

“Use Cases Deployment”

D8.2

‘’

Version: 1.0

Last Update: 22/6/2012 13:06:00 20126/P6

Distribution Level: PU (Report), RE (Prototype)

* ***Distribution level***

*PU = Public,*

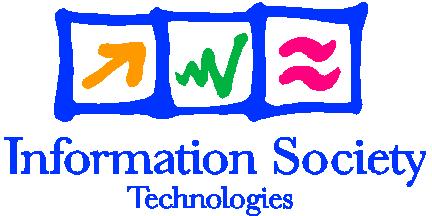
*RE = Restricted to a group of the specified Consortium,*

*PP = Restricted to other program participants (including Commission Services),*

***CO= Confidential, only for members of the I-SEARCH Consortium (including the Commission Services)***

***The research leading to these results has received funding from the European Community's Seventh Framework***

***Programme (FP7/2007-2013) under grant agreement n° 248296***

fp7.jpg

**The I-SEARCH Project Consortium groups the following organizations:**

|  |  |  |
| --- | --- | --- |
| **Partner Name** | **Short name** | **Country** |
| Centre for Research and Technology Hellas / Informatics and Telematics Institute | CERTH | GR |
| JCP-Consult | JCP | FR |
| Institut National de Recherche en Informatique | INRIA | FR |
| Athens Technology Center | ATC | GR |
| Engineering Ingegneria Informatica S.p.A. | ENG | IT |
| Google Ireland Limited | Google | IR |
| University of Genoa | UNIGE | IT |
| Exalead | EXL | FR |
| Fulda University of Applied Sciences | HSF | DE |
| Accademia Nazionale di Santa Cecilia | ANSC | IT |
| EasternGraphics GmbH | EGR | DE |

**Executive Summary:** In this deliverable, we report on work around the I-SEARCH EU (FP7 ICT STREP) project whose objective is the development of a multimodal search engine. We present the project's objectives, and detail the achieved results, amongst which a Rich Unified Content Description format.

Document Identity

|  |  |
| --- | --- |
| Title: | I-SEARCH user interfaces for PCs and mobile devices |
| Subject: | Description of user interface challenges and solutions for multimodal interaction with different classes of devices |
| Number: |  |
| File name: | WP 7.3 Google.docx |
| Registration Date: | 20 June, 2012 |
| Last Update: | 22 June 2012 |
| Authors: | GOOGLE, HSF, CERTH |

Revision History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Version | Edition | Author(s) | Date |
| 1 | 0.1 | 1 | Thomas Steiner | 20/06/2012 |
| Comments: | Initial release | | |
| 2 |  |  |  |  |
| Comments: |  | | |
| 3 |  |  |  |  |
| Comments: |  | | |
| 4 |  |  |  |  |
| Comments: |  | | |
| 5 |  |  |  |  |
| Comments: |  | | |
| 6 |  |  |  |  |
| Comments: |  | | |
| 7 |  |  |  |  |
| Comments: |  | | |
| 8 |  |  |  |  |
| Comments: |  | | |
| 9 |  |  |  |  |
| Comments: |  | | |

**Table of Content**

1. Introduction 7

2. Rich Unified Content Description (*RUCoD*) 7

3. *CoFetch*—A Tool for the Creation of Content Objects 7

4. Use Cases 9

4.1. UC1: Chiara, the italian musicologist 9

4.2. UC2: Party 9

4.3. UC3: Yannis the Architect 10

4.4. UC4: Sophia, smartphone user 10

4.5. UC5: Roman, motorcycle buyer 10

4.6. UC6: Patrick, the game developer 11

4.7. UC7: Toni the gamer 12

5. I-SEARCH Demonstration Portal 12

6. References 13

**List of Figures**

Figure 1: The CoFetch tool uses among others data from the Linking Open Data cloud (http://lod-cloud.net/) for the generation of content objects. 8

Figure 2: I-SEARCH graphical user interface on a mobile device 12

**List of Abbreviations and acronyms**

(in alphabetic order)

|  |  |
| --- | --- |
| **Abbreviation** | **Explanation** |
| API | Application Programmable Interface |
| CoFind | Collaborative Search |
| GUI | Graphical User Interface |
| HTML5 | Hypertext Mark-up Language 5 |
| pTag | Personalized Tagging Service |
| RUCoD | Rich Unified Content Description |
| UI | User Interface |
| UIIFace | Unified Interaction Interface |
| W3C | World Wide Web Consortium |
| XML | Extensible Mark-up Language |
| XMI | XML Metadata Interchange |
| SDK | Software Development Kit |
| URL | Unified Resource Identifier |
| CO | Content Object |
| DOM | Document Object Model |
| JSGF | Java Speech Grammar Format |
| REST | REpresentational State Transfer |

