Computer Science and Information Systems 12(1):115–133 DOI: 10.2298/CSIS140108085V

Customized QoS-based Mashups for the Web of Things: An Application of AHP

Nikos Vesyropoulos and Christos K. Georgiadis

Department of Applied Informatics, University of Macedonia
156 Egnatia Street, 54006, Thessaloniki, Greece
{nvesyrop, geor}uom.edu.gr

Abstract. During the past few years the notion of the Web of Things (WoT) as a collection of innovative ideas and technologies, is emerging as a compelling way for businesses to provide value-added services to fulfill their clients' needs. By creating virtual counterparts of a number of real life objects, whose functionality can be accessed through simple RESTful service operations, an abundance of physical services can complement the existing services provided by an enterprise. This however can complicate things in matter of selection and composition of services for end-users. As a result, a need for new filtering techniques arises that can identify both physical and virtual services and provide value added mashups, based on personalized and QoS-based criteria. In this work, we propose a framework in order to address this issue, integrating a Multicriteria Decision Analysis method, in order to create customized QoS-based service mashups.

Keywords: Web of Things, QoS, RESTful Services, AHP

1. Introduction

The WoT, as an extension to the Internet of Things (IoT), has recently gained wide attention, as it promises a plethora of benefits introduced by the IoT, along with the easiness of use of well-known Web standards. These standards are used in order to enable the communication and interoperation between Web Services (WS) and real-world objects, which can provide access to their operations through simple RESTful calls. In order to accomplish a connection to such an object, or thing, all that is needed is a simple URL that will give access to the virtual representation of the object. This visual representation can be in the form of a Web page that can occasionally provide access to services corresponding to the item in regard. Taking advantage of the well-established Internet's architecture and protocols, it is possible to facilitate a network of virtual representations of objects, which can be implemented with the integration of sensors and small sized servers (smart gateways) [8]. Instead of relying on SOAP-based WS, the notion of the WoT proposes the usage of the well-known HTTP protocol and standards that emerge from Web 2.0 [25]. In WoT, HTTP is used as an application protocol [18], thus making it ideal for adopting and applying RESTful services. These services can provide access to the functionality of the virtual representation of a "smart" physical object (one equipped with sensors and can access the net) by using URIs and a set of HTTP methods (GET, POST, PUT, DELETE). Thus data can be exchanged between "smart things", typically in JSON, Atom or XML format. This enables the handling of these virtual representations by service clients that are compatible with the HTTP protocol, such as Web browsers.

Apart from their operational characteristics, WS have a number of QoS attributes that are not directly connected to the operation of the service, nevertheless play a pivotal role during the WS selection process as they refer to characteristics (security, effectiveness, etc.), that are of fundamental importance especially in business-centric compositions. The above apply for both WS-* and for RESTful compositions. Especially in the case of REST and consequently in WoT environments, these compositions result in value added services in the form of Web2.0 mashups. In this work, we focus on QoS-based and personalized Web Service compositions in WoT scenarios. There is an abundance of Web Services available, which makes it difficult for end-users and enterprises to find the services that better suit their specific needs. In addition, when considering the plethora of services provided by the representation of physical objects in WoT scenarios, it is easily understandable that this issue escalates even more. While the operational logic of a service is the primary concern for users and enterprises, since there is plethora of available options that could eventually comply, the composition of WS into more value added services can be affected by QoS criteria. When two or more services have the same properties regarding the operations they perform, the end-user or the composition engine makes the selection based on QoS criteria. Using a technique from the field of operational research, and in particular Analytic Hierarchy Process (AHP), which is an acknowledged multi criteria decision analysis approach, the aforementioned values and weights can produce vectors, thus enabling an engine capable to compare such vectors (using AHP), to produce a personalized QoS composition, that will differ for each end user. In the proposed framework, users can receive personalized compositions of services in real time by providing their needs in the form of a list of weights of QoS criteria.

2. Service Architecture for the Web of Things

During the past decade, many enterprises have shifted their focus towards the use of WS, and in more particular WS-* services, in order to take advantage of the growing opportunities provided by the constant evolution and adoption of the Internet [28][45]. Nowadays though, with the shift to the Web 2.0, there is a trend towards the adaptation and usage of the more flexible and scalable REST services. With the advent of the IoT and the WoT paradigms, the need to shift to RESTful WS is even greater as they are an integral part of the aforementioned paradigms [43]. Nevertheless, WS-* services are still preferred for dynamic compositions in e-commerce transactions that require business level decisions (based on the orchestration and choreography of services) [31].

