SIGMA – SemantIc Government Mash-up Application.
Using Semantic Web Technologies to Provide Access to Governmental Data

Claudia Gheorghiu, Radu Nicolescu, Andrei Vasile Bogdan, Cristian Ochisor, Sabin C. Buraga, Lenuta Alboaie
Faculty of Computer Science, “A. I. Cuza” University of Iasi, Romania
busaco@info.uaic.ro – http://www.purl.org/net/busaco

Abstract

The paper presents SIGMA (SemantIc Government Mash-up Application), a platform able to create mash-ups by providing access to open governmental data. The proposed solution is based on the existing semantic Web technologies (especially RDF and SPARQL) and uses public distributed endpoints conforming to the Linked Data initiative. Also, SIGMA allows creating and deployment of independent plug-ins, encouraging the developers to build innovative applications, visualization solutions and mash-ups that give an improved access to the governmental data.

1. Introduction

Nowadays, a good government activity implies a good administration for a state’s welfare and, also, a good communication by promoting transparency and openness. There is a common trend that governments are adopting an open administration [7] by entering the digital world and publishing information on the Internet, encouraging participation and collaboration between administration agencies and citizens. One important aspect is to encourage the (re)use of the available data/information/knowledge for citizens and organizations by providing datasets of interest.

Datasets are found in a wide variety of formats, including spreadsheets, XML and RSS/Atom feeds. The structure and content of the existing data archives are sometimes hard to be understood and to be analyzed, requiring additional processing in order to be used even by the users with no programming skills.

To accomplish these goals, we are proposing SIGMA, a complex semantic Web-based platform in order to give support for creating and deployment of mashups using open (governmental) data.

The platform functionality is provided by separate modules (services) and plug-ins (helpers). By using a SOA-like architecture, SIGMA can be easily extended.

For the end-users, our system offers an interactive tool able to query certain SPARQL endpoints of interest, in order to visualize, analyze, and enhance the results. The platform uses current semantic Web standards (briefly presented in Section 2) and is addressed to any type of users, not necessarily the ones having a previous technical training. The previous SPARQL knowledge is not necessarily. Also, this tool is not restricted to a certain domain of interest (e.g., health, economy, legislation) and it can integrate heterogeneous data sources. Several current solutions refer only to specific domain or topic (see Section 3).

Other applications use datasets through REST Web services, or from locally stored databases. From the technical point of view, SIGMA is dynamically using SPARQL endpoints provided by governmental organizations (details provided by Section 4). It is a platform that allows creating and deployment of independent plug-ins, automatically incorporating their functionality into the system – consult Section 5.

The processed information are later presented to end users in a filtered, coherent and user friendly way.

SIGMA encourages developers to build innovative applications, visualizations and mash-ups that give to citizens an improved access to governmental data. A case study is described in Section 6.

2. Semantic Web Vision. Brief Presentation

The WWW space is primarily compounded by pages (documents that contain mark-ups) with information in the form of natural language text and multimedia intended for humans to read and to understand. Computers are principally used to render this hypermedia information, not to reason about it.
Information needs no longer to be intended for human readers only, but also to be processed by the machines, enabling – among others – intelligent information services, personalized Web applications, and semantically empowered search engines. This is the seminal idea of the semantic Web.

When advancing towards the semantic Web, the main obstacle is the effort of organizing the knowledge – content, meta-data, ontological constructs – made by the existing content providers. In the current systems, the users must work with certain vocabularies, tagging entities and relations. The purpose of these processes is to make the data comprehensible also for the computers [2, 10].

### 2.1. Denoting Resources

Each resource available on Web is uniquely addressed by using the Uniform Resource Identifiers (URI). The most representative URI subset is the one that deals with Uniforms Resource Locators (URL) – for example, to denote a geographical location we can use http://geo.info/Continent/Country/City.

### 2.2. Expressing Meta-data

To express and process meta-data (data about data), the Resource Description Framework (RDF) model can be used. RDF is a current standard of the World Wide Web Consortium (W3C) [10] and an important brick of the semantic Web. RDF is a language for representing information about resources in the World Wide Web space. The RDF format is intended to be used to capture and state the conceptual structure of existing information. The core concept is that of making statements about (Web) resources – the so-called RDF triples – of the form: “Subject has a property whose value is an object” in which the key components are the following:

- The **subject** is the resource being described (for example, a Web site, a software component, a person, or another specific concept).
- The **property** (predicate) is a characteristic of the subject on which the statement focuses – for example: `creator`, `hasLocation`.
- The **object** denotes the actual value of the property for the described subject (for example, a literal like “SIGMA” or another resource denoted by an URI).