# Introduction

Search engines over the years have coined a common interaction pattern: the search box. We enhance this interaction pattern with context-aware modality input toggles that create modality query tokens in the I-SEARCH search box[1](eswc20122.html#fn1x0) . The concept-centered I-SEARCH search engine lets users search by using different modalities and combinations of modalities. Supported modalities are audio, video, rhythm, image, 3D object, sketch, emotion, social signals, geolocation, and text. For example, in order to get a holistic view of the concept of Leonardo da Vinci’s painting of La Joconde, a user could search via a photograph of the painting, and—besides the textual description—also get back the exact geolocation of the painting in the Louvre museum, a video of a relevant section of a Louvre tour, or an audio recording of an explanation of the painting by a museum guide.

# Rich Unified Content Description (RUCoD)

In order to allow for the before-mentioned concept-centeredness of I-SEARCH, in the context of the project we have first introduced the notion of so-called content objects, and second, a description format named Rich Unified Content Description (RUCoD) [1]. Content objects are rich media presentations, enclosing different types of media, along with real-world information and user-related information. RUCoD provides a uniform descriptor for all types of content objects, irrespective of the underlying media and accompanying information. In the next section, we will introduce a tool for the creation of such content objects.

# CoFetch—A Tool for the Creation of Content Objects

CoFetch is a content object creation tool that can be operated in a fully automatic, or in a semi-supervised way. In order to generate content objects for a given term- and category-based query, it harvests textual and image data from DBpedia via the DBpedia Lookup service [4], 3D models from Google 3D Warehouse [2], images from Flickr [7], videos from YouTube [3], audios from Freesound [5], and historical weather data from Weather Underground [6]. In order to decide on the most relevant results from each service, a Levenshtein distance function based on the titles and descriptions is used. First, we generate a list of occurrences of the query terms within the results’ textual data. Second, we generate a list of distances between query and result titles. Third, the end result with the two most relevant items is determined. Finally, a descriptor file with extracted RUCoD features and pointers to all media items is generated.

(a) Architecture of the CoFetch tool.

(b) The CoFetch tool’s GUI.

Figure 1: The CoFetch tool uses among others data from the Linking Open Data cloud (<http://lod-cloud.net/>) for the generation of content objects.

# Use Cases

## UC1: Chiara, the italian musicologist

Chiara is a musicologist that has access to the archive of ethnomusicology of ANSC. As part of her job, she wants to gather material showing how the pattern made famous by Ravel in the Bolero piece was already in use in popular music.

After opening a session in the I-SEARCH system, she taps the rhythm on the microphone of the terminal, and submits it to the system. I-SEARCH returns a list of matches; at the top of the list, the recording of a concert lies, where the Bolero piece was played. However, in the top positions there is also a drum recording from Italian ethno musicological repertoire, the “Ritmo di tamburo” from Raccolta 24 (eg. Piece 119 at around 00:30, piece 124 at around 00:08) where various drum rhythms are played, sharing indeed the same triplet of Bolero. A little further away, she also finds a voice recording.

By adding more search items, e.g. the geographical position, or by playing a virtual drum with her hands, she can further reduce the size of the result content set in order to locate the correspondence she was looking for.

## UC2: Party

A group of friends are having a party in a private house, and would like to listen and dance on the rhythm of the music they like, but nobody wants to spend the evening behind a DJ console. Luckily enough one of them has subscribed to the I-SEARCH service, and, after logging in the system with his credentials, performs an initial query using the name of their favorite pop singer.

The video clip of the latest success of the artist is immediately displayed on the screen and the friends start dancing. It doesn’t take long before the dancers begin to respond to each others moves, synchronizing their movements with the movements of the person that seems to know what move to make at the right time.

Meanwhile the I-SEARCH system is monitoring the group of dancers; it has spotted who is the leader, and by examining his moves it finds out that his movements are much more fluid and smooth than the music would require. So it automatically submit a new query to the system, asking for a different set of songs from the same singer but with a happier tone that would better match the mood of the dancers.

The new song is cross faded with the old one, and the group of friends welcomes the choice by adapting to the new rhythm. A new leader emerges from the group, a person that likes slower rhythms and knows the right moves for these ones.

