Proceedings of the Federated Conference on
Computer Science and Information Systems pp. 911–914
ISBN 978-83-60810-51-4

# Implementing Ubiquitous Information Services with Ontologies: Methodology and Case Study

Alfio Costanzo[1], Alberto Faro[1], Daniela Giordano[1], Concetto Spampinato[1]
[1] *Department of Electrical, Electronics and Computer Engineering,*
*University of Catania, viale A. Doria 6,*
*Catania, 95125, Italy*

***Abstract*—Currently, data integration in Ubiquitous Information Systems (UISs) is achieved by a centralized relational database where all the information coming from the remote servers is stored. To improve such architecture we proposed to convert in each server the relational archives into an RDF triple stores where data are represented by standard terms interrelated by subject-predicate-object relations (also called data ontology). In this way, UISs could be developed by a distributed approach, where data integration is achieved by an shared RDF format although data used on the remote servers follow a proprietary format. Also, the paper proposes to implement such UISs per use stories, defined according to task ontologies, to facilitate implementation from specifications. A case study illustrates how software environments based on the paradigm Model-View-Controller, e.g., Ruby on Rails powered by JQMobile, facilitate the implementation of the methodology.**

***Index Terms*—ubiquitous information services, data ontology, task ontology, distributed systems.**

## I. INTRODUCTION

Ubiquitous information systems (UISs) in smart cities should help mobile users to take right decisions in time-varying urban/metropolitan environments. Therefore, they should be provided with sensing infrastructures to measure the conditions of the current traffic, weather or pollution, and with location rules to identify where are the relevant services [1], and how they can be obtained, possibly taking into account personal data that may influence the user context awareness such as user age, health status, and so on [2].

Currently, data integration is achieved by a centralized urban database at the core of a large urban data warehouse where all the information coming from the remote servers is stored on MySQL, Oracle or DB2 tables and processed by suitable intelligent algorithms to provide the required services for the users [3].

In this centralized architecture, privacy, updating policies and data management are carried out by proprietary solutions running on the proper machines, whereas data integration is obtained by transforming the proprietary datasets in relational archives on the main server, where they can be queried jointly to provide the ubiquitous mobile services.

In [4] we have proposed to improve such an architecture by converting the relational archives into RDF triple stores [5] (also called RDF ontologies) where data are represented by standard terms interrelated by subject-predicate-object relations depending on the application domain. In this way, ubiquitous applications could be developed independently on the technology used to collect the data and on how the data are formatted on the various specialized servers. However, both the above relational and RDF centralized architectures allows us to integrate all the urban datasets at condition that they are copied in a single data warehouse, resident on the central server, to be used in all the user transactions. Thus, a transition from a centralized to a distributed UIS is envisaged to achieve higher reliability and time performance. In this distributed scenario, the mobiles may host data and rules in RDF format dealing with the user personal sphere, whereas the urban data on traffic, weather and pollution, and the relevant administrative records are stored in RDF archives resident on the proper servers.

In this scenario, data retrieval may be obtained by distributed queries activated by the mobiles without the main server intervention, as well as data presentation may be carried out autonomously by the user devices, e.g., by running the relevant Google APIs on the user mobiles. The main server could carry out typical network operations to manage distributed databases such as detecting and repairing DBs featured by low response times or wrong information, providing the users with the directories of available city DBs, ensuring that these DBs are used only by authorized users, and managing the utilization times and costs.

However, the authors have expressed some criticisms to the above data driven approach to service design claiming that a suitable service interface is also important to improve usability because, according to the *situated cognition* paradigm [6], the user activities develop following task ontologies. Thus, we have suggested in [7] to organize the user interactions per task ontologies, also called *use stories*, pointing out the importance of the user interface [8].

As a consequence, in both the main server and the mobiles, the services should be structured per use stories, possibly following the design paradigm called Model-View-Controller (MVC) [9]. Indeed, in MVC the use stories may be implemented by a set of *controllers,* each grouping the actions pertaining to a story, the scenarios of the stories by *views*, whereas data are *modeled* as objects although they may be physically organized in MySQL tables or XML files.

For this reason, aim of the paper is to propose that the implementation of the ubiquitous information services will be based on *data ontologies* to facilitate the integration of the distributed DBs containing sensed and business information needed to accomplish the actions of the mobile user, whereas *task ontologies* based design is envisaged to support the right actions affordable by the users depending on their location and context, thus improving usability [10].

Fig.1 shows the functional architecture we have in mind to offer real time ubiquitous information services on the basis of the data currently stored on distributed DBs.

