Connecting your Mobile Shopping Cart to the Internet-of-Things

Nicolas Petitprez, Romain Rouvoy, Laurence Duchien

Connecting your Mobile Shopping Cart to the Internet-of-Things

Nicolas Petitprez, Romain Rouvoy, and Laurence Duchien

Inria Lille – Nord Europe,
LIFL - CNRS UMR 8022,
University Lille 1, France
`firstname.lastname@inria.fr`

Abstract. Online shopping has reached an unforeseen success during the last decade thanks to the explosion of the Internet and the development of dedicated websites. Nonetheless, the wide diversity of e-commerce websites does not really foster the sales, but rather leaves the customer in the middle of dense jungle. In particular, finding the best offer for a specific product might require to spend hours browsing the Internet without being sure of finding the best deal in the end. While some websites are providing comparators to help the customer in finding the best offer meeting her/his requirements, the objectivity of these websites remains questionable, the comparison criteria are statically defined, while the nature of products they support is restricted to specific categories (*e.g.*, electronic devices). In this paper, we introduce MACCHIATO as a user-centered platform leveraging online shopping. MACCHIATO implements the principles of the Internet-of-Things by adopting the REST architectural style and semantic web standards to navigate product databases exposed on the Internet. By doing so, customers keep the control of their shopping process by selecting the stores and comparing the offers according to their own preferences.

1 Introduction

With the explosion of the Internet and the increasing number of e-commerce sites, online shopping has reached an unforeseen success. This domain is raising a yearly revenue of several billions and involves major companies like Amazon or eBay. While online shopping was initially dedicated to high-tech products, one can observe that nowadays e-commerce websites are selling a variety of products ranging from food, to clothes, to spare parts, and even to cars. Nonetheless, the wide diversity of e-commerce websites does not really contribute to foster the sales, but rather tend to leave the customer in the middle of dense jungle. In particular, finding the best offer for a specific product might require to spend hours browsing the Internet without being sure of finding the best deal in the end. While some websites provide specialized comparators to help the customer in finding the best offer meeting her/his requirements, the objectivity of these websites remains questionable, the comparison criteria are statically defined, while the nature of products they support is restricted to specific categories.

In this paper, we therefore introduce MACCHIATO as a user-centered platform leveraging online shopping. MACCHIATO integrates the principles of the Internet-of-Things by adopting the resource-oriented architectural style and semantic web standards to navigate product databases exposed on the Internet. By doing so, customers keep the control of their shopping process by selecting the stores and comparing the offers according to their own preferences.

The remainder of this paper is organized as follows. Section 2 introduces the challenges addressed by this paper, while Section 3 describes our contribution in terms of distributed infrastructure. Section 4 compares this contribution to the state-of-the-art, before concluding in Section 5

2 Motivations

The distribution of more and more powerful mobile devices and the emergence of the *Internet-of-Things* (IoT) raisewhich is representati a growing interest for the retail industry, which has to deal with a new generation of customers. Theses customers are characterized by a clear acquaintance to new technologies (Internet, smartphones, etc.) and a capacity to seamlessly switch between various sources and canals of distribution. In particular, shopping malls are more and more facing the competition of online stores since consumers can easily compare in-store product offers with online ones. Considering products as *things* that are exposed on the Internet is a raising concern for the retail industry, and vendors are more and more investing to properly advertise their products on the Internet. For the time being, this investment takes the form of product comparators that are proposed to customers by the chains in order to promote their offers. However, such applications are clearly not objective and cannot guarantee the best possible offer to the consumer. Furthermore, we believe that such an IoT can provide new categories of applications to better support the consumer in her/his shopping activities. Before detailing the challenges we identified in Section 2.2, we therefore describe a short scenario to illustrate a new generation of shopping system connected to the IoT in Section 2.1.

