State of the Art

Iuliana Bejan, Bogdan – Ștefan Hagiu, Sergiu Lucuțar, Valentin – Marian Șpac

Facultatea de Informatică, Universitatea „Alexandru Ioan Cuza”, Iași

[iuliana.bejan@info.uaic.ro](mailto:iuliana.bejan@info.uaic.ro)

[stefan.hagiu@info.uaic.ro](mailto:stefan.hagiu@info.uaic.ro)

[sergiu.lucutar@info.uaic.ro](mailto:sergiu.lucutar@info.uaic.ro)

[marian.spac@info.uaic.ro](mailto:marian.spac@info.uaic.ro)

**Abstract.** In this paper …tralalala

**Key words:** Web Semantic, reactive programming, ontologies, RDF, RDFS, OWL, SPARQL

1. Introduction

One of the most important and successful services of the internet, World-Wide-Web, - shortly called Web is an information space where documents and other web resources are identified by URLs, interlinked by hypertext links, and can be accessed via the Internet.

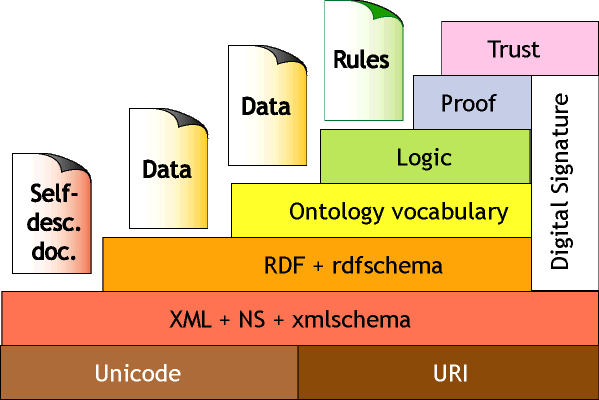
This quantity of information is growing since Web came into being. [WorldWideWebSize](http://www.worldwidewebsize.com/) estimates that today the web contains over 4.74 bilions of pages. This causes the data to become the new gold due to the complexity of finding in a fast way relevant information.

SAY SOMETHING ABOUT OUR APP, BapR

1. Technologies

Ideea de Web Semantic a fost introdusă de către Tim Berners-Lee. Astfel, un motor de căutare într-un web semantic ar lua în considerare întreaga sintagmă, precum și contextul, returnând în urma unei astfel de interogări doar rezultatele relevante pentru utilizator. Web-ul semantic ar trebui să permită ca datele, localizate oriunde pe web, să fie accesibile și întelese atât de oameni, cât și de calculatoare.

În vederea soluționării aspectelor prezentate mai sus, Web-ul semantic adoptă o viziune stratiﬁcată.



**Fig. 1.** Stratificarea web-ului semantic

Pentru ﬁecare nivel, Consortiul Web a standardizat sau urmează să standardizeze diferite limbaje bazate pe familia XML.

* 1. XML

**Extensible Markup Language** (**XML**) is a [markup language](https://en.wikipedia.org/wiki/Markup_language) that defines a set of rules for encoding documents in a [format](https://en.wikipedia.org/wiki/File_format) which is both [human-readable](https://en.wikipedia.org/wiki/Human-readable_medium) and [machine-readable](https://en.wikipedia.org/wiki/Machine-readable_data). It is defined by the [W3C](https://en.wikipedia.org/wiki/World_Wide_Web_Consortium)'s XML 1.0 Specification and by several other related specifications, all of which are free [open standards](https://en.wikipedia.org/wiki/Open_standard).

It is the base level needed for exchanging data in a structured or semi-structured method. It can not express the semantics but only reflects an uniformed sintactic way to represent data.

* 1. RDF

Pentru a reprezenta datelor într-un mod unitar (standardizat) se folosește Resource Description Framework (RDF).

Resource Description Framework (RDF) (http://www.w3.org/TR/2004/REC-rdf-primer-20040210/) este un limbaj folosit pentru reprezentarea informațiilor despre o resursă in World Wide Web. RDF-ul lucrează pe triplete ce ajută la descrierea unei resurse:

Subiectul are un predicat - o proprietate, a cărei valoare este un obiect.

• subiectul reprezintă o resursă care este descrisă;

• predicatul reprezintă o caracteristică a subiectului;

• obiectul reprezintă valoarea predicatul ce descrie subiectul.

