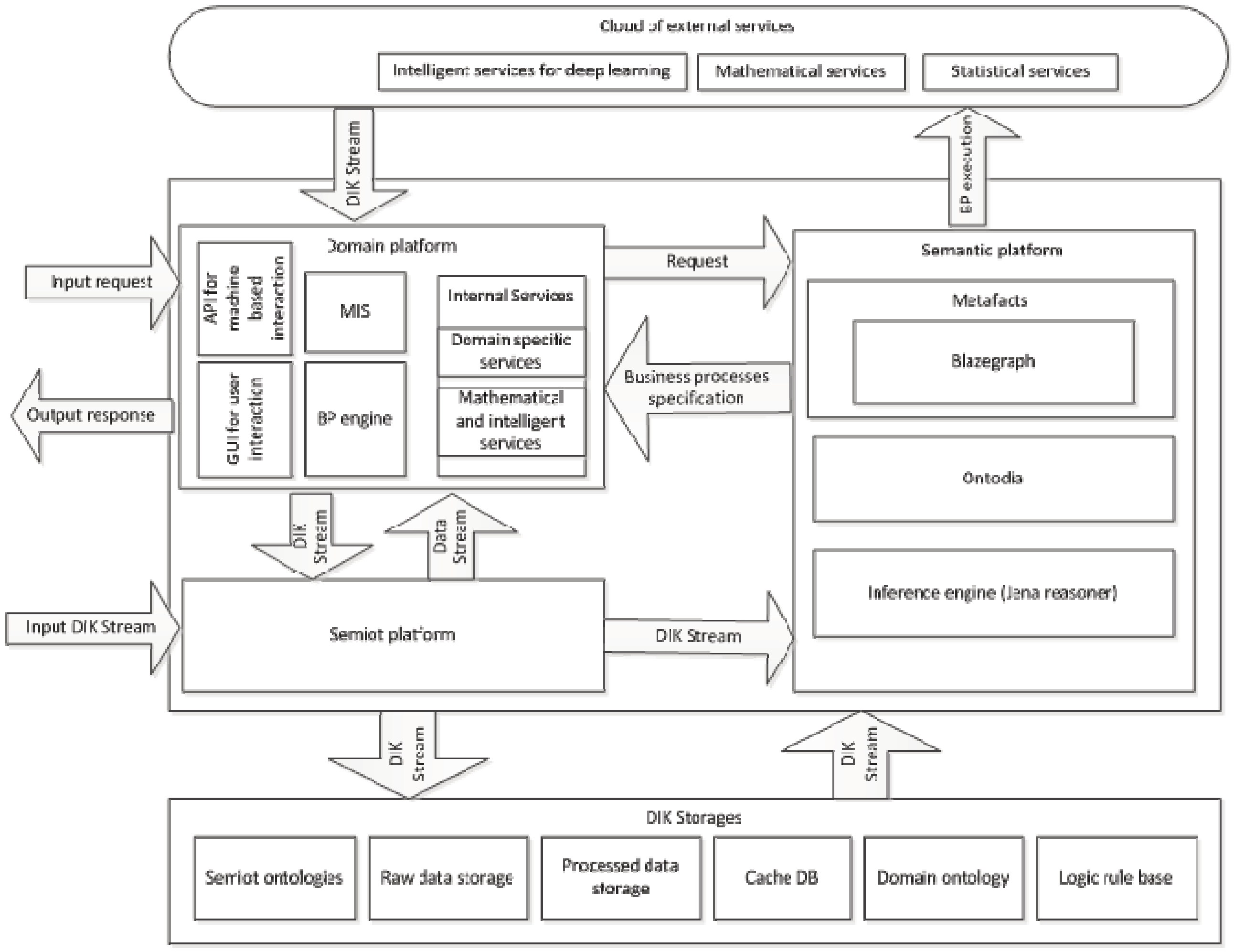


Fig. 2. Structure, breif description of the components and logic of their interaction

The platform is constituted of a number of layers including a number of components grouped by tasks: storage, internal processing and external processing. The central part of the platform is internal domain-oriented semantic processing of data including domain area (input and output of domain-oriented information), processing (based on stream of incoming data, information and knowledge, DIK) and semantic blocks (storage of knowledge and tools to process and display knowledge and produce new knowledge). We give more information on each block below:

Semantic platform as a component of knowledge storage base and queries, production-rule-based inference and knowledge base display and analysis:

(1) BlazeGraph as a scalable and flexible triple store and SPARQL-endpoint for search and inference and a high-performance graph database platform with support of RDF for knowledge data-set upload with a support of up to 50 billion edges;

(2) MetaPhacts as a product providing powerful solutions and numerous services for describing, querying and interchanging graph-based data, as well as a user-oriented open platform to visualize and interact with knowledge graphs;

(3) OntoDia as a simple and free of charge online OWL and RDF diagramming tool to represent a knowledge base with no programming involved;

(4) Apache Jena as a free and open source framework for Java to build Semantic Web and Linked Data applications based on onologies and elements of enference (needs to be supported with powerful inference engines based on pruduction logic such as Drools).

