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
stefan.hagiu@info.uaic.ro
sergiu.lucutar@info.uaic.ro
marian.spac@info.uaic.ro

Abstract. This paper is meant to familiarize the reader with the programming paradigm and technologies that will be used in implementing the Backpacker Recommender application.

1. Introduction

Backpacker Recommender (BapR) is a web application which it’s meant to assist travelers in finding places to visit, where to stay or where to get first aid. Backpacker it’s a smart application making use of knowledge databases and semantic web to provide meaningful results. The application will provide support for actions like finding a tourist information point, shopping, finding financial help, where to eat/sleep, what to visit/attend, contacting local hospitals, contacting mountain/sea/swamp rescue.

2. Technologies

2.1 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, al-

lowing 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.

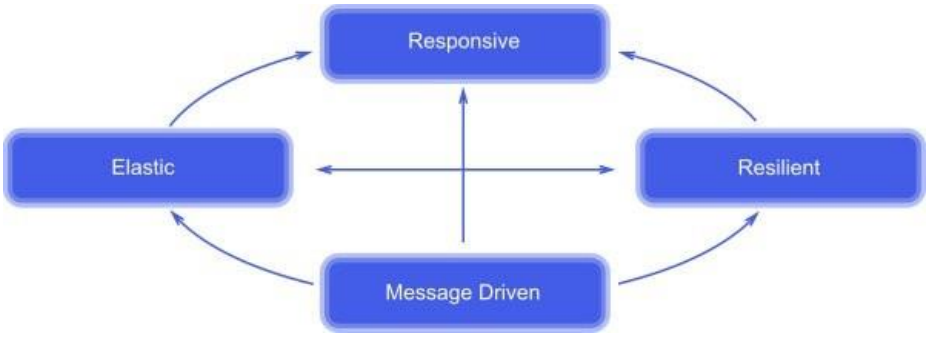

Fig. 1 Reactive system's traits (Reactive Manifesto)

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.

2.2 RDF

Resource Description Framework (RDF) is a flexible approach for data representation associated to a resource from World Wide Web. The RDF works on data that helps on defining a resource as following:

The subject has a predicate – the property that stores an object as it's value.

- the subject is the resource being described;
- the predicate is a characteristic of the subject;
- the object is the value of the predicate associated to the subject.

By using this model, that can be serialized in multiple ways such as: XML, Turtle, N-Triples, etc, we can describe any type of resource no matter how complex it is, by combining and interconnecting their descriptions.

The description for a resource must be seen as a declaration made by an entity and not as a definition for that resource. If in the description of a resource we don't find described certain properties it doesn't mean that they don't exist, but that the author itself doesn't

know or he cannot say anything more about them, another person can come and fill the properties that are missing..

In order to be referred by two entities, the subject, predicate and the object of a resource (in case it isn't a literal such as: a number or a string) from a RDF declaration, has to be an Universal Resource Identifier (URI).

The subject of a RDF declaration may be an URI or an empty node, in both cases, representing a resource. The resources identified by empty nodes are so called anonymous resources. The predicate is an URI that indicates a resource and represents the relation between the subject and the object. The object may be an URI to a new resource (most valuable links are those that connect a resource to an external data) or a literal.

2.3 RDFS

RDF Schema provides a data-modelling vocabulary for RDF data which helps describe groups of resources and the relations between them. These resources are used to determine characteristics of other resources, such as the domains and ranges of properties.

The RDF Schema class and property system is similar to the type systems of object-oriented programming languages but while these describe a class through the properties assigned to each individual instance, the RDF Schema properties are globally defined.

These are not contained in the in the class' definition, allowing to define and add new properties without changing the class to which the instance belongs to.

RDFS provides a standard syntax for specifying ontologies and a set of data-modeling properties like sub-class/instance relationship.

2.4 OWL

OWL (Web Ontology Language) is the ontology responsible for defining the vocabulary and the semantic that will be used in the RDF documents. In OWL, classes, subclasses and instances, the so called individuals, are used to define the vocabulary and the terminology of the RDF documents. These individuals are the members of one OWL class or they could be extensions of that class.

The class in OWL represents the classification of some individuals in groups that have similar characteristics. If an individual is a member of one class, it tells to the reader that it fits in the semantic classification given by the OWL class.

The OWL extenze the level of expressivity offered by RDFS by introducing the possibility of creating hierarchy between classes,

being able to say if two entities represent the same resource or if they are in a relation of disjunction.

The ability to unite different schemes of data (pooling of data describing for one resource from different databases) is very important because it allows the extraction of information for the same entity, object from databases such as DBpedia, Linked Database etc. obtaining a more complete description for that resource .

2.5 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), 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.

2.6 RDFa

RDFa 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.

2.7 Microdata

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.

3 Conclusion

Creating a application capable of retrieving the relevant data for a request made by a user implies, not only using the semantic data, but also to have a vocabulary consisting of informations that can be understood both by humans and computers. Also, the database queries have to be able to maintaining the context. In order to do that, the databases that allows the interpretation at a semantic level must form a vocabulary, in this case we are using OWL and for the database queries we are using SPARQL.

4. Bibliography

- Berners-Lee, T. Weaving the Web. În T. Berners-Lee, Weaving the Web.
- Buraga, S. C. Semantic Web fundamente şi aplicaţii.
- 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/
- “The introduction to Reactive Programming you've been missing” (André Staltz): https://gist.github.com/staltz/868e7e9bc2a7b8c1f754;
- “ReactiveX”: http://reactivex.io/tutorials.html;
- “The Reactive Manifesto”: http://www.reactivemanifesto.org/
- “Introduction to RX” (Lee Campbell): http://www.introtorx.com/
- “Overview of Reactive Programming” video (Jafar Husain): https://hackhands.com/overview-of-reactive-programming/
- “What is Reactive Programming?” (Kevin Webber): https://medium.com/reactive-programming/what-is-reactive-programming-bc9fa7f4a7fc
- “Reactive ReactJS: improving data flow using reactive streams” (Arian Stolwijk): http://www.aryweb.nl/2015/02/16/Reactive-React-using-reactive-streams/