In this manner, the RDF assertions – triples of URIs having the structure `<subject, property, object>` – can be viewed as a data model for describing semantics of data that could be processed by the computers.

Via the RDF assertions, we can attach meta-data to the Web resources, by eventually using common vocabularies describing “things” of interest: persons, software components, properties, domains, etc. For example, relations between certain types of resources can be expressed via popular vocabularies such as DCMI (Dublin Core Meta-data Initiative), FOAF (Friend Of A Friend), etc. Also, meta-data can be embedded into the Web pages themselves – this is the case of the RDFa [10] initiative.

Using these simple assertions, for which RDF defines diverse means of serialization (in XML, N3, etc.), complex representations of things can be built by composing them and interlinking resource descriptions. In the context of describing data, RDF and the semantic Web promote the idea of an open world where descriptions are not meant to be interpreted as exhaustive, but only as statements about the knowledge from a particular point of view: the fact that some things are not contained in a description does not mean they do not exist, only that the description author does not know them, or cannot state anything about them, another source could provide those descriptions.

To gain benefit of the full potential of Semantic Web, the main idea is to start publishing data as RDF. Existing data can be interconnected for further uses, in order to assure – among others – the interoperability.

Additionally, a query language for RDF data is already a W3C standard: SPARQL (SPARQL Protocol and RDF Query Language) [2, 10]. This language is allowing access to resources in the triple statement data model, and defining the access guidelines to expose such data through services – so-called SPARQL end-points.

### 2.3. Specifying Ontological Models

The **ontologies** are the next step of knowledge description, evolved from the need to represent concepts and relations between concepts, to enable formal definitions of domains knowledge and reasoning about the knowledge, in a rigorous (potentially machine understandable) form. An ontological model offers an abstract, simplified version of the world – conceptualization – in a formal and declarative representation, able to be processed by a computer [2].

To represent these formalizations on the Web, RDFS (the RDF Schema) [10] was defined as an extension of the RDF to include basic features to describe simple ontologies (vocabularies, taxonomies). For a higher expressivity, Web ontology languages
were specified to allow the description of more complex relations, such as the cardinality of a property, characteristics of properties, class relations, or the equivalence of concepts. One important standard is the Web Ontology Language (OWL) [2] that can be used to structure and characterize resources and/or relations between them. Knowledge about the resources can be shared within a given community of practice. By using OWL and related languages, we model ontologies – documents describing abstract, inference and logic information representation: knowledge about resources.

For details and case studies, consult [4] and [10].

3. Governmental Mashups. State of the Art

3.1. Open Government Data Metaphor

Good governance should be focused to an open administration. Open state management must be transparent (the public can have access to information regarding the government’s activities). Ideally, any citizen could express her/his opinions about designing and implementing public policies, opinions that can be taken into account by the governors.

An important initiative is the creation of the Open Government Data (http://okfn.org/), which implies sharing governmental records with all citizens.

Defining what open government data represents must take into account the complexity of the legal and social implications involved. The ALA – American Library Association’s key principles of open government broadly describes the rules that a government must meet in sharing information to the public. Information cannot be categorized as public if it cannot be accessed on the Internet for free.

Also, open data must be timely, accessible, machine processable, non-discriminatory, non-proprietary, license-free, with permanent Web addresses. Additionally, open data promotes analysis and can offer provenance and trust.

The public and digital information obtained from governmental sources can refer to reports, databases, media files, and any type of data that does not have restrictions regarding privacy, security, or copyright. The records that present restrictions governed by law are not considered to be in the open data category, according to the principles stated by http://www.opengovdata.org/home/8principles.

To implement these principles, one of the highest profile efforts is data.gov, the main open portal for the United States administration. Data.gov [6] makes available a large amount of official datasets, on topics ranging over all aspects of government activity, by using linked data principles [3]. In the United Kingdom, data.gov.uk [9] was created as a data store for administration records, with topics ranging from geospatial, financial, and legal data.