Semiot platform is a platform for gathering data from sensors and web services and providing data for semantic processing and analyses.

Domain platform including a wide range of tools from initial data display tools (information systems, IS) to data processing (domain-specific and internal intelligent services) to provide a processing pipeline from input as an API request to user interface as a result of all processing stages requested by user.

All of the raw, preprocessed and semantically processed data are stored in a lower layer in a data storage of four data state levels: unstructured data, preprocessed semi-structured data, data covered by ontologies and a rule base. These are unstructured raw data storages, semi-structured DB Cache and processed data storages and ontology storages of SemIoT and domain area.

The other part of the platform is constited of a combination of externall services including intelligent data processing services (mathematics and statistics), deep learning services and other auxiliary services and tools distributed all over the wide web to provide a powerful way for data processing.

Main data flow assumes the following processing chain:

(1) input of raw data into SemIoT platform;

(2) processing by services providing us with data for convertion to semantic data;

(3) request to BlazeGraph accompanied with visual representation of the process;

6 SEMIOT PLATFORM

The architecture of the platform was developed to meet two main high-level requirements:

- support of different communication protocols and data models for collecting data from connected devices, and ease extension for a new protocol and model;
- access to the data of the devices through a unified interface, so the data could be accessed and queried regardless of the data model supported by a particular devices.

The architecture of the Platform is presented on Fig. 3. It consists of 5 modules, 2 database and device drivers.