While it is of great importance to create virtual counterparts for smart devices, and “things” which exist in the physical world, the true innovation in the WoT paradigm is in moving a step forwards: from just having a simple informative Web page for each individual thing or device, to the abstraction of WS provided from those devices discoverable and reachable through the HTTP protocol. In this way, “things” can be considered as resources exploitable by these WS. But the question that rises is which of the aforementioned WS architecture is preferable and better suited in fulfilling the needs in communication and interoperation between smart devices, and composition of value added services [14]. In order to facilitate the resolution of this concern we further analyze the two dominant paradigms.

2.1. Soap Based Web Services (WS-*)

WS-* services are based on the exchange of messages through Simple Object Access Protocol (SOAP) envelops which, as a transmission medium, enables the interoperation of servers and clients based on remote procedure call (RPC) methods.. In order to reveal the operations provided by a WS and to describe its interface, Web Service Description Language (WSDL) files are being used. WSDL files describe the type of inputs allowed and type of outputs expected, using parameters such as time parameters (current time and date, time that a service stays accessible), the general set of operations and effects that the invocation produces. All the aforementioned details are stored in a general registry where clients and services can discover them. While WS-* services are considered very reliable, currently the trend is towards the stateless RESTful WS, partly due to the complexity of developing and monitoring SOAP-based Web Services

2.2. Representational State Transfer (REST)

Representational State Transfer (REST) is a service-oriented architecture for distributed systems [12]. REST defines specific architectural principles on designing WS, based on resources and their representations, allowing loosely coupled systems to interoperate. While resources can be every possible information, or concept, representations are the document or format that presents the resource or its current state. A number of representations can be provided for a single resource. Most common representations are simple HTML pages, XML formatted pages and JSON files.

The main idea behind REST is that services can be easily accessed through a Universal Resource Identifier (URI), and that the different states of the resources are exclusively addressed and transferred over HTTP, using specific HTTP verbs. Every RESTful WS, provides access to resources and their representations, through URIs. Unlike the UDDI repository provided in WS-*, clients can only identify and interact with services knowing their URIs.

3. Composing Web Services

3.1. Service Composition

Soap-based and RESTful architectures differ significantly on how they manage the composition of services into value-added services for business processes [36]. WS-* service compositions are based on the Business Process Execution Language (BPEL). It is an XML based language that enables the description of business processes. BPEL defines the interactions between services and the service composition engines. BPEL presents the invocation order and handles information regarding the state of a service.

On the other hand, RESTful service compositions are usually handled as Web 2.0 mashups. A Web mashup, is a Web application or a Web page which usually uses application programming interfaces (APIs) in order to blend information from multiple sources to create compelling services.

While Web 2.0 mashups are considered popular method for the composition of RESTful WS [13], other alternatives exist. It is possible to create value-added services using an extension to BPEL (proposed in [32]) or through the use of JOpera tool [33], which is an eclipse plugin. JOpera is indeed a promising alternative, although it is in fact a remote procedure call-oriented in nature tool.

As more and more embedded devices (like smartphones and sensor equipped appliances) will be apply to provide their functions as services online, and an abundance of real objects will essentially become a part of ambient spaces (AS) (interoperating and communicating over TCP/IP networks), the need to create value-added services by composing numerous embedded-device enabled services, including if possible traditional WS, is growing exponentially.

3.2. Mashups

3.2.1 Physical-Virtual Mashups

These mashups regard the composition of services provided not only from traditional Web-based services (virtual services) but from services provided by embedded devices and physical objects in the real world. Physical services enable enterprises, end users or other smart devices to interact with the embedded devices by sending HTTP requests [26]. Compelling services can be provided by the composition of physical-virtual services in mashups where for example information regarding energy consumption in various appliances in an enterprise can be presented on an interior map service.