3.2. Existing Mash-ups

Many local administrations are starting to use the Internet to provide open government data. The goal is to create useful Web and/or mobile applications and also to encourage people to be an active part of their citizenship duties [1, 7, 8].

Nowadays, the developers have a huge amount of information to work with in order to create different software applications like portals, wikis, mash-ups or other useful solutions. The purpose can vary from crime statistics by neighborhood to finding best jobs in a town and from providing the environmental health status of a region to assisting users with special needs.

But the existent applications are using local and closed resources of previously restructured datasets obtained from governmental providers. They are not dynamically using the information shared by the public administration (data can easily be deprecated).

iLive.at mash-up merges records found in multiple data sources including transportation data (subways, buses), emergency services, crime statistics, recreation facilities and demographics. As a data-source, the application uses information gathered from Dublin Core Data Catalogs stored in a local MySQL database that is queried each time a request is issued.

Another example is This We Know that allows exploring U.S. government data providing different facts, such as demographic data from the Agency of Commerce, factory information from the Environmental Protection Agency or information about employment from the Department of Labor. The system uses as primary data source data.gov catalog on the basis of the semantic Web standards.

WhereDoesMyMoneyGo.org is a free, online tool to find out about where public money in the UK is spent. Access to the information is done through a JSON API, so the result of a request is a JSON object.

GovWILD application integrates open government data about politicians, parties, government agencies, funds, companies, and industrial leaders into a joint data set. The tool has a search engine that allows querying and browsing the data in an interactive manner. Besides search results, it graphically displays on a Web interface the connected entities from the RDF graph, to be explored by users.
4. Our Proposal: SIGMA Platform

The main target of our solution is to provide a platform that enables the development and integration of separate mash-ups using open data from government servers storing RDF files. The end user will be able to visualize, filter and analyze the results obtained by issuing queries. The application is not restricted to a certain domain of interest (e.g., health, economy, or legislation) by integrating heterogeneous types of data.

In other words, SIGMA represents a generalized method for managing knowledge [2] on the basis of government sources. Unlike other applications that use datasets through REST Web services, or from locally stored databases, SIGMA dynamically queries governmental SPARQL endpoints. These distributed endpoints are Web services using REST paradigm.

SIGMA is a mash-up-like platform, due to the different sources that are incorporated in the application, and presents the information filtered, in a coherent and user friendly manner.

4.1. Data Sources

The used data-sources are the current governmental endpoints – Web applications that enable users and applications like SIGMA to submit queries against RDF data-stores and obtain the result in JSON, XML, XLS or RDF file formats. The currently registered endpoints in SIGMA application are: data.gov, data.gov.uk, and the SPARQL endpoint from the University of Berlin – these data sources are available on the basis of the Linked Data initiative [3].

Additionally, we are using our custom endpoint to have access to Romanian administrative articles. Currently, Romanian government does not provide public datasets.

By visiting the data.gov portal, a user can have access to “raw” datasets and various tools that offer the possibility to mine the searchable catalogs. In the “raw” data catalog, documents can be shared in different formats. The site also hosts a set of RDF files created by converting a number of data.gov datasets into RDF triples. The RDF files can be interrogated by using a single SPARQL endpoint.

Data.gov.uk hosts a great amount of information stored as datasets that are available for download for regular users in order to have access to the information provided by the central UK government departments and a number of public sector agencies and local authorities. Similar to the US solution, data.gov.uk stores “raw” data catalogs available in various formats for download, but also a set of RDF documents. The UK endpoint server is divided depending on the category of the data. This is why one can use different endpoints for launching queries regarding: analytics, environment, finance, education, transport, ordnance survey or crime.

The University of Berlin makes available to query the CIA World Factbook using a single SPARQL endpoint. The World Factbook provides information on the history, people, government, economy, geography, communications, transportation, military, and trans-national issues for 267 world entities.

4.2. SIGMA Architecture

The platform functionality is provided by separate modules (services) and plug-ins (helpers). By using a SOA-like architecture, SIGMA can be easily extended. This can be achieved by adding a new functional component to the list of the existing ones, registered in the application. Adding a new component does not require a previous configuration or modification inside the source code of the SIGMA solution. Therefore any developer can create new features, insert them inside the platform, and SIGMA will automatically load them into the system.