2.1 Scenario: Towards a New Generation of Shopping System

This section introduces a scenario, which is the representation of the expected usages of the shopping system. In this scenario, Nathalie uses her tablet-PC to browse recipes that are published on the website `cooking.com`. Once she made her choice, Nathalie wants to order all the ingredients that are needed to prepare the selected recipe. Nathalie therefore pastes the recipe URL within the MACCHIATO application and specifies the expected number of guests. MACCHIATO analyzes the content of the recipe and extracts the list of ingredients. Then, MACCHIATO computes the correct quantities according to the number of guests mentioned by Nathalie. In parallel, MACCHIATO queries *i)* an online folksonomy with the list of ingredients in order to infer equivalent terms, and *ii)* a directory service to identify the closest stores according to the current position of Nathalie. Then, MACCHIATO interrogates the surrounding stores with the enriched list of ingredients in order to retrieve a consolidated list of relevant products for her. MACCHIATO guides Nathalie in the process of selecting a specific product for each of the ingredients she needs.

Meanwhile, Nathalie's husband runs out of coffee pods, and before throwing the pods' bag into the trash, he scans the barcode as a reminder for buying new ones. This product immediately appears on the shopping cart that Nathalie is currently updating for the purpose of her recipe. The coffee pods are therefore seamlessly included in the comparison of offers triggered by MACCHIATO on behalf of Nathalie.

All the selected products are therefore grouped in the shopping cart of the family, which is then submitted by MACCHIATO to each store in order compute offers for the shopping cart. Nathalie therefore gets the opportunity to compare different offers and she finally decide to order all the products from the closest drive-in store. The product order, including the delivery preferences, is therefore automatically placed with the drive-in store by MACCHIATO. Nathalie is informed by MACCHIATO when and where she can pick up her products.

2.2 Challenges

Based on the above scenario description, we elaborate on the key challenges raised by such a system. In particular, we differentiate business challenges from more technical challenges.

Interoperability is a fundamental challenge to publish legacy systems on the Internet. Actually, information systems in e-commerce are compartmentalized, and it remains difficult to break the boundaries between heterogeneous sites in order to expose the products in a uniform way. Being able to integrate product offers from heterogeneous sources therefore requires to provide a versatile model for reasoning on products and matching consumer preferences.

Semantics is another challenge that a new generation of e-commerce platforms should exhibit. Beyond interoperability, it is also critical for items exposed by vendors to include enriched data that can be automatically processed by client applications. Leveraging semantics would therefore enable the development of smart services that can process and adapt the content available on Internet in order to bring it to the consumer.

Scalability is a critical challenge in our context since consumer traffic is naturally subject to strong variations. While some of these variations are predictable, like sales periods, some others are related to unexpected events, and therefore cannot be anticipated. The MACCHIATO system should therefore be able to support traffic peaks and to keep serving consumer requests with a reasonable quality of service. In particular, the MACCHIATO system should scale with regards to the number of concurrent consumers, the number of requests they emit, and the volume of data published by the stores.

3 Exposing Products as a Resource-Oriented Architecture

In MACCHIATO, products are considered as *things* (according to the IoT terminology) that are made available on the Internet. We therefore adopted a *Resource-Oriented Architecture* (ROA) to design a system that meets the challenges we introduced in Section 2.2, namely interoperability, semantics and scalability.

3.1 Architecture Overview

The MACCHIATO system processes data collected from heterogeneous vendors. For example, many stores expose their product catalog, the consumer must therefore be able to query and understand the data that comes from these different sources. While ROA styles, like *Representational State Transfer* (REST) [6], support standard representations for a given resource (*e.g.*, HTML, XML, JSON), we believe that IoT architectures should encourage the wide adoption of semantically-rich representations. By enriching resource representations with semantic descriptions, the client can benefit from typed information in order to seamlessly perform data alignments and conversions (*e.g.*, automatically converting prices from dollars to euros). We therefore choose to use the W3C semantic representation standard *Resource Description Framework* (RDF) [15] to expose semantically rich product representations. For example, this specification is already used by ProductDB [9] to expose the representations of 20,000 products. In addition to that, to publish and share e-commmerce resources in RDF, we need to agree to a common vocabulary. We therefore decided to reuse GoodRelations [7], a standard ontology for e-commerce, which is already adopted by companies such as Google, Yahoo!, BestBuy, or Sears. This vocabulary is described according to OWL recommendation [3] and it contains all the terms and concepts required to describe products and offers. Finally, to process the product representations, we use the SPARQL language [12], which is dedicated to query and navigate RDF documents.