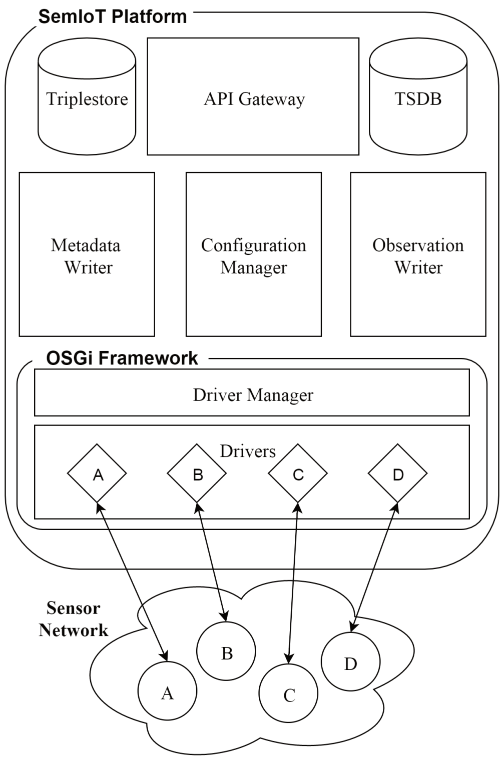


Fig. 3. An architecture of the SemIoT Platform

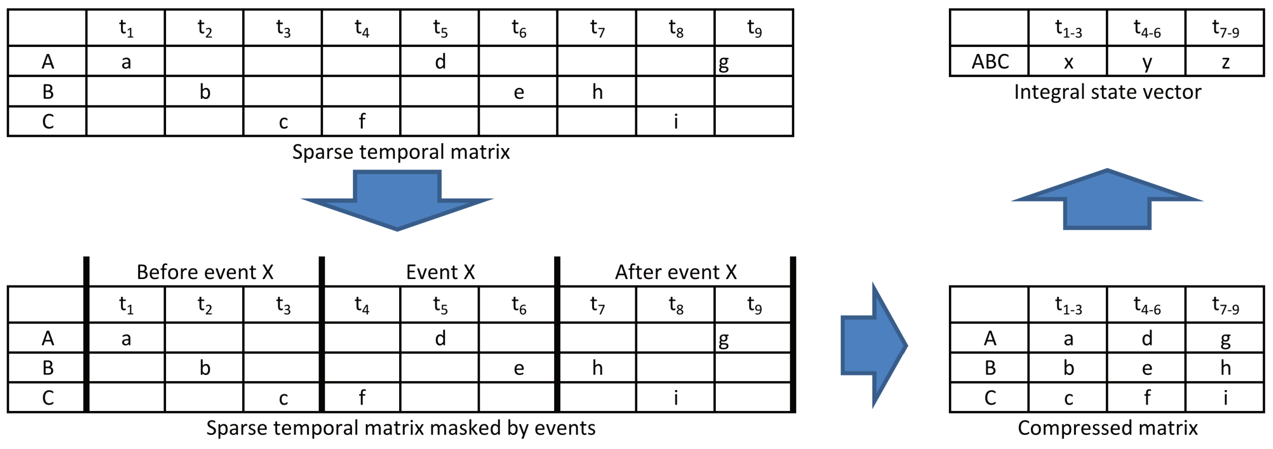
- Device drivers are OSGi services implementing an algorithm of data collection from connected devices over some communication protocol and data model. E.g. HTTP API and JSON-based model. The drivers are installed, uninstalled and configured by Configuration Manager.
- Driver Manager is an OSGi service responsible for collecting the data from device drivers and passing into Metadata Writer and Observation Writer.
- Metadata Writer is a module that manages metadata of the devices, such as structure, measuring capabilities, location and etc. The metadata is stored in Triplestore.
- Triplestore is an RDF database which provides a SPARQL endpoint and implements the GeoSPARQL extension for querying devices by their location. In the current implementation we use Apache Fuseki.
- Observation Writer is a module responsible for managing observations of the devices. They are stored in TSBD.
- TSBD is a time series database optimised for storing time-based data such as sensor observations. In the current implementation we use OpenTSDB (http://opentsdb.net).
- API Gateway is a REST-like programming interface for accessing data of the devices and accessing Configuration Manager.

7 CASE STUDY

Coherent net of smart services is now build and used in medical domain. One of the significant examples is Botkinsky sheet - a temporal matrix of patient's state indicators and manipulations over patient during the curation process: medicine prescriptions, operations and other influencing events. The matrix is named after Russian physician contributed a lot to medicine. Botkinsky sheet is considered to show the advantages of the semantic platform in multi-service intelligent data processing tasks.

Temporal matrix entity is a way to represent various temporal patient's characteristics (body temperature, blood pressure, great number of analysis results) measured within a time interval in a set of coordinate planes consisting of x axis as a unified temporal axis and a number of y axis as indicators' value axis. Thus, all patient's parameters are represented simultaneously to form a convenient and efficient analytical space for the doctor.

In order to display of the desired characteristics statistical analysis and intelligent algorithmical processing of significant amount of data is required. The best way to organise the process pipeline is to present single-processing-task srvices as a semantic graph to be able to pass through the analysis and processing depending on data initial state to form a final-state report. Patients' temporal data is a matrix of analysis indicators' resultes in columns distributed along a time axis in rows. The fact that analysis and measurements are not taken or obtained immediately on a regular basis (for example, body tempurature and blood pressure are measured everyday, however blood analysis is done once or twice a week), the matrix is not uniform and contains gaps. To build a proper temporal regular matrix prior calculations are desired (Fig. 3).



	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9
A	a				d				g
B		b				e	h		
C			c	f				i	

Sparse temporal matrix

	t_{1-3}	t_{4-6}	t_{7-9}
ABC	x	y	z

Integral state vector

	Before event X			Event X			After event X		
	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9
A	a				d				g
B		b				e	h		
C			c	f				i	

Sparse temporal matrix masked by events

	t_{1-3}	t_{4-6}	t_{7-9}
A	a	d	g
B	b	e	h
C	c	f	i

Compressed matrix

Fig. 3. Matrix transformation: from sparce matrix to integral state vector

Thus, to organize a temporal matrix, the following chain of processing stages is required:

(1) requesting a list of episodes by patient's identifier;

(2) requesting analysis and temporal event data by episode identifiers;

(3) segmentation of sparse temporal matrix based on temporal events and value changes;

(4) matrix compression based on event intervals;

(5) calculation of integral patient's state vector.

Note that the entrance point may vary (a complete chain of processing methods 1-5 is not obligatory and may vary depending on initial state of data and desired state). Service semantic graph is an intelligent way to define the processing path provided that each service requires definite state and type of data and outputs processed data in a described well-defined state and type which is described ontologically.

Each service of semantic service graph should provide information on input data it requires (taking into consideration that multiple input may be needed), provided output data, which may be multiple as well, service API end-point and service metadata such as service name and description. Both input and output data objects are individuals of one superclass (same range of objects) so that output of one service easily becomes an input of the following service.