The general view of the SIGMA architecture is depicted by Figure 1.

One central component is the Semantic Interrogation Tool (SIT) that allows the user to interrogate administration endpoints. All the queries use SPARQL language and the data is modeled in RDF format. The SIT module retrieves information from the registered endpoints and, depending on the nature of the data, performs statistical analysis or displays them on a map. Using geo-location meta-data (latitude and longitude), specific data can be placed on a virtual map using different markers. It may be combined with other results and become the input for different types of charts.

The governmental sites make available thousands of raw datasets. But for regular users, without technical knowledge in processing this type of data, the provided data is useless. SIGMA offers a tool that
enables any inexperienced users to have access to the information, without worrying about the format or nature of the files. They can navigate through filtered and human readable facts and use it as they need.

A more detailed view of the SIT internal structure – using Model-View-Controller (MVC) [5] paradigm – is provided by Figure 2.

View contains four main parts: (1) the list of the categories where the data types found on the selected endpoint are displayed; (2) the subjects list that contains all the subjects available for the selected category; (3) the predicates list that displays all the properties found for the selected category; (4) the plug-ins view where each plug-in is displayed on screen after the loading is complete.

When an endpoint is selected, the SPARQL service launches a query against it and retrieves the list with all the categories (data types) that are available. Once the request is complete the result is stored in the Application model and the View is notified to refresh the categories list.

When the user selects one category from the list, the SPARQL endpoint is invoked again to get the list of the subjects and predicates corresponding to the selection. After the application receives and parses the response, the data is stored in Application model and both subjects and predicates list are updated to reflect
the changes in the model. At this point, the user can select several subjects from the list and choose the property that (s)he wants to analyze.

Once the selection is made, the Plugins service searches for the available plug-ins. Once the search is completed, the list is saved in the Plugins model and is filtered to determine which one can display the selected type of data. An instance of each found plug-in is created and displayed in the Plugins view. If no plug-in is found, then a message is displayed notifying the user that the selected property cannot be processed by the application at the moment.

Our ambitious idea is to create a specific Romanian version of the US and UK open data services.

The www.gov.ro Web site provides updated information about Romanian government activity such as press releases, executive meetings, and events. In order to give a better perspective about an event, links to audio/video files, photo galleries, reports, statistics, laws or other kind of resources can be attached to the articles. Also, related articles are suggested (articles from the same event or on the same topic).

5. Implementation Details

5.1. Semantic Interrogation Tool

The Query Tool Module allows users to have access to raw government data in open formats. It was implemented in ActionScript using Adobe AIR Framework due to its rich user interface components.

The classes in the model are responsible for storing all the information needed in using the interrogation tool: endpoints, query results, application state – the current endpoint and subject or predicate.

The business logic of the application is built inside the services package from the application, which serves as the controller. The user can select one endpoint from the list of servers registered in the tool in order to see what kind of information is available on the chosen server. This can be accomplished through a SparqlService class which asynchronously invokes the service.

To visualize the information regarding the results, the user must choose one (or more) of the subjects and then specify what property to be analyzed regarding that subject by consecutively electing one predicate. According to the type of data that the selected predicate refers to (the rdfs:range of the property in the triple RDF model), the controller will search for an appropriate plug-in embedded in the application. This functionality is provided by PluginsViewMediator which searches inside the plug-ins folder for files having .swf extension, loads them and verifies which ones can process the selected predicate. The chosen plug-ins will be instantiated and graphically displayed.

The controller could select the ChartPlugin for displaying a column chart with the input, the values of the predicate for the selected subject(s). In case the range of the predicate is vcard, then the VCardPlugin will be instantiated to present the name of the subject, the street, postal code or locality.

Regardless of the range of the predicate, the application will test if the resource has geo-location
meta-data. If so, the MapPlugin will load the location on a Google Map and allow user to interact with it. This plugin makes available multiple functionalities: creating layers on the map, retrieving information for each country using the CIA Factbook dataset and displaying POI (point of interest) near the location in question – e.g., hospitals, schools, or hotels.

All the graphical content is rendered by the classes found in the view package.

