# TOWARDS FEDERATED CONSUMER PRODUCT SEARCH FROM HETEROGENEOUS SOURCES

Daniel Schuster, Maximilian Walther, Iris Braun
*Technische Universität Dresden, Department of Computer Science*
*Helmholtzstr. 10, 01062 Dresden, Germany*
*{schuster, walther, braun}@rn.inf.tu-dresden.de*

**ABSTRACT**

Federated search techniques improve search experience in a number of application areas, especially consumer product search. Existing systems for consumer product search often depend on one database only or federate multiple homogeneous information sources like Web Services of online shopping malls. We investigate possibilities for federated product search using heterogeneous sources like manufacturer websites and the Deep Web. The architecture and a Ruby on Rails prototype implementation are presented with emphasis on information extraction from semi-structured websites.

**KEYWORDS**

Federated product search, Web information extraction.

## 1. INTRODUCTION

Consumer product search may be cumbersome due to the number and heterogeneity of available information sources such as online shopping malls, vendor and manufacturer websites, product forums, and social commerce websites. Especially for extensive buying decisions, like buying a new digital camera, the consumer wants to use as many information sources as possible and extract and consolidate information from the different sources.

Federated search, i.e., simultaneous search in multiple information sources, would improve this process as the user only needs one browser window where all the information from different sources is presented. While this one window paradigm is much more convenient for the user, the federation furthermore enables access to information sources the user would not have been searching himself. Existing methods for federated search already offer a means to access multiple information sources of the same type like multiple search engines or multiple online shopping malls. But they do not yet involve semi-structured or unstructured sources.

We introduce the *Fedseeko* system, a research prototype built to realize such heterogeneous federated consumer product search. The system bridges the gap between information in online shopping malls and manufacturer information using information extraction (IE) methods to retrieve product information from manufacturer websites. The paper contributes a reference architecture as well as a method for information extraction from product websites and other semi-structured Web pages.

After a short overview of related work (Section 2), we discuss our architecture and method for information extraction in Sections 3 and 4. Section 5 contains an overview of the implementation of the Fedseeko system demonstrating the feasibility of our approach.