Prin folosirea acestui model, pentru care există mai multe modalități de serializare (XML, Turtle, N-Triples, etc), se poate descrie orice tip de resursă indiferent de complexitatea ei (prin computarea și interconecta-rea descrierilor).

Descrierea unei resurse trebuie văzută ca o declarație făcută de o enti-tate și nu ca o definiție pentru acea resursă. Dacă în descrierea unei re-surse nu regăsim anumite proprietăți asta nu înseamnă că ele nu există ci doar că autorul nu știe sau nu poate spune nimic despre ele, un altul pu-tând să completeze acele proprietăți.

Pentru ca două entități, (autori) să poată vorbi despre aceeași resursă, subiectul, predicatul și obiectul (în cazul în care nu e un literal exemplu: număr, string) dintr-o declarație RDF, trebuie să fie un Universal Re-source Identifier (URI).

Subiectul unei declarații RDF poate fi un URI sau un nod gol, în am-bele cazuri ele reprezintă o resursă. Resursele identificate de noduri goa-le sunt așa numite resurse anonime. Predicatul este un URI ce indică o resursă, care reprezintă relația dintre subiect și obiect. Obiectul fie este un URI spre o nouă resursă (cele mai valoroase legături sunt acelea care conectează o resursă de date externe) sau un literal.

* 1. RDFS

RDFS oferă un vocabular de modelare a datelor, bazându-se pe RDF, prin care se descriu grupuri de resurse (clase) și relațiile dintre acestea (descrierea relațiilor este un plus major pentru că ne oferă posibilitatea interpretării la nivel semantic a proprietăților, relațiilor dintre resurse). Modelul oferit de RDF Schema este similar celui oferit de limbajele de programare orientat obiect, însă spre deosebire de acestea ce descriu o clasă prin proprietățile ce îi pot fi atribuite instanțelor sale, RDF Schema definește global proprietățile. Acestea nu sunt încapsulate ca atribute în definiția clasei, astfel o persoană (autor) poate să definească noi proprie-tăți fără a schimba clasa din care face parte instanța.

RDFS pune la dispoziție o sintaxă standardizată pentru specificarea de ontologii, și un set de primitive de modelare, precum relații de tipul subclasă sau instanță.

* 1. OWL

O ontologie se ocupă cu definirea vocabularului, a semanticii ce va fi folosită în documentele RDF. Pentru definirea vocabularului, a terminologiei, se folosesc clasele, subclasele și instanțe care, în OWL, sunt numite indivizi. Aceștia sunt membrii unei clase OWL și sunt extensii ale acelei clase.

Clasa în OWL reprezintă clasificarea unor indivizi în grupuri ce au caracteristici asemănătoare. Dacă un individ este membru al unei clase, îi spune cititorului că acesta se încadrează în clasificarea semantică dată de clasa OWL.

În OWL, există două tipuri de proprietăți ce leagă indivizi, instanțele claselor:

* owl:ObjectProperty leagă doi indivizi ce fac parte din clase diferite;
* owl:DatatypeProperty leagă un individ (instanță) a unei clase de o valoare li-terală

Pentru exempul de mai sus vom avea :

<rdf:RDF

xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"

xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"

xmlns:owl="http://www.w3.org/2002/07/owl#"

xmlns:mf="http://www.myfriends.fake/mf#"

xmlns:animals="http://www.test.com/animals#"

>

<owl:Class rdf:about="http://www.test.com/animals#animaltype">

<rdfs:label>The animal type</rdfs:label>

<rdfs:comment>The class of all animal types.</rdfs:comment>

</owl:Class>

<!-- Define the family property -->

<owl:DatatypeProperty rdf:about="http://www.test.com/animals#family"/>

<!-- Define the similarlyPopularTo property -->

<owl:ObjectProperty rdf:about="http://www.test.com/animals#similarlyPopularTo"/>

<owl:Class rdf:about="http://www.test.com/animals#cats">

<rdfs:subClassOf rdf:resource="http://www.test.com/animals#animaltype"/>

<rdfs:label>Animal cat</rdfs:label>

<rdfs:comment>...</rdfs:comment>

</owl:Class>

<owl:Class rdf:about="http://www.test.com/animals#dogs">

<rdfs:subClassOf rdf:resource="http://www.test.com/animals#animaltype"/>

<rdfs:label>Animal dog</rdfs:label>

<rdfs:comment>...</rdfs:comment>

</owl:Class>

<rdf:Description rdf:about="http://www.test.com/animals#Bonnie">

<rdf:type rdf:resource="http://www.test.com/animals#dogs"/>

<animals:family>Mammals</animals:family>

<animals:similarlyPopularTo rdf:resource="http://www.test.com/animals#similarlyPopularTo"/>