5.2. RoGov Crawler and Public Endpoint

For extracting the semantic meta-data concerning selected Web pages from www.gov.ro, the crawler uses the Alchemy library (www.alchemyapi.com/api/) to enhance the RDF files with data regarding people, companies, organizations, cities, geographic features. Also, important terms (keywords) are extracted, the text being categorized (for example: science technology, culture politics, legislation) and automatically tagged in a manner similar to human-based tagging. Concept tagging is capable of making abstractions – e.g., Romania + Hungary + Ukraine + Moldova = Eastern Europe Countries.

The concrete implementation is based on C# within .NET Framework.

Each extracted article is stored as RDF triples in the N3 format by using the Virtuoso server. The gathered information will be publicly exposed via a SPARQL endpoint made available by Virtuoso.

5.3. Performance and Security Considerations

SIGMA is built as a platform. Each time the information needs to be displayed, the system search for available plug-ins that are designed to load the specified type of data. If one (or more) plug-ins are available, it will be loaded and the needed data will be passed to the plug-in (the URI of the predicate, the URI of the resource and the selected endpoint). Once the plug-in receives this information, it can start loading the records and display them in a custom manner. Beside the RDF data, a plug-in can also use other external data-sources or APIs, independent from the main application. Using this approach, the platform can be extended by adding new plug-ins created by third party developers.

Because SIGMA relies completely on remote endpoints for retrieving data, the performance of each operation depends on the quality of the services offered by the data provider.

From the security point of view, the implementation uses the primary security model of Adobe AIR platform. Despite the fact that the application is mainly based on Web technologies, it has more privileges than a regular Web application, like access to the local file system, which imply a greater degree of trust. For this, the security system is based on two different sandboxes: application sandbox (permits only the content loaded from the application home directory and gives direct access to the AIR APIs) and non-application sandbox (permits local or remote content to be loaded into the application and restricts direct access to the AIR APIs).

6. Case Study

We consider the situation when a user needs information regarding the education facilities available in a given geographical region.

On the education SPARQL service, the UK government has published all sorts of information regarding schools and nurseries: free school meals, address, district, close date, religious character, etc. Imagine there is a parent who wants to obtain an overview of schools in England in order to select the best unit for her child. She would like to view all categories of schools. Because she does not want to miss anything in terms of criteria for evaluating such an institution, she needs to know all available information about a school. She also wants to be able to perform comparisons between several schools, the information to be graphically displayed, so that she can analyze and filter them as appropriate.

In order to see the registered categories of UK schools, the user must select the Education endpoint. The application will retrieve the main types of schools found on this category and display them.

Figure 3. Properties of interest about a school unit

We can suppose that the user is interested to find more information about the foundation schools. When selecting the item of interest, the institutions from this category will be presented. In the properties panel, for
the located items, the inferred predicates (properties of interest) will be available – see Figure 3.

To see the associated information for several schools and to obtain a comparison between the chosen units, the user could select one or more schools and then specify what kind of property to be analyzed. For example, if the free school meals property is chosen, the ChartPlugin will display a line chart using the values of the provided property for each item.

Automatically, the SIGMA application will try to point on a map the selected items, by testing if they have geo-location meta-data. This feature is provided by the MapPlugin which uses Google Maps service. Additionally, on the map, the user can be able to view information about each country by selecting the countries view layer. Also, she can view different points of interest, like hospitals or hotels near the marked area (Figure 4).

![Figure 4. MapPlugin in action](image)

### 7. Conclusions and Further Work

A platform (SIGMA) – able to create mash-ups providing access to open governmental data – was presented. Our approach is based on the existing semantic Web technologies and uses public distributed endpoints conforming to the linked data initiative [3].

SIGMA performs queries by using SPARQL endpoints, retrieving the available data and displaying it in a human readable and user friendly way. Functional plug-ins can perform complex data analysis and correlations. SIGMA platform is addressed to any kind of users that want to use the governmental information published on administration sites [7]. The system does not require previous knowledge about creating SPARQL queries nor about the ontologies used to model and organize these kinds of data. All the transformation and filtering is automatically performed by the platform, so that users can visualize the output of their request in a comprehensive way. Also, the input data-sources can be easily modified, SIGMA not being constrained to a specific data type. The system is flexible enough to accept any types of sources. From the architecture point of view, the application is extensible via independent plug-ins.

We shall consider several future developments – e.g., performing inferences and aggregations in order to obtain a relevant result regarding specific topics of interest and integrating new functional plug-ins.

### 7. References