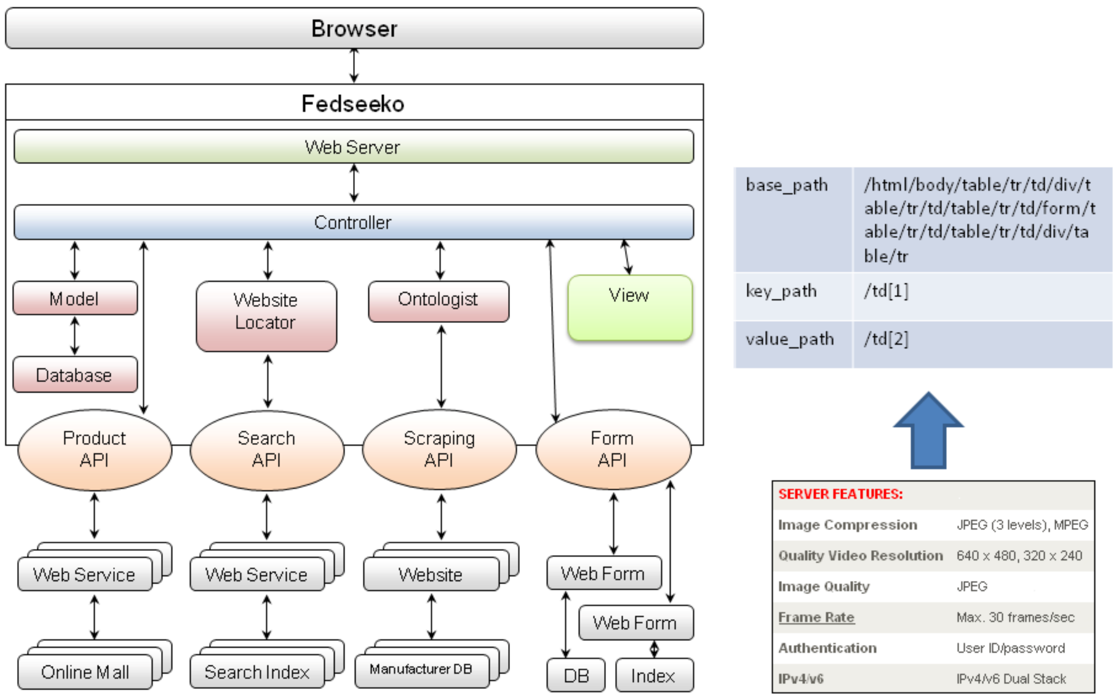
## 2. RELATED WORK

Federated product search is a typical use case for Semantic Web technologies and has thus already been considered in related work. (Kim et al., 2004) defines an architecture for federated search in multiple online shopping malls like amazon.com and buy.com using ontology mapping as a key technology. A semantic product query has to be generated by the user with the help of product categories. A comparable approach is the shopinfo.xml (Wolter et al., 2006) standard. Shop operators may define a shopinfo.xml that can be downloaded by the shopping portal easily providing both, RESTful Web Services as well as downloading an XML product file for shop federation. Shopbots (Fasli, 2006) are another alternative to search for products from different vendors based on Web scraping or Web Services.

Our approach differs from these works, as we extend the federated product search from integration of different shops to federation with heterogeneous sources like Web search engines, product websites, and additional websites with a search form. As part of these information sources do not provide Web Service interfaces, we introduce specific Web information extraction techniques for this purpose.

## 3. ARCHITECTURE

Our architecture extends the well-known Model View Controller (MVC) pattern towards a federated search MVC as shown in Figure 1. The Web browser is the central access medium communicating with a Web server. As the central access point, the Controller coordinates Model and View and contains the main application logic.



| | |
|---|---|
| base_path | /html/body/table/tr/td/div/table/tr/td/table/tr/td/form/table/tr/td/table/tr/td/div/table/tr |
| key_path | /td[1] |
| value_path | /td[2] |

| SERVER FEATURES: | |
|---|---|
| Image Compression | JPEG (3 levels), MPEG |
| Quality Video Resolution | 640 x 480, 320 x 240 |
| Image Quality | JPEG |
| Frame Rate | Max. 30 frames/sec |
| Authentication | User ID/password |
| IPv4/v6 | IPv4/v6 Dual Stack |

**Figure 1: Architecture and IE example**

The Model layer is split into internal and external information. Internal information resides in a database that is represented by an internal object model. External information is accessed via a set of APIs to the following external sources:

1. Online shopping malls - structured information from online shopping malls is used as primary information. They are accessed using Web Service interfaces like the Amazon Associates Web Service (AAWS) (Amazon Inc., 2008).
2. Web search engines - structured information from search engine indices is retrieved using search engine APIs provided as Web Services. The Website Locator uses search engines to find external websites with additional information about certain products.
3. Manufacturer websites – the Scraping API provides a means to extract information from so-called product detail pages, where information about certain product features are given within a table or list. The Ontologist integrates this information in the internal object model using ontologies.

4. Third-party websites – the Form API allows access to Web pages of third parties such as product communities. These pages are often part of the so-called Deep Web (Bergmann, 2001).

## 4. INFORMATION EXTRACTION

Extracting and integrating information from manufacturer websites is one of the key elements of our approach. In a first step, the Search API is used to find the product detail page of a selected product. The product name as returned by the shopping mall Web Service often contains additional information and cannot be used directly as a search query to find this page. Thus the algorithm first looks for the manufacturer website and then focuses the search on the manufacturer domain. The product name is split up into terms and one term is removed at the end of the string in each cycle until the search engine returns a hit.

In a second step, the product detail page is analyzed with the Scraping API. As can be seen in Figure 1 (right part), product details are often provided in HTML tables with a layout consistent throughout all Web pages of the same manufacturer. More generally speaking, a well-defined transformation process takes structured data from the database as input and generates semi-structured HTML pages. The idea of our Scraping API is to invert this process to retrieve structured product data from semi-structured HTML.

The right part of Figure 1 shows a detail view of a product detail page of a Panasonic network camera. As can be seen, keys and values of structured product data are arranged in an HTML table. Within the Scraping API, the XPath of such product data in HTML pages is used as information extraction descriptor. A user has to copy an example data set such as “Image Quality | JPEG” from the product detail page to an input field and the IE method generates XPaths for keys and values of product data within the HTML table. Once this process has been done, information from all product detail pages of the manufacturer can be extracted without user interaction.

The next step in the IE process is matching this structured information to the portal terminology. An ontology is created for this purpose that matches product attributes with attributes from the portal. Presently, one mapping ontology has to be created for each manufacturer. This can be done by the manufacturer or by any third person. We plan to investigate strategies for ontology matching to automate this process.