</rdf:Description>

<rdf:Description rdf:about="http://www.test.com/animals#Clyde">

<rdf:type rdf:resource="http://www.test.com/animals#cats"/>

<animals:family>Mammals</animals:family>

<animals:similarlyPopularTo rdf:resource="http://www.test.com/animals#similarlyPopularTo"/>

</rdf:Description>

</rdf:RDF>

* 1. SPARQL

To be able to access the resources from the data graph we use an W3 Consortium specification named **SPARQL Protocol and RDF Query Language** (SPARQL) (<http://www.w3.org/TR/2013/REC-sparql11-query-20130321/>), an RDF query language.

The SPARQL query language extracts and interates through RDF graph using the SELECT instruction (it can be replaced with CONSTRUCT to extract information and transform the result into a valid RDF, ASK used to provide a simple True/False result for a query and DESCRIBE used to extract an RDF graph from the SPARQL endpoint) to determine which subset of the graph will be returned.

Each of these query forms takes a WHERE block which consists of triplets needed to build a filter graph which will be searched in the database.

For example, to extract the name and email of every person from a database, we will use the next query:

PREFIX foaf: <http://xmlns.com/foaf/0.1/>

SELECT ?name ?email

WHERE {

?person a foaf:Person.

?person foaf:name ?name.

?person foaf:mbox ?email.

}

To enable working with multiple ontologies, SPARQL provides a way to define prefixes similar to Turtle language of XML namespaces. In the example above, the „foaf” prefix replaces the URL where the ontology is located (<http://xmlns.com/foaf/0.1/>).

The results are returned in XML format.

To upgrade the current Web (Social Web) to Semantic Web some annotation mechanisms were developed which enables adding semantics to HTML data. This initiative helps the developers to easily add metadata to already existing web pages without major modifications.

* 1. RDFa

**RDFa** (http://www.w3.org/TR/xhtml-rdfa-primer/) defines a set of expressions which extends the XHTML and HTML5 by adding new attributes (property, datatype, about, typeof, etc) enabling the possibility to mark up existing human-readable Web page content to express machine-readable data. An HTML page can contain attributes to mark an article title, the content but also more complex attributes like personal information of the author.

* 1. Microdata

**Microdata** (http://www.w3.org/TR/microdata/) is a WHATWG HTML specification used to nest metadata within existing content on web pages. Search engines, web crawlers, and browsers can extract and process Microdata from a web page and use it to provide a richer browsing experience for users. Microdata uses a supporting vocabulary (e.g.: schema.org, GoodRelations etc.) to describe an item and name-value pairs to assign values to its properties. Microdata is an attempt to provide a simpler way of annotating HTML elements with machine-readable tags than the similar approaches of using RDFa and microformats.

* 1. Reactive programming

“In computing, reactive programming is a programming paradigm oriented around data flows and the propagation of change. This means that it should be possible to express static or dynamic data flows with ease in the programming languages used, and that the underlying execution model will automatically propagate changes through the data flow.” (Wikipedia)

“Reactive programming is programming with asynchronous data streams.” (André Staltz)

Reactive programming can be defined as a programming paradigm oriented around working with asynchronous data streams, allowing the date to be processed as it flows through the system, by creating, transforming, filtering, merging, mapping (, etc.) the data streams.

In reactive programming, a data stream is seen as a series of events ordered in time. An event can produce either a value, an error or a signal that symbolizes completion (no more data will be transmitted). For every event, it can be defined a function (callback function) that will capture the event and will produce an appropriate response. As seen, this methodology is based on the Observer Pattern, the functions are “subscribing” to the event and then are “observing” the stream and react accordingly.