While SPARQL queries can be communicated through the network using the SPARQL protocol [4], this solution tends to introduce performance bottlenecks and requires to invest in powerful server-side infrastructure to tolerate the request load. In MACCHIATO, we rather encourage to store SPARQL queries within the server and to expose these queries as REST resources, which can be can be accessed by client applications. This solution reduces the volume of data sent by applications to the server, avoids the execution of malicious queries and better tolerate request peaks by applying server-side optimization to the SPARQL queries. In addition to that, standard web intermediaries (*e.g.*, cache, proxy) can be applied to the requests exchanged on the network [13]. For example, any client application can request via HTTP a specific product to be retrieved as a semantically rich representation (*e.g.*, `text/turtle`, `application/rdf+xml`, `text/rdf+n3`). Such an HTTP request can easily be processed by web intermediaries before being delivered to the server-side infrastructure. If a semantic representation is requested, the associated SPARQL query is executed by the server to build an RDF graph describing the query result, and then serialize it according to the requested representation.

In MACCHIATO, the integration of ontologies can be achieved in two ways. First, legacy systems are supported by the deployment of specific gateways that reflect the product database as a semantically rich model. Second, using web frameworks, such a Forgeos[1], new generations of online stores can automatically expose their data as semantically rich REST resources.

[1] Forgeos: `http://www.forgeos.com`

3.2 Empirical validation

This section reports on different experiments we conducted in order to assess the server-side architecture we developed. The server infrastructure uses the FRASCATI [14] platform dedicated to the development of SCA applications. For the RDF resource manipulation, we use Apache Jena [10] and the SPARQL implementation ARQ. The resulting platform is hosted on single Xeon W3520 server with 16GB of memory running Ubuntu 11.10 amd64 with Java 1.6 and one instance of Apache Tomcat 7.0.

To evaluate the scalability of a REST/RDF service, we deployed an e-commerce service endpoint. This service exposes $76,915$ product details, price and delivery informations. This represents 75MB of RDF data in W3C N3 serialization. We use a representative scenario that simulates a consumer searching for 5 types of products. For each search result, the consumer queries for 5 products details. So, each consumer initiates 30 requests to retrieve search and product details. Consumers are simulated by the Gatling stress tool [2]. We increase gradually the load up to 450 concurrent customers on the server. In the initial configuration, we naively deploy this service and we observe, in Figure 1, that the response time is linearly bound to the number of consumers. Furthermore, the server fails when the load reaches about 500 concurrent customers.

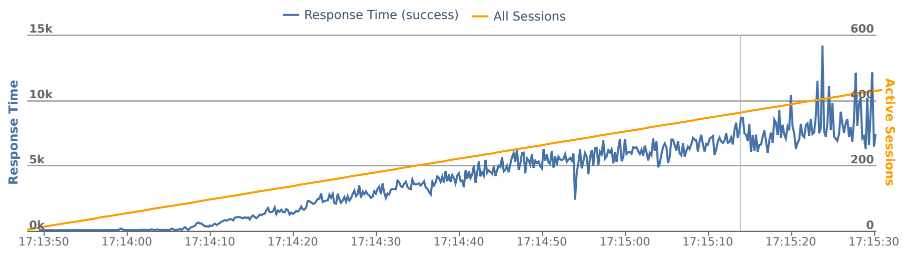

Fig. 1. Stress test of a MACCHIATO server.

In the second configuration, we include HTTP caching technology in the Tomcat server with *ehcache* [3] to demonstrate the benefits of web intermediaries. This choice is motivated by the observation that most of the requests (between 80 and 95% depending on vendors) received by e-commerce websites are read-only requests. Based on this statement, the deployment of a cache intermediary can be used to store the results of SPARQL queries and avoid to systematically trigger SPARQL computations, which would produce the same result. Products that are frequently requested are automatically stored in the cache and therefore quickly delivered to the customers. One can observe in Figure 2 that the response time goes slightly up when caching results, and then remains constant regardless of the number of customers. By adopting this organization, the server can therefore handle up to 7,000 requests per seconds, which makes

[2] Gatling Stress tool: `http://gatling-tool.org`
[3] Ehcache: `http://ehcache.org`

the adoption of RDF standards a sounding choice for implementing an IoT for the retail industry.