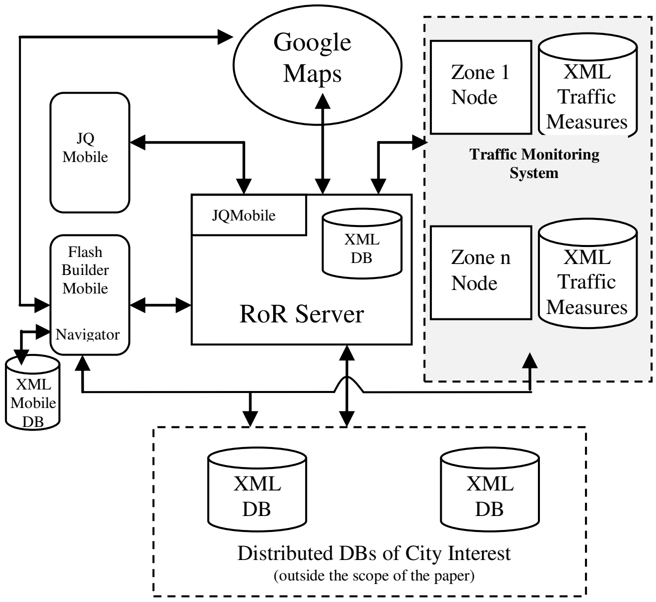

Figure 1. Traffic information system based on real time information

In such architecture we point out two environments that help data integration and are able to offer the required services per use stories, i.e.: a) the server Ruby on Rails (RoR) [11], following the MVC paradigm, whose controllers implement the use stories of the mobile users, whereas the views are JQMobile scripts [12] implementing for each story the most suitable scenario by an user interface based on the familiar Google Maps; and b) the Flash Builder based software [13] resident on the user mobiles to provide the users with similar RoR views, but saving RoR time.

Let us note that that the above architecture is suitable to manage both sensed and business data. To this aim, fig.1 points out how the data on car flow, density and velocity taken from cameras, e.g., [14], [15], [16], and [17], or other in situ technologies [18] dealing with the traffic of city zones are stored on the relevant monitoring nodes in XML formats. The integration of such XML data is done by an RoR server, where the data are used to compute the best paths to destination or the best circuit to collect/deliver goods.

In particular, sect. 2 shows how data and task ontologies could be defined to support both design and implementation of the ubiquitous information services. Sect.3 illustrates the basic implementation steps, obtained using RoR and JQMobile, to provide the mobile users with effective ubiquitous services. The ontology based implementation on Flash Builder is for further works.

## II. TASK AND DATA ONTOLOGIES TO IMPROVE USABILITY AND DATA INTEGRATION IN UBIQUITOUS SERVICES

Typically, in the MVC framework an action consists of a set of instructions processing the data stored on relevant tables. The action outcome is given by a view where the user may activate the next actions to reach the goal of the story.

Formally, a story may be defined as a temporal ordering of actions and related views that terminates when the last action is executed successfully. In particular, a story may be described by an algebraic calculus, i.e., the Theory of Interactions and Scenes (TIS) proposed in [19] inspired by the Milner's Calculus of Communicating Systems (CCS) [20]. For example, assuming that "," is the sequence operator the most simple story may be expressed formally by a notation such as:

$$S_1 = a_1, a_2, \dots a_n \quad (1)$$

being $a_i$ a general action. Using the choice operator "+", one may express a more articulated story as follows:

$$S_2 = a_1, a_2 + a_3, a_4 \quad (2)$$

whereas the parallel operator "|" allows us to represent the parallel behavior of the actors involved in a story. For example A | B means that the actions executed by A (e.g., the mobile user) are interleaved with the ones executed by B (e.g., the server). Therefore, the contemporaneous execution of two actions a and b is represented by:

$$a, b + b, a \quad (3)$$

unless a and b are complementary actions, i.e., unless action b is due to action a or vice-versa. Indeed, in the latter case the joint execution a | b produces an unobservable action i.

A complete description of TIS is outside the scope of the paper. Here, it is enough to recall that a story may be specified by an activity frame [21] consisting of a temporal ordering of actions executed by cooperating actors and that, in our case, it results in a temporal ordering of views governing the interactions between the mobile user and the application running on the server or on user mobile. In TIS we have shown that software tools that support the story based design should be preferred since they make easier to derive system implementation from use case specifications.

For example in an RoR server, the above use story $S_2$ may be implemented by a controller $S_2$ consisting of five main actions index, $a_1$, $a_2$, $a_3$, and $a_4$, where the action index produces the first view of the story. The temporal ordering of the actions is implemented by imposing that the view index offers two actions $v_1$ or $v_3$. Once activated, view $v_1$ should produce view $v_2$ whereas $v_3$ should produce $v_4$. In both cases the story $S_2$ terminates and the user may enact another story.