3.2.2 Physical-Physical Mashups

Such mashups are created by composing functions provided as services solely by embedded devices with data provided by real world objects. Sensors can communicate through mini-servers, named Smart Gateways [40], without the need for programmers to develop deep knowledge on their architecture, as all functions can be accessed through direct HTTP calls. Applications of such mashups include the cooperation between sensors in the production line of enterprises, where for example the increase of temperature in a specific manufacturing machine can result in a message exchange between the machine and the air ventilation and air condition management systems, and to an executive employee for monitoring purposes.

3.2.3 Business Intelligence Mashups

Business Intelligence Mashups regard to web application that effectively integrate a number of their own business-specific local applications and information with resources provided by external WS, in value-added services [20]. In order to achieve Business

Intelligence Mashups, occasionally Business logic semantics should be involved into the composition, resulting in the need of BPEL-based orchestration and WS-* protocols. It is crucial to consider a dynamic adaptation to arising user needs and requirements in real-time. In more detail, available WS should be added, altered or removed dynamically in order to respond to the rapid business-environment changes and QoS requirements.

4. The Analytical Hierarchy Process

The Analytical Hierarchy Process (AHP) is a multi-criteria decision analysis methodology proposed by Saaty [35], in 1980. It is an approach aiming at organizing data for proper decision making, which can seamlessly process both qualitative and quantitative data [21]. AHP has been used in an abundance of applications related with decision-making, in various fields such as process planning, optimal selection between alternatives, resource allocations and resolution of conflicts [30].

The afore-mentioned methodology consists of four steps [6]: i) Designing the hierarchy of the components of the problem, as seen in Fig.1 ii) Setting the priority of the criteria, by inputting pairwise comparisons, while using an ordinal scale from 1 to 9 iii) Ranking all the options according to the aforementioned pairwise comparisons iv) Providing an overall relative score for each alternative. As a result, AHP receives as inputs a number of different measures, weights and criteria and allows the computation of a single value, representing a unique and overall score.

In this work, we will use AHP in order to evaluate the optimal service mashup, based on criteria and their appropriate weights, given by the end user, regarding QoS characteristics. In addition user evaluations will also be taken into consideration, thus combining two selection methodologies that are rarely used in conjunction in modern literature: selection based on external evaluation and selection based on user's criteria.

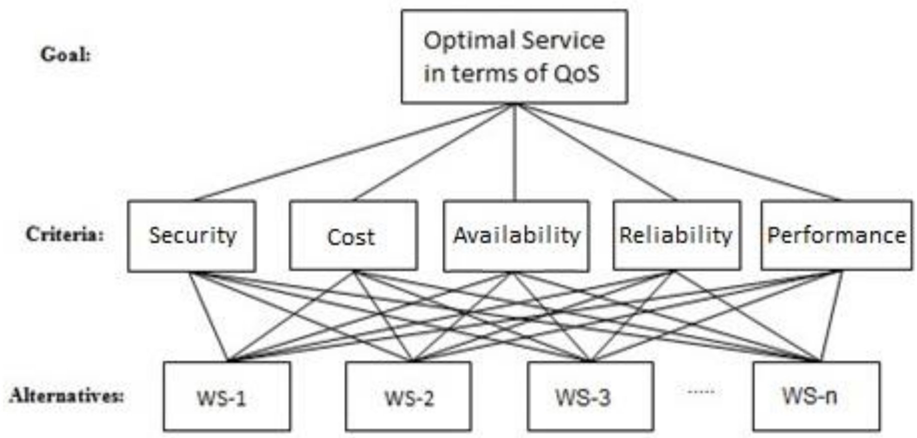

Fig. 1. The hierarchy in AHP, in a QoS-based composition example

In order to ensure that selections satisfy the business logic requested by the end-users, meaning both the operational/functional features of the WS and the QoS-based features, we will provide a framework in section 6. In this framework describe the selection process of both the RESTful and SOAP-based services, the integration of user weights and external evaluation and the application of the AHP steps mentioned above in this context, using values that occur after the aforementioned integration.