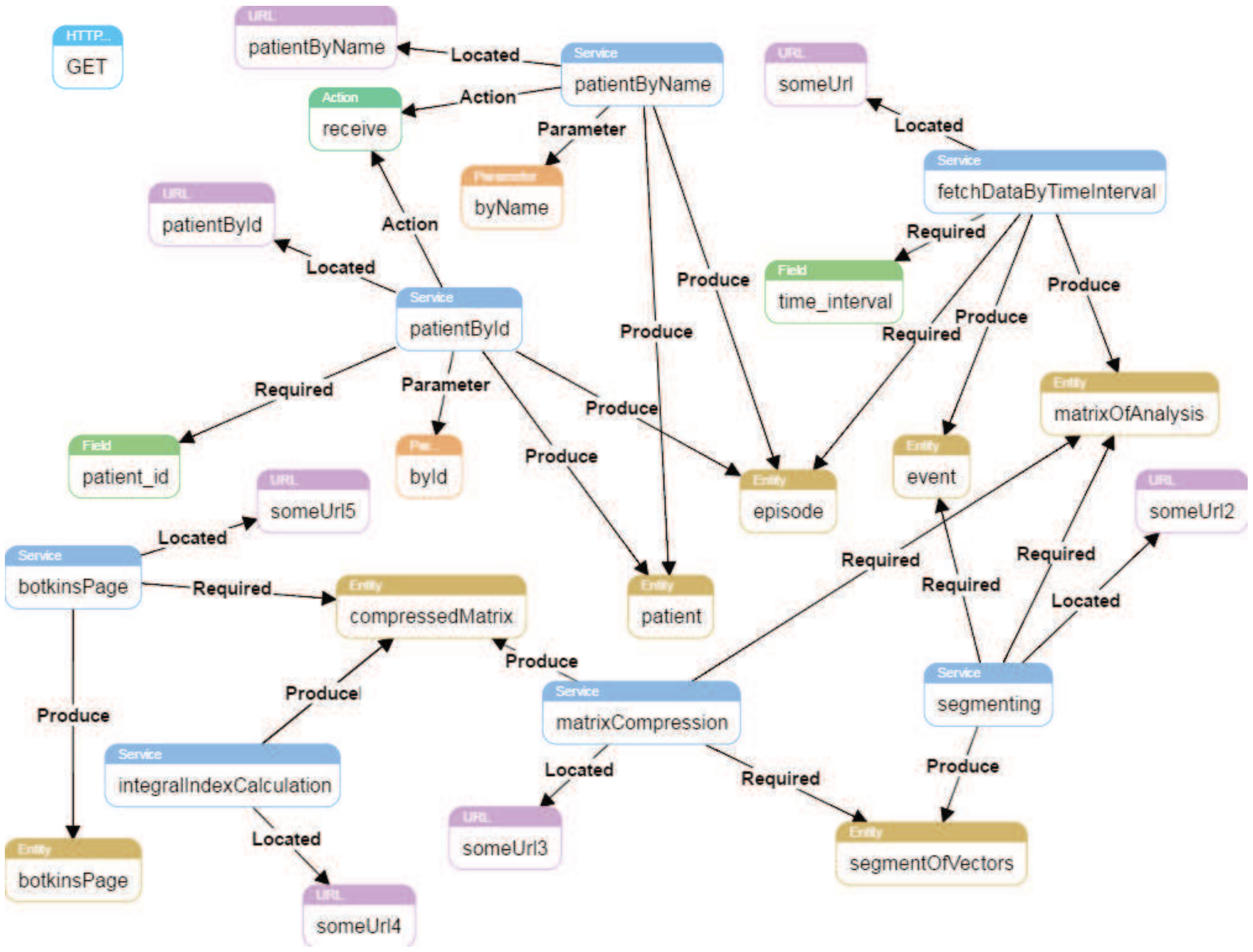


Fig. 4. Semantic service knowledge graph

As soon as all processing services are orgainazed as a semantic knowledge graph and SPARQL-endpoint is deployed, a basic combination of queries is able to trace a service path to process initial data so that it is converted to desired data formats. Three types of queries are of high importance for and data transformation and temporal matrix display:

(1) define services based on initial data state and relations between the services;

(2) define which service is required to proccess existing state of data;

(3) trace tha path from initial data state to temporal matrix final data.