However, our support for using the MVC framework is not only because it facilitates implementation but also because it facilitates data integration. In fact, in such framework data may be managed by using objects derived respectively from either tables or XML files, and data integration may be obtained by converting all the relevant datasets resident on separate computers in MySQL tables or in XML files that can be queried by a distributed platform, e.g., [22], and [23].

Let us note that downloading data derived from joining different tables are feasible using MySQL cluster or specialized XML platform at condition that the datasets on different machines are represented by the same structure of the MySQL tables or the same XML formats. This is a difficult constraint unless one uses a common vocabulary for representing data. As a consequence, we have chosen to represent the datasets by RDF triples or OWL expressions [24] following standard data ontologies consisting of controlled vocabularies containing interrelated terms.

Simple RDF based functions or SPARQL1.1 procedures [25] may be used to carry out distributed RDF queries and transactions. The RDF file resulting from the query is then converted into objects. In the next section we will show how task and data ontology may be used by discussing the implementation of a use story dealing with searching and reserving a park.

## III. UBIQUITOUS SERVICES BASED ON ONTOLOGIES: RoR IMPLEMENTATION POWERED BY JQMOBILE SCRIPTS

In general, a correct task ontology consists of the temporal ordering of the interactions carried out by an user M in cooperation with the information system U to achieve the task goal (liveness) without remaining blocked in some intermediate state (safety).

Therefore, to verify that an use story develops from an initial to a final state (possibly coinciding with the initial one) passing through a sequence of desired states is enough to prove that the desired sequence of the use story is obtained by the joint behavior of the actors M and U.

In particular, to prove that the actors M and U behave correctly during the park searching and reserving story, we have to carry out the following steps:

- first, we have to define the desired sequence of global states by an expression such as:

$$S0, V_{park\_on\_GoogbleMaps}, V_{park\_vacancies}, V_{park\_reserve}, V_{reservation\_pay}, V_{reservation\_paid}, S0 \quad (4)$$

  where we have assumed to characterize the global state of the overall system by the view V displayed to the mobile user to support the next actions of a story.
- then, we should prove that the sequence (4) may be obtained if the actors M and U behave as follows:

actor M

$M_0 = A_{park\_show\_all}!, M_{01} + ... + A_{back}!, M$

$M_{01} = A_{park\_vacancies}!, M_{011} + A_{back}!, M_0$

$M_{011} = A_{park\_reserve}!, A_{reservation\_pay}!, A_{reservation\_paid}!, M_0 + A_{back}!, M_0$

actor U

$U_0 = A_{park\_show\_all}?, V_{all\_parks\_on\_GoogbleMaps}, U_{01} + ... + A_{back}?, U$

$U_{01} = A_{park\_vacancies}?, V_{park\_vacancies}, U_{011} + A_{back}?, U_0$

$U_{011} = A_{park\_reserve}? \; V_{reservation\_pay}, A_{reservation\_paid}?, V_{reservation\_paid}, U_0 + A_{back}?, U_0$

The TIS approach recommends that only after having proved that the joint behavior M | U supports correctly all the desired use stories, the actor U is implemented by the proper RoR controllers. Fig.2 shows the JQMobile views obtained by the joint operations of actors M and U to enact the use story specified by sequence (4).

In particular, after pressing the icon show all (upper left) actor M receives from U the view of all the parks by $V_{park\_on\_GoogbleMaps}$ (upper center). The users M may obtain the vacancies of a certain park by pressing the relevant park icon. These vacancies are displayed on the mobile according to the view $V_{park\_vacancies}$ (upper right). User M may reserve the park by pressing the car icon (upper right), thus obtaining the view $V_{reservation\_pay}$ (lower left) to pay for the service. The payment is done by pressing the icon pay and filling the relevant form on the lower right.