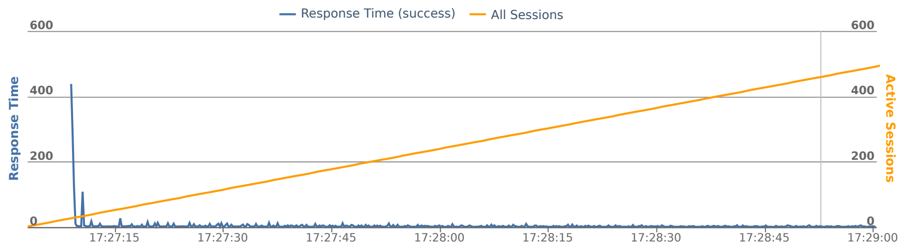

Fig. 2. Stress test of a MACCHIATO server with a caching intermediary.

3.3 Discussion & perspectives

With regards to the challenges we introduced in Section 2.2, we address the interoperability and semantics issues by adopting *i)* a REST architectural style to accommodate the client diversity and *ii)* RDF ontologies to share common vocabularies for exposing products, respectively. The scalability issue is tackled by the deployment of web intermediaries that can be used to reduce the resource-consuming computations and to improve user response time.

In the current solution we propose, most of the REST resource representations are the result of the execution of a SPARQL query on a remote RDF model. However, these queries are statically defined in the resource implementations. In order to accommodate the flexibility of the system, we are interested in supporting the dynamic deployment of SPARQL queries as REST resources. A mobile application could therefore post a SPARQL query to a server, which would host the query on behalf of all the client applications. The server would reply with the URL of the resource created with the attached query. The client could then query this resource to retrieve the results of the execution, or future updates. By adopting this approach, consumers can let long-running queries executing on server, and collect the results whenever needed. They could easily share these resources with other consumers and be notified of result evolutions, like the evolution of item prices.

4 Related work

Price engines. Many mobile applications already allow consumers to compare prices of products. LiveCompare [5] is an application to compare local prices of different products. It uses a combination of barcode decoding and GPS/GSM location to automate the detection of the product and the store location. However, the application only collects the pictures of the product tags. This means that the application mostly reports pictures to the user and is not able to provide advanced product comparisons. Furthermore, the proposed solution is based on contributions from users, which can results in reporting deprecated prices to users. The solution we promote is rather based on up-to-date product catalogs exposed by vendors. The exploitation of semantically rich product descriptions provides the foundations for supporting advanced product comparison not only based on the price of product, but also other properties, such as the nutrition facts labels or the carbon footprint.

Another approach described in the literature focuses on the decision assistance for the purchase process. *Will I Like It* [8] therefore analyzes consumer reviews to extract the most discriminating features of a given product and respective consumer opinions. This approach helps the consumer in choosing a particular product by exposing its discriminating factors. However, this approach mostly focus on consumer reviews and does not help in choosing the offer that better matches the consumer preferences, such as the delivery method, the location of the store, etc.

Finally, the UBIRA platform [1] tries to unify e-commerce and the brick-and-mortar stores. The proposed application allows the customer to switch from online to offline stores at each step of her/his shopping process. This approach helps the customer in locating the best offer from various online and offline sources, but does not help to choose between different products.

Interoperability. Interoperability is a critical challenge in the domain of distributed systems. Several solutions have already investigated the exploitation of ontologies to support interoperability. In particular, the ability to use RESTful services for interoperability of distributed systems has already been explored [2]. This solution proposes to create a SPARQL endpoint that query execution along multiple services. Performing the query division is achieved by a ontology mapping implemented in the endpoint. However, this solution requires an *a priori* knowledge on the ontologies used by different services, which does not make it a scalable and customizable solution. Furthermore, it does not offer any solution to improve the response time, which is a key criteria in such responsive systems.