However, the group doesn’t seem to enjoy too much the new music, so someone stops dancing while someone else starts clapping his hands to give a new rhythm on which the dancers should align. The I-SEARCH system captures the rhythm and submits a query to retrieve a list of songs sharing the same pattern and the energetic features of the group; but as the similarity index between the currently played song and the song requested by the user is very low and the transition would be very sharp, it resubmits a new query to the system containing the energetic and emotional features of the group, the requested rhythmic pattern and the low-level descriptors of a virtual song obtained by mixing the descriptors of the two songs.

The result is a song from a different singer having a rhythm close enough to the requested one, but still similar to the song that was previously played; even if this is the first time the friends hear it, they like it and start dancing all together, while on the screen are displayed in sequence the cover of the album and images of the artist during a concert that was held a few months before in their city.

## UC3: Yannis the Architect

Yannis is an architect and has to create for his customer a new office planning. To make the solution more realistic, he needs furniture that the customer can get in the reality. He uses the I-SEARCH interface to submit an image (photo of upholstery, rendering of his 3D model he used in his non-customized room plan) and dimensions of his planned area for the office chair by text. Alternative queries are also the sketch (or drawing) of the furniture or even similar 3D objects that Yannis has on his local disk.

After submitting the query to the I-SEARCH system, Yannis retrieves relevant furniture items that are available in multiple media types: 3D models, images, photos, drawings, videos. He, selects the most relevant 3D objects and imports them to the planning. Additionally, he exploits his customer’s location (through the real-world-based search functionality of I-SEARCH) in order to find manufacturers whose deliver dealers closer to the customer’s place to minimize the delivery costs.

Through the search functionality, I-SEARCH offers the users access to the product catalogues, which are provided by EGR. The scenario envisages that the user performs his/her search through an online-application, using alternative end user devices, including mobile phones with multimedia support. Furthermore, a scanner device is foreseen, which will enable the end users for hand-drawn sketches and picture drawings through a computer graphic program.

Typical search and retrieval tasks include the query for identical, similar or related articles, the search for realistic articles on the base of an imaginary sketch-input, the search for single, extension or replacement parts and cross-modal search (e. g. sketch-to-3D).

## UC4: Sophia, smartphone user

Sofia has a new smartphone with the I-SEARCH client application installed. Through her smartphone, she is able to search for multimedia content at every place: at work, at shopping or even on vacation. She can simply take a photo from the smartphone’s digital camera and use it as query to the I-SEARCH application to search for similar items.

At shopping, she can take a photo of a chair, which looks like the chair she wants to buy, and perform a similarity-based search for 3D furniture models or images. In case most of the retrieved items are linked to dealers located far from her place, she enhances the search criteria by adding her location information (GPS) to the query interface. Now, the retrieved results consist of furniture items, which are available from furniture suppliers close to her.

When sightseeing, she can take photos (or video) of a cultural heritage item and search for additional information (textual information) or other similar items (3D models, images or videos). She may also use real-world (location and time information) to search for cultural items related to a specific time period or place. In case the results are not satisfying, she is able to refine the query using the relevance feedback functionality of I-SEARCH.

This use case can be generalized in order to include a wider variety of application areas (except for furniture and cultural heritage content).

## UC5: Roman, motorcycle buyer

Roman wants to buy a (very special) motorcycle, however, he does not have a clear image of it. He has seen many motorcycles and he has quite clear preferences regarding particular parts of motorcycle, like the geometrical shape of fuel tank, the shape of exhaust pipes, the sound of the engine etc.

There is a specialized website, based on the I-SEARCH technology, that helps bikers to search most appropriate motorcycle models (close to bikers imagination) and also give dealer contacts that sells such motorcycles.

Thus, Roman uses several possibilities, to find a motorcycle very close to his imagination:

Picture query: he upload pictures of different motorcycles, that he likes, or chooses among a set of picture already provided on the website

Video query: he previews short clips of videos, provided on the website and chooses 2-3 to use as query

Hand drawn sketch query: he uploads a couple of hand-drawn sketches of fuel tank shape

3D model query: he chooses a couple of 3D models of exhaust pipes and handlebars (from the set provided in the website)

Audio query: he listens to different sounds of motorcycle engines (from the set provided on the website) and chooses the one or two that he likes more

After he creates the multimodal query, he retrieves a couple of motorcycles that best match the search criteria. Search results are: Pictures, Videos and flash animations (where Roman can make 3D view of motorcycles). To his surprise, one of them looks even better than he had imagined so he decides to buy it.

A list of dealers close to Roman’s geographic location is given. Additionally, more information about the bike is given and, finally, Roman picks the dealer that offers the best price.