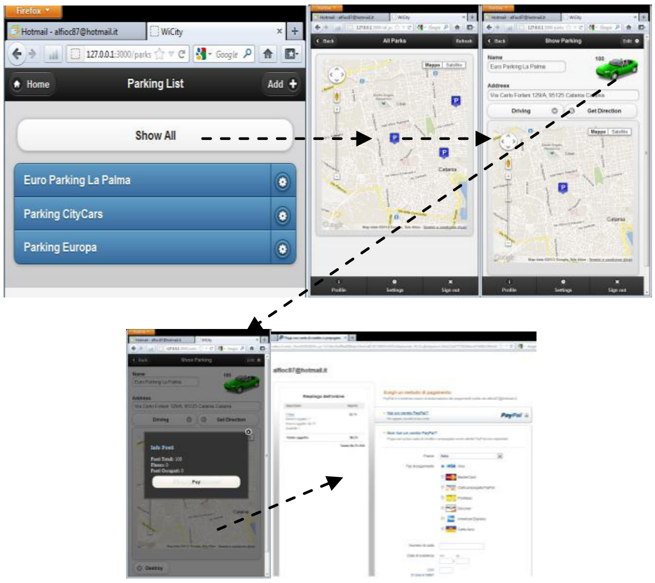

Figure 2. Sequence of views related to the park searching and reserving story

To integrate the data resident on different machines, we have to follow the procedure indicated in the previous section. Therefore, besides the vocabulary FOAF describing people [26], we have to define the other relevant terms of the distributed application vocabulary by suitable RDF expressions, e.g., parks defined as follows:

<wicity:Park rdf: about =" ">
name, address, geo:lat, geo:lng, occupancies,
capacity, cost_per_hour, cost_half_day.
</wicity:Park>

In this way, the RoR park controller may execute the function read_rdf shown in fig.3 to send a query to all the remote machines in which park information is stored according to the above RDF vocabulary.

```
def read_rdf
wicity = RDF::Vocabulary.new("...")
graph = RDF::Graph.load
("http://www.wicity.altervista.org/parks.xml")
    query = RDF::Query.new
    ({parks =>
      {RDF.type   => wicity.Park,
       wicity[:name] => :name,
       wicity.address => :address,
       wicity.lat => :lat,
       wicity.lng => :lng,
       wicity.occupancies => :occupancies,
        wicity.flow => :flow,
      ... }
    })

...
    @park = []
    i=0
    query.execute(graph).each do |solution|
    @park[i] = Pp.new
    @park[i].name = solution.name
    @park[i].address = solution.address
    @park[i].lat = solution.lat
    @park[i].lng = solution.lng
    @park[i].occupancies = solution.occupancies
    @park[i].flow = solution.flow
     ...
    i=i+1
    end
end
```

Figure 3. Function read_rdf for downloading remote RDF files and converting them into RoR objects.

After executing the query, the received RDF fields are stored in the RoR object called Pp in fig.3, assuming that the object Pp has been previously defined in the RoR *Model* section. Once the data featuring the remote parks are converted from RDF triples to RoR objects, they may be displayed to the users according the outlined task ontology.

## II. CONCLUDING REMARKS

The paper has illustrated how the use of data ontology consisting of controlled vocabularies may help data integration in ubiquitous information systems. Also, a suitable task ontology consisting of the temporal ordering of user actions supported by the system should be adopted to favor the system usability and the formal verification of correctness of the overall system (safety and liveness).

The definition of both data and task ontologies should take into account the actions usually carried out by the user to execute a task. This may be obtained by storing the use stories in a design memory shared by a community of designers so that the use stories may be progressively refined in search of the ones that best fit the user behaviors [27], [28], [29]. Software environments following the MVC paradigm, such as Ruby on Rails, should be preferred to implement the ubiquitous information system.

In future we plan to implement all the services offered by an ubiquitous information system using an ontology based approach. To this aim, we are deepening the task and data ontologies that better help usability as well as data integration, coherence and privacy without degrading the system performance. This will be carried out by taking into account the proposals available in literature, e.g., [30], [31], [32]. Ontologies will be also used to help user decisions [33] and to develop mobile recommending services on the base of the past user preferences [34].

## REFERENCES

[1] Kupper A., Location based services: fundamentals and operations, Wiley, 2005

[2] Bettini C., Maggiorini D., Riboni D., Distributed Context Monitoring for Continuous Mobile Services, MOBILE INFORMATION SYSTEMS II, IFIP, Vol.191, 123-137, 2005

[3] T.M. Nguyen, Complex Data Warehousing and Knowledge Discovery for Advanced Retrieval Development, Information Science Reference, 2010

[4] Faro, A., Giordano, D. and Spampinato, C., Integrating Location Tracking, Traffic Monitoring and Semantics in a Layered ITS Architecture. *Intelligent Transport Systems,* vol.5(3), 197-206, 2011

[5] Powers,S. Practical RDF, O'Reilly Media, 2003

[6] Elsbach K. D., Barr P.S., Hargadon A.B., Identifying Situated Cognition in Organizations, Organization Science Vol. 16(4), 422–433, 2005.