5. Related Work

The AHP method has been successfully applied in many fields, in order to address issues regarding optimization based on criteria. In [7], the methodology is used in order to identify the ranking of software versions, based on specific metrics. In addition a comparison of AHP and another method named Data Envelopment Analysis (DEA) is performed, and both wield similar results, thus confirming the methods validity. In [1], researchers applied the AHP method, as a multi-criteria evaluation approach employed in a scenario where multiple alternatives where compared regarding environmental loading, and calculate the weighting factors of impact categories in life cycle impact assessment. In [11] researchers apply AHP in order to compare different database designs and select the optimal one. Closer to our case study are the results of [2], where AHP has been applied in order to identify key locations between alternatives, with the highest potential for successfully selling clothes in a retail store. The importance of weighted criteria in user-based selections is demonstrated. Nevertheless we believe that in Web 2.0 environments, the aforementioned study can be enhanced, using evaluations from other users, something we integrated in our approach.

In modern literature, a number of researchers have applied Multicriteria Decision Analysis algorithms in order to rank services based on their QoS characteristics. In [39] researchers have created a QoS-based ontology for SOAP-based Web Services, adopting AHP as a general mechanism in order to evaluate services. While it offers promising results, in our approach we try to implement the AHP methodology in a composition of both WS-* services and RESTful services, under the context of the WoT. In [22] they apply another MCDA methodology in order to enable a composition engine to automatically select the optimal WS, based on QOS criteria. In conjunction they also apply the Analytical Network Process (ANP) in order to find intercorreletions between QoS characteristics. This is an interesting approach but we consider it outside the scope of our research. In addition as PROMETHEE does not apply pair wise comparisons, it is not as suitable as AHP for highly personalized service selections. In [5], the authors apply an alternative approach by examining QoS characteristics in hospital web sites, by taking advantage of differentiation of AHP, namely the Fuzzy AHP.

Regarding the field of the WoT notions, modern literature sheds light to the importance of integrating WS, as they provide the means to enforce scalability and interoperability in a plethora of applicable areas, while in addition they are a well-accepted standard in WoT scenarios. So, in our opinion they outperform other related similar approaches. Nevertheless, while the WoT promises state of the art, and pioneering solutions, many challenges rise, which are tackled in recent research topics, regarding the discovery of physical and virtual WS, the effective and efficient interoperation of smart devices and real world objects and the composition of Web mashups to provide enhanced Web applications. Stirbu et al. in [37], propose an approach for integrating devices to the Web, through REST principles while focusing on the discovery of services created by embedded devices, while in [15], a set of requirements are presented for the efficient querying and discovering of services that can be applied to virtual services as well as services from the function of physical objects. Those include the Minimal Service Overhead, Minimal Registration Effort, Support for Dynamic and Contextual Search and Support for On-Demand Provisioning.

Guinard et al. in [16] describe an interesting application of the WoT notions, as they present a schema of home appliances creating a mashup, presenting their energy consumption, giving users the ability to monitor and lower their energy demands, based on Smart Gateways. In [24] a smart gateway approach is presented in order to handle the requirements of Cyber-physical systems, by exploiting Restful principles. On the other hand, Akribopoulos et al. in [3] present an architecture of smart objects that expose their functions as WS, without the need of Smart Gateways. This is a compelling approach, since the sensor-equipped devices can instantly issue HTTP-based requests, though the exchange of messages is more complicated. Duquennoy et al. in [9], [10] present Smews, an embedded Web server to enable the transformation of physical objects to service providers, providing support of multiple in-flight packets while handling several simultaneous TCP connections. In a nutshell, they describe efficient tiny embedded Web servers that request minimal volatile memory.

Guinard et al. in [17] attempt to formalize design parameters, and illustrate how applications can be built on top of the RESTful oriented architecture. Ostermaier et al. in [29] demonstrate the appliance of well-known Web infrastructures in publishing services and data by sensors and objects. They present Dyser, a real-time search engine for the devices, capable to find real-world entities, based on the state they exhibit during a query. Mayer et al. in [27] illustrate the use of a semantic discovery service for Web-enabled smart things, named DiscoWoT, which allows users, to dynamically alter the method of finding representations of resources at run-time. On a different perspective, Zhong et al. in [46], propose the notion of the Wisdom Web of Things based on the data cycle, namely “from things to data, information, knowledge, wisdom, services, humans, and then back to things”, to allow harmonious interaction between human, societies, and smart things.