## UC6: Patrick, the game developer

Patrick created a pick-up truck in his 3D modeling software.

He uploads his 3D object to I-SEARCH and wants to find other 3D objects of vehicles which are related to specific attributes (e.g. shape and size) of his pick-up truck. The I-SEARCH system uses the inserted 3D object to extract low-level descriptors from the content and create a query based on the RUCoD format. The query retrieves similar multimodal results which are prepared for their user representation via visual analytics techniques and returned to Patrick’s computer.

I-SEARCH presents the result as groups (e.g. created by the modality of the search results) and tagged with categories derived from previous searches of him and other users. The most characteristic item of a group is used as the representative of the group. General group information (e.g. modality type, annotation, population size, etc.) combined with each group representative support the understanding of the grouping in an intuitive manner.

He picks the group which contains the 3D object results and starts to browse through them. Because the results in a group are visual organized in multiple-level hierarchies, Patrick is able to navigate through the different levels and can observe any particular subgroup of the results. Throughout the navigation process he can make use of his multimodal input devices.

Patrick utilizes the relevance feedback option and marks the most appropriate 3D models of the result set as relevant.

Soon the result set refines according to the feedback he gave showing many very suitable pick-up trucks.

Patrick picks a suitable model of a car and so he selects the preview of the model to display more information of the model.

Patrick obtains the car model via the offered options

## UC7: Toni the gamer

Toni is searching for a new Ninja fighter avatar. In order to achieve that he makes a screenshot of the Ninja avatar from his game and uses the screenshot as query input for I-SEARCH.

I-SEARCH examines the low-level descriptors of the screenshot and returns the multimodal search results to Toni.

Toni navigates through the prepared visual organization of the results by using his preferred input method (e.g. mouse, touch, gestures or speech commands).

He chooses 3D avatars from the result set and utilizes the relevance feedback mechanisms to read the annotations from other users and individually rank the found avatars.

He stores his search query for later use or sharing of the results.

Finally an appropriate avatar for his game is chosen and imported in the game.

# I-SEARCH Demonstration Portal

The I-SEARCH demonstration portal is available at the URL <http://isearch.ai.fh-erfurt.de/>, hosted by the project partner Hochschule Fulda (HSF), formerly Fachhochschule Erfort (FHE). The code is hosted on a shared GitHub repository at the URL <https://github.com/tomayac/isearch>. Changes to the repository are directly reflected on the demonstration portal.

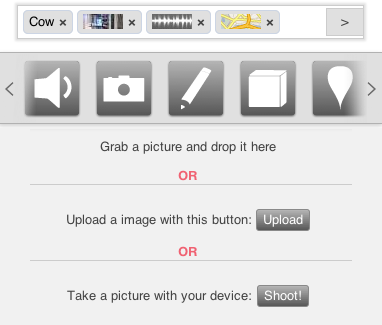


Figure 2: I-SEARCH graphical user interface on a mobile device

The code of the project follows the Google JavaScript guidelines, available at the URL <http://google-styleguide.googlecode.com/svn/trunk/javascriptguide.xml>. The I-SEARCH graphical user interface source code is separated into several modules. For the dependency management, we use RequireJS, which is a JavaScript file and module loader. It is optimized for in-browser use, but it can be used in other JavaScript environments, like Rhino and Node. It is available at the URL <http://requirejs.org/>. The graphical user interface requires a server-side component for session management, collaborative search, etc. Therefore, a node.js server has been installed at HSF. Node.js is a software system designed for writing scalable Internet applications, notably web servers. Programs are written in JavaScript, using event-driven, asynchronous I/O to minimize overhead and maximize scalability.

# References

[1] Daras, P., Axenopoulos, A., Darlagiannis, V., Tzovaras, D., Le Bourdon, X., Joyeux, L., et al.: Introducing a Unified Framework for Content Object Description. Inter- national Journal of Multimedia Intelligence and Security (IJMIS) 2, 351–375 (2011)

[2] Google: 3D Warehouse (2012), <http://sketchup.google.com/3dwarehouse>

[3] Google: YouTube API (2012), <http://code.google.com/apis/youtube/>

[4] Kobilarov, G.: DBpedia Lookup Service (2012), <http://lookup.dbpedia.org/>

[5] Music Technology Group: Freesound API (2012), <http://tabasco.upf.edu/>

[6] Weather Underground: Weather API (2012), <http://www.wunderground.com/>

[7] Yahoo!: Flickr API (2012), <http://www.flickr.com/services/api>

---End of Document ---