## 5. IMPLEMENTATION

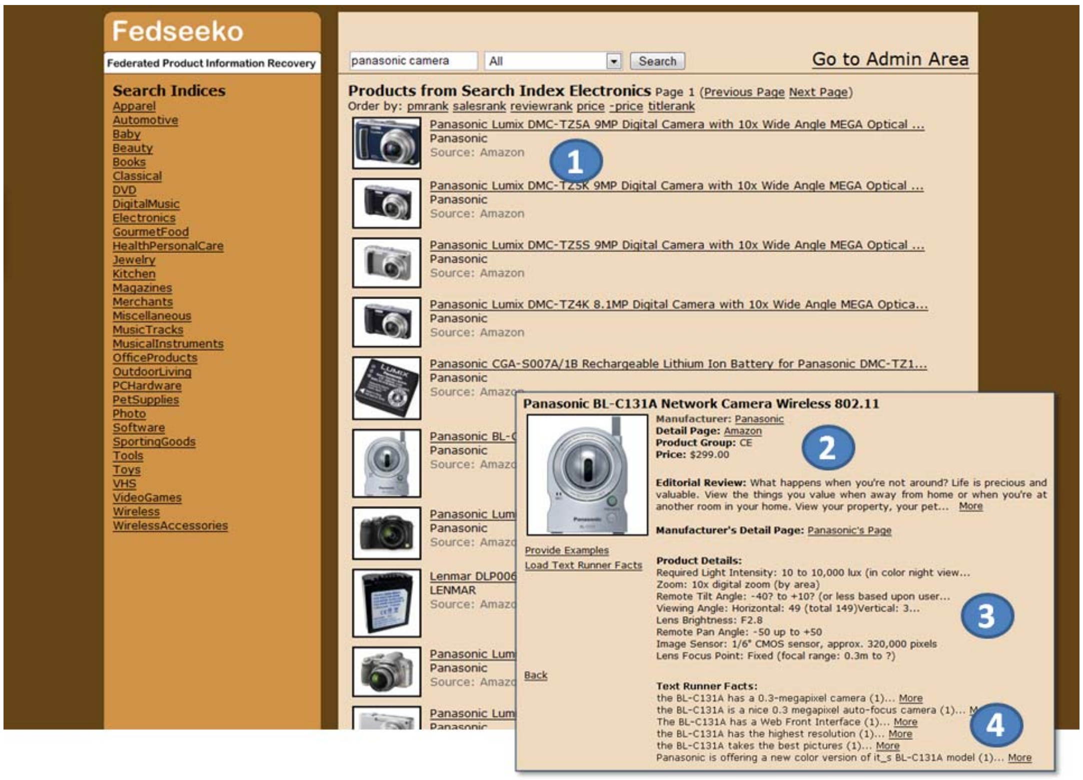
We implemented the approach as an Ajax-based Web application using the Ruby on Rails framework (Hansson, 2008) for agile Web development. Using the JRuby interpreter, it is possible to embed Java code into Ruby objects and thus be able to integrate Java frameworks, too. We used the following extensions to standard Ruby: the Hpricot HTML parser, the htree HTML/XML library, the ruby-aws library for access to the Amazon Associates Web Service, the Jena Semantic Web Framework (Jena, 2008) and the scRUBYt (scRUBYt, 2008) Web scraping toolkit.

The screenshot in Figure 2 shows the user interface of the Fedseeko system. A search form as well as a list of categories can be used to browse products (1). Product information is retrieved from Amazon using the AWS interface. Once the user selects a product, a detailed view is shown (2-4) with information from different sources: Amazon product details (2), additional product details from the manufacturer website (3), and facts from the Web about the product (4) using Textrunner (Banko et al., 2007). As locating external product pages and extracting information from them may last several seconds, the website uses Ajax technology to build pages incrementally. So the user gets detailed product information (from the AWS interface) immediately. This information is complemented later by information from product detail pages and Textrunner facts.

## 6. CONCLUSION

We have introduced the Fedseeko system, a research prototype for federated consumer product search. The design of the prototype was inspired by the need to integrate information from structured as well as semi-structured and unstructured sources to improve the search process. A general architecture and method for information extraction from manufacturer Web pages was shown.

While the results achieved so far already seem to be promising, there still remain some challenges. Integration of semi-structured information currently relies on user interaction and the availability of product ontologies. Removing these prerequisites is an essential next research step. Furthermore, unstructured information is yet only presented "as is" to the user. Extracting structured information out of this information using natural language processing will complete the federated search. Furthermore, we plan to advance and evaluate the IE methods introduced in this paper.