While service mashups is a relatively new approach in creating value-added services, it has attracted attention both from researchers and from entrerprises. In [44] a framework for QoS-based mashups is presented, adopting skyline approaches and adaptive Particle Swarm Optimization. Nevertheless, in contrast to our approach, the weights inserted in QoS characteristics by the end-user are limited to a sum of values, where in our approach, the application of AHP allows users to insert pairwise comparisons for each criterion regarding each alternative. This results in a higher level of personalization for the end-user. In [19] the WTE+ framework for automated service mashups, based on QoS characteristics is described. It is a business-centric framework that enables enterprises to automatically create and monitor mashups, without the need of human intervention, based on planning algorithms and exhibits promising results. The main differentiation compared to our approach is that the focal point of our methodology is the end-user, to whom we provide the means for a custom selection and composition. A more user-oriented approach is the work presented in [4], as well as the metamodel for composite end-user based Web mashups, presented in [34]. Early work findings in [42] also highlight the ever-growing need for QoS-oriented frameworks for the development of user-centric mashups.

6. Customized QoS-based selection and composition

While RESTful services are nowadays widely spread and a majority of enterprises provide such services, they lack the complexity and business rule handling that can be found in WS-* services. As a result in our composition framework we decided not to omit SOAP based services. Nevertheless, when taking into account the abundance of physical services that can be provided in WoT related scenarios, which can solely be accessed through RESTful operation calls, and, in addition, the emerging Web2.0 technologies that are associated with the WoT, it is easily understandable that our primal focus in composition techniques constitute of Web2.0 mashups. As it is possible to integrate SOAP based services, or simple calls to invoke such services into mashups (e.g., see http://www.ibm.com/developerworks/xml/library/x-mashups.html) [38] and since mashups are commonly constructed on the client's side, usually through a browser, they provide a compelling and highly personalized value added service.

Based on the principles of the social web, where evaluations, comments and feedback on products and services are more than ever reachable, more and more WS users rely on evaluations provided by other users that have previously used a service. The benefits of this approach coming from the Web2.0 are unparalleled [23], and thus cannot be omitted in WoT scenarios. In order to comply with the above, it is necessary for the composition engine to additionally provide a series of evaluations by users or enterprises that have already used a specific service. These are distinct evaluations of each criterion. Provided this evaluation is not in the form of a single value, but is analyzed in sub categories (according to the predefined QoS criteria categories), we propose the inclusion of criteria specific evaluations by users in the widely adopted service discovery model. These evaluations can be retrieved by the composition engine and matched with entries in the WS repositories.

6.1. Framework Description

We propose a framework consisting of:

1) UsQoS – A list of initial QoS criteria requested by the user.
2) ProvQoS – A list of values for certain QoS criteria (provided-oriented criteria), such as privacy awareness, offered by the service providers. These are fixed values that represent a kind of self-evaluation for their services.
3) EvQoS – A list of values for certain QoS criteria (user-oriented criteria), not necessarily identical with providers-oriented criteria (e.g. response time) offered by other users (e.g. users' evaluation / fame).
4) UserWeights – A list of values-weights that the end user gives, that correspond to his needs in terms of QoS criteria (UsQoS, ProvQoS or EvQoS).