Execution of a processing chain leads to a final data state required for Botkinsky sheet which includes patients's identifier, analysis matrix provided with event meta-information (segments) and a vector of patient's integral state indicator. Note that segments are desired to be manually shifter in cases doctor considers it to be necessary which means that not a full transformation chain is to be executed, but only the integral indicator calculation stage which is resolved by means of a new query which defines a contracted processing stage based on another current state of data.

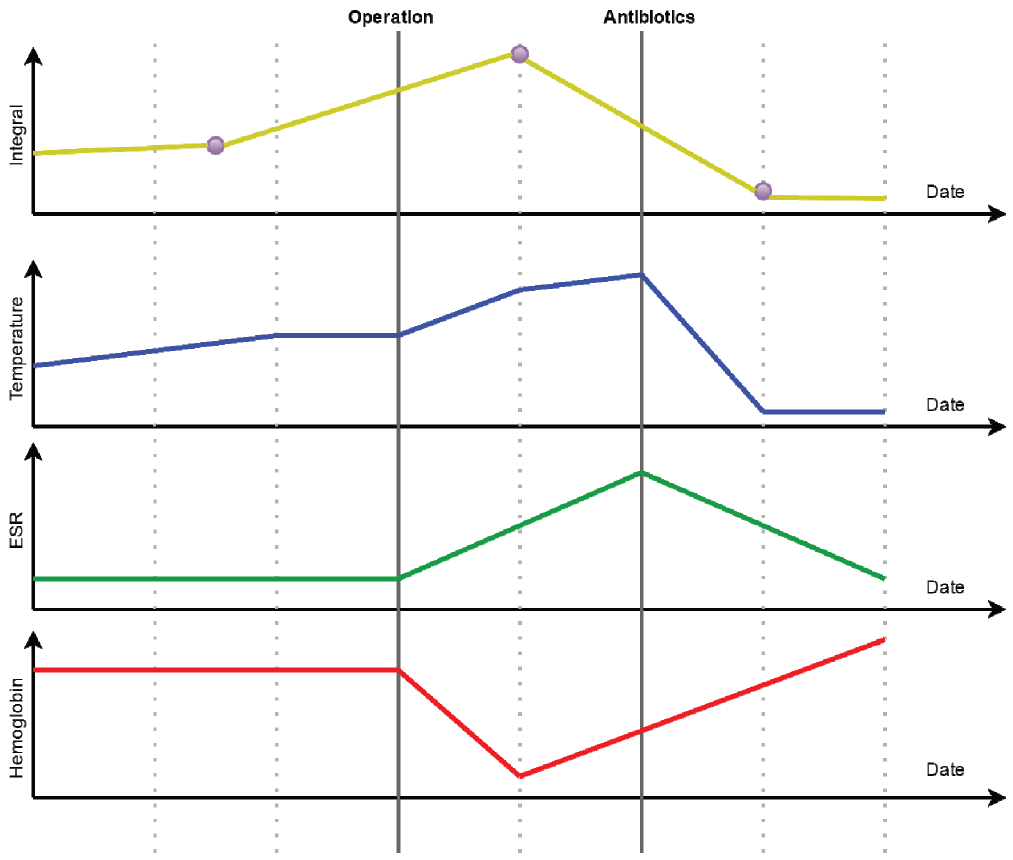


Fig. 5. Botkinsky sheet for three parameters (hemoglobin, ESR and body temperature) and patient's integral state indicator based on two events

8 CONCLUSION

Information systems of modern cities grow very intensively, mumber of available services permanently grows that makes very difficult to realize effective business processes in the frames of information systems used in city infactructure. In the paper a platform for services agile integration that allows link services is proposed. The agile platform can be positioned as a platform that implements a light weight semantic web services approach. The platform was tested while solving problems in medcine domain. Results of tesing proved efficiency of the proposed solution and showed that it can be applied to solve complex problems in any applied subject domain up to the domain of cities infrastructure. The platform is capable to form the base for morden urban systems.

Future directions of the platform development are oriented on adding new features and tools, which should allow to solve more complex problems such as coordinated development of several cities.

9 REFERENCES

Rudi Studer, Stephan Grimm, Andreas Abecker. SemanticWeb Services Concepts, Technologies, and Applications. Springer-Verlag Berlin Heidelberg 2007
Jorge Cardoso Semantic Web Services: Theory, Tools, and Applications IGI Global. Hershey PA 2007
http://www.w3.org/Submission/OWL-S/. OWL-S: Semantic Markup for Web Services
Zhao-hui Wu, Hua-jun Chen Semantic Grid: Model, Methodology, and Applications. Springer Berlin Heidelberg New York. 2008