**Search Indices**
Apparel
Automotive
Baby
Beauty
Books
Classical
DVD
DigitalMusic
Electronics
GourmetFood
HealthPersonalCare
Jewelry
Kitchen
Magazines
Merchants
Miscellaneous
MusicTracks
MusicalInstruments
OfficeProducts
OutdoorLiving
PCHardware
PetSupplies
Photo
Software
SportingGoods
Tools
Toys
VHS
VideoGames
Wireless
WirelessAccessories

**Products from Search Index Electronics** Page 1 (Previous Page Next Page)
Order by: pmrank salesrank reviewrank price -price titlerank

Panasonic Lumix DMC-TZ5A 9MP Digital Camera with 10x Wide Angle MEGA Optical ...
Panasonic
Source: Amazon

Panasonic Lumix DMC-TZ5K 9MP Digital Camera with 10x Wide Angle MEGA Optical ...
Panasonic
Source: Amazon

Panasonic Lumix DMC-TZ5S 9MP Digital Camera with 10x Wide Angle MEGA Optical ...
Panasonic
Source: Amazon

Panasonic Lumix DMC-TZ4K 8.1MP Digital Camera with 10x Wide Angle MEGA Optica...
Panasonic
Source: Amazon

Panasonic CGA-S007A/1B Rechargeable Lithium Ion Battery for Panasonic DMC-TZ1...
Panasonic
Source: Amazon

Panasonic BL-C
Panasonic
Source: Amazo

Panasonic Lum
Panasonic
Source: Amazo

Lenmar DLP006
LENMAR
Source: Amazo

Panasonic Lum
Panasonic
Source: Amazo

Panasonic Lum
Panasonic

**Panasonic BL-C131A Network Camera Wireless 802.11**
**Manufacturer:** Panasonic
**Detail Page:** Amazon
**Product Group:** CE
**Price:** $299.00

**Editorial Review:** What happens when you're not around? Life is precious and valuable. View the things you value when away from home or when you're at another room in your home. View your property, your pet... More

**Manufacturer's Detail Page:** Panasonic's Page

Provide Examples
Load Text Runner Facts

**Product Details:**
Required Light Intensity: 10 to 10,000 lux (in color night view...
Zoom: 10x digital zoom (by area)
Remote Tilt Angle: -40? to +10? (or less based upon user...
Viewing Angle: Horizontal: 49 (total 149)Vertical: 3...
Lens Brightness: F2.8
Remote Pan Angle: -50 up to +50
Image Sensor: 1/6" CMOS sensor, approx. 320,000 pixels
Lens Focus Point: Fixed (focal range: 0.3m to ?)

Back

**Text Runner Facts:**
the BL-C131A has a 0.3-megapixel camera (1)... More
the BL-C131A is a nice 0.3 megapixel auto-focus camera (1)... M
The BL-C131A has a Web Front Interface (1)... More
the BL-C131A has the highest resolution (1)... More
the BL-C131A takes the best pictures (1)... More
Panasonic is offering a new color version of it_s BL-C131A model (1)... More

**Figure 2: Screenshots Fedseeko**

## REFERENCES

Amazon Inc., 2008. *Amazon Web Services @ Amazon.com*, http://www.amazon.com/E-Commerce-Service-AWS-home-page/b?node=12738641.

Banko, M., Cafarella, M. J., Soderl, S., Broadhead, M., Etzioni, O., 2007. *Open information extraction from the Web*, Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI), Hyderabad, India.

Bergmann, M. K., 2001. *The Deep Web: Surfacing Hidden Value*, The Journal of Electronic Publishing, vol. 7, no. 1.

Fasli, M., 2006. *Shopbots: A Syntactic Present, A Semantic Future*, IEEE Internet Computing, vol. 10, no. 6, pp. 69-75.

Hansson, D.H., 2008. *Ruby on Rails*, http://www.rubyonrails.org/.

Jena, 2008. *Jena – A Semantic Web Framework for Java*, http://jena.sourceforge.net/.

Kim, W., Choi, D.W. and Park, S., 2004. *Intelligent Product Information Search Framework Based on the Semantic Web*, 3rd International Semantic Web Conference (ISWC), Hiroshima, Japan.

scRUBYt!, 2008. *scRUBYt! - a Simple to Learn and Use, yet Powerful Web Scraping Toolkit Written in Ruby*, http://scrubyt.org/.

Wolter, T., Schumacher, J., Matschi, M. and Kuhlins, S., 2006. *Der shopinfo.xml-Standard: Datenintegration zwischen Shops und Shopping-Portalen*, XML-Tage, Berlin, Germany.