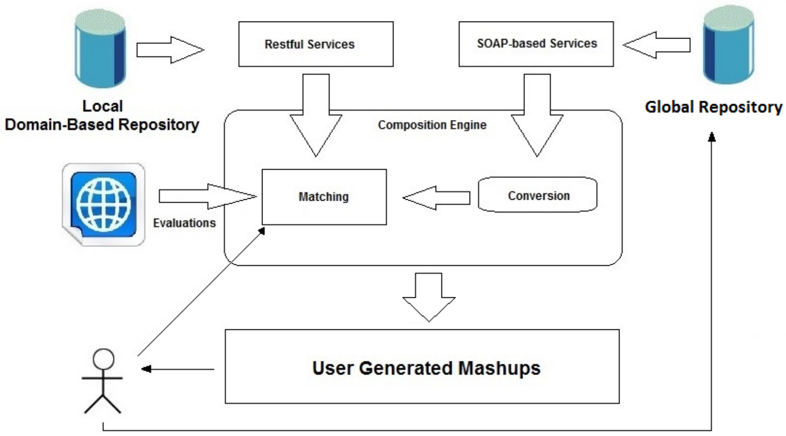

Fig. 2. The customized service composition framework

As will be analyzed below, in our approach the selection of the criteria weights is a two-step process. Initially, a selection of suitable SOAP-based services is taken place in Global Repository, based on their functional features and according to pure operational requirements. Furthermore, the user selects a number of QoS-based criteria that she requests from a service provider. In addition, as we will discuss below, the user provides her weights. As a result the composition engine returns a number of services that comply with the required business logic (expressed by both functional and QoS-based criteria). The returned services might comply with all or with a number of the QoS-based criteria, but in addition could also provide several other criteria, apart from the ones entered before, that the end-user should also consider.

6.2. Implementation

Our methodology consists of a number of phases. The first phase regards the input of the functional requirements. The next two phases regard the selection of the criteria weights, while the final phases include the implementation of the AHP methodology. It should be noted that while AHP only accepts numerical values for the criteria evaluations, linguistic selections can also be provided through normalization. This results in a more user-friendly approach, as the choices can be part of a Likert scale, which especially in user-oriented criteria can be presented as linguistic expressions.

Phase 1:

The end-user provides her functional requirements, regarding the operations he requests in the final mashup. The composition engine returns a number of WS that correspond to the user's needs.

Phase 2:

The user inputs a number of QoS criteria (UsQoS), from a predefined list of QoS_i criteria that are registered in the repository, where i=1, 2…n. The user also inputs pairwise comparison weights (UserWeights) for her choices. For example for $UsQoS_{i,}$ where i = 3 the user may provide the following preferences:

$\{(UsQoS_1, UsQoS_2, PwC_{12}), (UsQoS_2, UsQoS_3, PwC_{23}), (UsQoS_1, UsQoS_3, PwC_{13})\}$

Based on the weights, and following the principles of the AHP methodology an initial Eigenvector is being calculated, which clearly indicates which criteria are the most important for the user. In addition the user selects how strict the composition engine should be regarding services that only have a number of the QoS criteria requested. The returned response, based on the level of strictness, can be in the form:

$ProvQoS_{WS1} = \{(QoS_1, value_1), (QoS_{3,} value_3), (QoS_4, value_4)\}$
…..
$ProvQoS_{WSm} = \{(QoS_1, value_1), (QoS_4, value_4), (QoS_6, value_6)\}$

and is a list of m services that have values assigned by providers (ProvQoS) for some or all of the requested QoS criteria. Typically, we choose to exclude services that lack the criteria with the highest eigenvalues.

Phase 3:

The returned list of services can be altered as follows: some criteria may exist in services returned, that were not taken into account by the user in the previous phase. In this phase the user may enhance her previous selection, with some of these additional QoS criteria.

It is of importance to stress that both the ProvQoS and UsQoS are subsets of QoS, while they represent subjunctive criteria from different perspectives, thus their intersection equals null.

$ProvQoS \subseteq QoS$
$UsQoS \subseteq QoS$
$UsQoS \cap ProvQoS = \emptyset$

Phase 4:

In any scenario involving Web 2.0 technologies it is of profound importance to include users' evaluations. As the inclusion of both custom user weights and external evaluations, is rarely documented in modern research, we believe that it constitutes a compelling feature in our proposal. Previous users of the services involved, can leave evaluations regarding their level of satisfaction. These evaluations are subjunctive and thus differ from thee ProvQoS and UsQoS values. For example the perceived level of security is a EvQoS characteristic, that differs from the ProvQoS characteristic of security. In this phase a number of EvQoS are added.

Phase 5:

In this phase AHP is applied. The steps of the methodology are as follows

Step 1

As a result of adding ProvQoS and EvQoS criteria the two dimensional vector needs to be completed with additional pairwise comparisons in order for the AHP methodology to be initiated. In an example involving 3 UsQoS, 2 ProvQoS and 2

EvQoS criteria, the two dimensional matrix of pairwise comparisons (PwC) is depicted as follows:

Table 1. Pairwise comparisons matrix

$$\begin{bmatrix} \mathbf{1} & \boldsymbol{PwC_{12}} & \boldsymbol{PwC_{13}} & \boldsymbol{PwC_{14}} & \boldsymbol{PwC_{15}} & \boldsymbol{PwC_{16}} & \boldsymbol{PwC_{17}} \\ \boldsymbol{PwC_{21}} & \mathbf{1} & \boldsymbol{PwC_{23}} & \boldsymbol{PwC_{24}} & \boldsymbol{PwC_{25}} & \boldsymbol{PwC_{26}} & \boldsymbol{PwC_{27}} \\ \boldsymbol{PwC_{31}} & \boldsymbol{PwC_{32}} & \mathbf{1} & \boldsymbol{PwC_{34}} & \boldsymbol{PwC_{35}} & \boldsymbol{PwC_{36}} & \boldsymbol{PwC_{37}} \\ \boldsymbol{PwC_{41}} & \boldsymbol{PwC_{42}} & \boldsymbol{PwC_{43}} & \mathbf{1} & \boldsymbol{PwC_{45}} & \boldsymbol{PwC_{46}} & \boldsymbol{PwC_{47}} \\ \boldsymbol{PwC_{51}} & \boldsymbol{PwC_{52}} & \boldsymbol{PwC_{53}} & \boldsymbol{PwC_{54}} & \mathbf{1} & \boldsymbol{PwC_{56}} & \boldsymbol{PwC_{57}} \\ \boldsymbol{PwC_{61}} & \boldsymbol{PwC_{62}} & \boldsymbol{PwC_{63}} & \boldsymbol{PwC_{64}} & \boldsymbol{PwC_{65}} & \mathbf{1} & \boldsymbol{PwC_{67}} \\ \boldsymbol{PwC_{71}} & \boldsymbol{PwC_{72}} & \boldsymbol{PwC_{73}} & \boldsymbol{PwC_{74}} & \boldsymbol{PwC_{75}} & \boldsymbol{PwC_{76}} & \mathbf{1} \end{bmatrix}$$

Step 2

In the next step, the eigenvector is being calculated in order to acquire the ranking of priorities from the two-dimensional matrix. After turning fractions into decimals, according to AHP method we raise the pairwise comparison matrix to the power of two. X_{ij} symbolizes the resulting matrix:

Table 2. The outcome of raising the PwC matrix to the power of two

$$\begin{bmatrix} \boldsymbol{X_{11}} & \boldsymbol{X_{12}} & \boldsymbol{X_{13}} & \boldsymbol{X_{14}} & \boldsymbol{X_{15}} & \boldsymbol{X_{16}} & \boldsymbol{X_{17}} \\ \boldsymbol{X_{21}} & \boldsymbol{X_{22}} & \boldsymbol{X_{23}} & \boldsymbol{X_{24}} & \boldsymbol{X_{25}} & \boldsymbol{X_{26}} & \boldsymbol{X_{27}} \\ \boldsymbol{X_{31}} & \boldsymbol{X_{32}} & \boldsymbol{X_{33}} & \boldsymbol{X_{34}} & \boldsymbol{X_{35}} & \boldsymbol{X_{36}} & \boldsymbol{X_{37}} \\ \boldsymbol{X_{41}} & \boldsymbol{X_{42}} & \boldsymbol{X_{43}} & \boldsymbol{X_{44}} & \boldsymbol{X_{45}} & \boldsymbol{X_{46}} & \boldsymbol{X_{47}} \\ \boldsymbol{X_{51}} & \boldsymbol{X_{52}} & \boldsymbol{X_{53}} & \boldsymbol{X_{54}} & \boldsymbol{X_{55}} & \boldsymbol{X_{56}} & \boldsymbol{X_{57}} \\ \boldsymbol{X_{61}} & \boldsymbol{X_{62}} & \boldsymbol{X_{63}} & \boldsymbol{X_{64}} & \boldsymbol{X_{65}} & \boldsymbol{X_{66}} & \boldsymbol{X_{67}} \\ \boldsymbol{X_{71}} & \boldsymbol{X_{72}} & \boldsymbol{X_{73}} & \boldsymbol{X_{74}} & \boldsymbol{X_{75}} & \boldsymbol{X_{76}} & \boldsymbol{X_{77}} \end{bmatrix}$$

Next we compute the sum of rows.

$$\text{Sum}_i = \sum_{j=1}^{7} \boldsymbol{X_{ij}} \qquad (1)$$

for i=1,2,…n, where n in this scenario is 7 as the number of criteria.