CONNECT [11] proposes to use ontologies to support the dynamic interoperability of systems based on heterogeneous protocols. This approach infers ontology representatives message types of protocol in order to generate the connectors between these protocols. This allows for the discovery and adaptation of protocols at runtime. However, this is a very low-level approach that offers no solution to the alignment data. While our solution focus on application-level ontologies for e-commerce, we would like to investigate the solutions proposed by CONNECT to mine vendors which are not using the GoodRelations ontology and seamlessly connect them to the MACCHIATO infrastructure.

5 Conclusion

The emergence of mobile devices is deeply impacting consumption usages in the e-commerce domain. In particular, one can observe that the consumer can use more and more sources to make her/his choice. In order to help consumers to buy the products that fit their preferences, we need a new generation of e-commerce platforms, which have to tackle a variety of technical and functional challenges. To address these challenges, this paper reports on the design and the implementation of the MACCHIATO platform. To expose products, we propose a *Resource-Oriented Architecture* that exposes semantically rich representations of product catalogs.

In the future, we plan to work on more dynamic resource-oriented architectures. This service could allow user to deploy new resources from SPARQL queries. This will allow consumers to be alerted from complex resource updates, and to share information between consumers. We also plan to work on client application adaptation by exploring end-user programming technics in order to easily customize actor choregraphies.

References

1. Udana Bandara and James Chen. Ubira: a mobile platform for an integrated online/offline shopping experience. In James A Landay, Yuanchun Shi, Donald J Patterson, Yvonne Rogers, and Xing Xie, editors, *Ubicomp*, pages 547–548. ACM, 2011.
2. Robert Battle. Bridging the semantic Web and Web 2.0 with representational state transfer (REST). *, Services and Agents on the World Wide Web*, 2008.
3. Sean Bechhofer, Frank van Harmelen, Jim Hendler, Ian Horrocks, Deborah L. McGuinness, Peter F. Patel-Schneider, and Lynn Andrea Stein. OWL Web Ontology Language Reference, 2004. `http://www.w3.org/TR/owl-ref`.
4. Kendall Grant Clark, Lee Feigenbaum, and Elias Torres. SPARQL Protocol for RDF, 2008. `http://www.w3.org/TR/rdf--sparql--protocol`.
5. Linda Deng and LP Cox. Livecompare: grocery bargain hunting through participatory sensing. *Proceedings of the 10th workshop on Mobile*, 2009.
6. Roy T Fielding. *Architectural Styles and the Design of Network-based Software Architectures*. PhD thesis, University of California, Irvine, 2000.
7. Martin Hepp. Goodrelations: An ontology for describing products and services offers on the web. *Knowledge Engineering: Practice and Patterns*, pages 329–346, 2008.
8. Silviu Homoceanu, Michael Loster, Christoph Lofi, and Wolf-Tilo Balke. Will I Like It? Providing Product Overviews Based on Opinion Excerpts. *2011 IEEE 13th Conference on Commerce and Enterprise Computing*, pages 26–33, September 2011.
9. Ian Davis. ProductDB, 2012. `http://productdb.org`.
10. B McBride. Jena: a semantic Web toolkit. *Internet Computing, IEEE*, 6(6):55–59, 2002.
11. Vatsala Nundloll and Paul Grace. The role of ontologies in enabling dynamic interoperability. *Distributed Applications and Interoperable*, 2011.
12. Eric Prud'hommeaux and Andy Seaborne. SPARQL Query Language for RDF (Working Draft). Technical report, W3C, 2007.
13. Ulrich Scholten, Robin Fischer, and Christian Zirpins. Perspectives for Web Service Intermediaries: How Influence on Quality Makes the Difference. In *EC-Web*, volume 5692 of *LNCS*, pages 145–156. Springer, September 2009.
14. Lionel Seinturier, Philippe Merle, Damien Fournier, Nicolas Dolet, Valerio Schiavoni, and Jean-Bernard Stefani. Reconfigurable SCA Applications with the FraSCAti Platform. In *IEEE Int. Conf. on Services Computing*, 2009.
15. W3C. Resource Description Framework (RDF): Concepts and Abstract Syntax, 2004.