Reactive Programming raises the level of abstraction of your code so you can focus on the interdependence of events that define the business logic, rather than having to constantly fiddle with a large amount of implementation details. Code in Reactive Programming will likely be more concise. The benefits of using this methodology is highlighted in the current webapps which are highly interactive and need to respond to a multitude of events.

A system developed in a reactive manner is:

* responsive - providing rapid and consistent response time in order to ensure a positive user experience;
* resilient - the system remains responsive in case of a failure;
* elastic - reactive system can react according to the input, by increasing or decreasing the resources allocated;
* message driven - Reactive Systems rely on asynchronous message-passing to establish a boundary between components that ensures loose coupling, isolation, location transparency, and provides the means to delegate errors as messages.

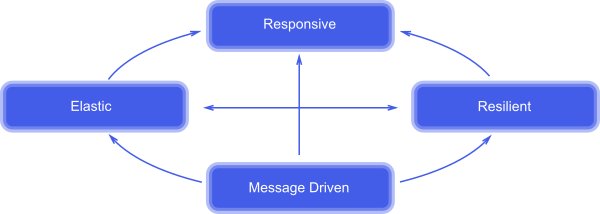


Fig. 1 Reactive system’s traits (Reactive Manifesto)

Due to the multitude of advantages this paradigm provides, many libraries embracing this principle emerged. Two of the most important are Rx and ReactJS.

Rx is a library for composing asynchronous and event-based programs in which asynchronous data streams are represented using a Observable interface, are queried (e.g. filter, project, aggregate, compose and perform time-based operations, etc.) using LINQ operators, and, in which, the concurrency in asynchronous data streams is parameterize using Schedulers (which controls when a subscription starts and when notifications are published).

Rx library family is widely available for many languages and platforms (.NET, Java, Scala, Clojure, JavaScript, Ruby, Python, C++, Objective-C/Cocoa,Groovy, etc).

ReactJS is an open-source JavaScript library for creating user interfaces that aims to address challenges encountered in developing single-page applications. React is intended to help developers build large applications that use data that changes over time. Its goal is to be simple, declarative and composable. React only handles the user interface in an app; it is considered to only be the view in the model–view–controller (MVC) software pattern, and can be used in conjunction with other JavaScript libraries or larger MVC frameworks such as AngularJS.

1. Concluzie

Pentru a putea crea un motor de căutare care să fie atent la semantică și care să returneze răspunsuri relevante este necesar ca, în spate, datele să formeze un vocabular ce să poată fi înțeles atât de om cât și de calculator iar interogarile ce sunt aplicate asupra bazelor de date să fie capabile să mențină contextul. Bazele de date ce permit interpretarea la nivel semantic a informației trebuie sa formeze un vocabular, pentru aceasta se folosește OWL , iar pentru interogarea acestora vom folosi SPARQL sau MQL.

1. Bibliografie

Berners-Lee, T. Weaving the Web. În T. Berners-Lee, Weaving the Web.

Buraga, S. C. Semantic Web fundamente și aplicații.

Dan Brickley, R. G. (fără an). RDF Schema 1.1. Preluat de pe W3C Recommendation 25 February 2014: http://www.w3.org/TR/2014/REC-rdf-schema-20140225/

Fabien Gandon, I. H. (fără an). An introduction to Semantic Web and Linked Data.

Flanagan, D. MQL Reference Guide. Preluat de pe Table of Contents: https://developers.google.com/freebase/mql/

Hayes, P. RDF Semantics. Preluat de pe W3C Recommendation 10 February 2004: http://www.w3.org/TR/2004/REC-rdf-mt-20040210/

Herman, I. Introduction to the Semantic Web International Conference on Dublin Core and Metadata Applications.

Leigh Dodds, T. P. web-integrated-data. Preluat de pe Open Data & The Semantic Web London Knowledge Lab: http://creativecommons.org/licenses/by/2.0/ uk/

linkeddatatools. Introducing Linked Data And The Semantic Web. Preluat de pe Semantic Web: http://www.linkeddatatools.com/semantic-web-basics

Mike Dean, G. S.OWL Web Ontology Language Reference. Preluat de pe W3C Recommendation 10 February 2004: http://www.w3.org/TR/owl-ref/

Steve Harris, A. S. SPARQL 1.1 Query Language. Preluat de pe W3C Recommendation 21 March 2013: http://www.w3.org/TR/sparql11-query/