[7] Faro, A., Giordano, D. Concept Formation from Design Cases: Why Reusing Experience and Why Not. Knowledge Based Systems Journal, vol.11(7), 437-448 , Elsevier, 1998

[8] Faro, A, Giordano, D., Ontology, esthetics and creativity at the crossroads in Information System design, Knowledge Based Systems , Volume 13 (7), 515-525, Elsevier, 2000

[9] Dubberly Design Office, The Model-View-Controller Pattern in a Rails-Based Web Server, http://www.dubberly.com/wp-content/ uploads / 2011/04/DDO_Article _MVC_ Pattern . pdf, 2011

[10] Goldstein E. B., The Ecology of J. J. Gibson's Perception, Leonardo, Vol. 14(3), 191-195 , MIT Press, 1981

[11] Hartl M., Ruby on Rails 3, Addison Wesley, 2011

[12] David M., Developing Websites with jQueryMobile, Focal Press, 2011

[13] Corlan M., Flash Platform Tooling: Flash Builder, Adobe, 2009

[14] Faro, A., Giordano, D. and Spampinato, C., Adaptive background modeling integrated with luminosity sensors and occlusion processing for reliable vehicle detection. IEEE Transactions on Intelligent Transportation Systems, Vol.12(4), 1398 - 1411, 2011

[15] Faro, A., Giordano, D. and Spampinato, C.: Evaluation of the Traffic Parameters in a Metropolitan Area by Fusing Visual Perceptions and CNN Processing of Webcam Images, IEEE Transactions on Neural Networks, Vol. 19 (6), 1108-1129, 2008

[16] Crisafi A., Giordano D., Spampinato C., GRIPLAB 1.0: Grid Image Processing Laboratory for Distributed Machine Vision Applications. Proc. Int. Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, WETICE '08. 188-191, IEEE, 2008

[17] Cannavò F., Nunnari G., Giordano D., Spampinato C., 2006. Variational Method for Image Denoising by Distributed Genetic Algorithms on GRID Environmentc. Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, WETICE '06. 227-234, IEEE, 2006

[18] Leduc, G., Road Traffic Data: Collection Methods and Applications, Working Papers on Energy, Transport and Climate Change, N.1, JRC European Commission, 47967, 2008

[19] Faro, A., Giordano, D. A theory of interactions and scenes for user centered systems specification and verification , Proc. of Asia Pacific Software Engineering Conference. APSEC '97 and ICSC '97, IEEE, 1997.

[20] Milner R., A Calculus of Communicating Systems, Springer Verlag, 1982

[21] Goffman E., Berger B., Frame Analysis, Northeastern, 1986

[22] Bunch C., An Evaluation of Distributed Datastores Using the AppScale Cloud Platform, Proc. 3rd International Conference on Cloud Computing (CLOUD), IEEE, 2010

[23] Abitebout S., et al., A Framework for Distributed XML Data Management, LNCS, 3896, 1049-1058, Springer, 2006

[24] Lacy L.W.,Owl: Representing Information Using the Web Ontology Language, Trafford Publishing, 2004

[25] Quilitz B., and Leser U., Querying Distributed RDF Data Sources with SPARQL, Semantic Web: Research and Applications LNCS 5021, 526-538, 2008

[26] W3C, Fiend of a Friend project, http://www.foaf-project.org/

[27] Giordano, D. Evolution of interactive graphical representations into a design language: a distributed cognition account, International Journal of Human-Computer Studies, Vol. 57(4), 317-345, 2002

[28] Faro, A., Giordano, D. StoryNet: an Evolving Network of Cases to Learn Information Systems Design. IEE Proceedings SOFTWARE, vol.145(4), 119-127, 1998

[29] Faro, A., Giordano, D., Design memories as evolutionary systems: socio-technical architecture and genetics, IEEE Proc on Systems Man and Cybernetics, vol.5, 4334-4339, IEEE, 2003

[30] Zhai, J., Jiang, J., Yu, Y. and Li, J.: Ontology-based Integrated Information Platform for Digital City, *IEEE Proc. of eWireless Communications, Networking and Mobile Comp.,* WiCOM '08, 2008.

[31] Faro, A., Giordano D., Musarra, A.: Ontology Based Mobility Information Systems. Proc. of IEEE Systems, Men and Cybernetics SMC03, vol.3, 4288-4293, 2003.

[32] Faro A., Spampinato C., Implementing ITS 3.0 Applications by Integrating Ruby on Rails, Sesame and Protegè Technologies, Seventh International Conference on Signal Image Technology & Internet-Based Systems, Dijon, France, IEEE, 2011

[33] Sauter V.L., Decision Support Systems for business intelligence, Wiley, 2011

[34] Jannach D. et alii, Recommender Systems (an introduction), Cambridge University Press, 2010