The eigenvector is then calculated based on (2):

$$\text{EV}_i = \frac{\boldsymbol{SUM_i}}{\sum_1^7 \boldsymbol{SUM_i}} \qquad (2)$$

The process should be repeated by raising the new two-dimensional matrix (X_{ij}) to the power of two and calculating the new eigenvector, until no significant differences are found between the eigenvectors.

Step 3

In the third step a matrix of pairwise comparisons is created for each criteria, each containing comparisons of alternative services in in regard to the corresponding criterion. Let the returned services be four, which means that for seven criteria, the result will be 7 square 4x4 matrixes in the form:

Table 3. Indicative pairwise comparison matrix for one criterion

$$\begin{array}{c} \\ \boldsymbol{WS_1} \\ \boldsymbol{WS_2} \\ \boldsymbol{WS_3} \\ \boldsymbol{WS_4} \end{array} \begin{array}{c} \begin{array}{cccc} \boldsymbol{WS_1} & \boldsymbol{WS_2} & \boldsymbol{WS_3} & \boldsymbol{WS_4} \end{array} \\ \begin{bmatrix} \boldsymbol{C_{11}} & \boldsymbol{C_{12}} & \boldsymbol{C_{13}} & \boldsymbol{C_{14}} \\ \boldsymbol{C_{21}} & \boldsymbol{C_{22}} & \boldsymbol{C_{23}} & \boldsymbol{C_{24}} \\ \boldsymbol{C_{31}} & \boldsymbol{C_{32}} & \boldsymbol{C_{33}} & \boldsymbol{C_{34}} \\ \boldsymbol{C_{41}} & \boldsymbol{C_{42}} & \boldsymbol{C_{43}} & \boldsymbol{C_{44}} \end{bmatrix} \end{array}$$

Where $C_{11} = C_{22} = C_{33} = C_{44} = 1$. The step is finalized by computing the eigenvector (using relation (1)) for each matrix ($AltEV_k$ for k=1,2,3,4) and thus determine the relative ranking for the alternative services, according to each criterion.

Step 4

Figure 3 shows the result of placing the values of each eigenvector on the AHP hierarchy tree.

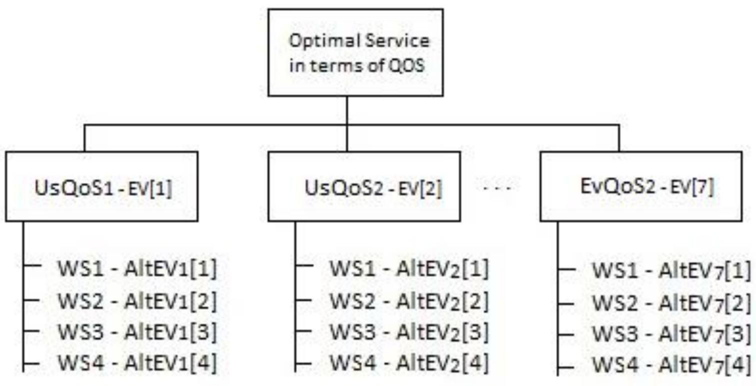

Fig. 3. Applying the Eigenvalues on the hierarchy tree

The final step in the procedure is to calculate the matrix with the alternatives' ranking AR[i], using (3)

$$\text{AR[i]} = \sum_{j=1}^{7} \boldsymbol{AltEv_j[1]} * \boldsymbol{EV[j]} \qquad (3)$$

The WS with the higher ranking in the AR[i] matrix is the optimal service in terms of QoS.

6.3. Case study

The integration of WoT notions is of high interest in scenarios involving e-commerce transactions, through mobile devices, along with the physical presence of the user in a transactional environment. In [41], we have described a plethora of benefits that could be reaped from applying the notions of the WoT in such a context. By the integration of the AHP methodology users can accomplish customized service compositions, as AHP provides the means to apply personalized user weights. In this section we offer a demonstration of the proposed framework, in a scenario involving both RESTful (software-based and physical-based) services and WS-* services. In our example we consider a mall that provides assistance, references and recommendations along with a number of exclusive services, to visitors that use its own application in their smart devices. Visitors receive both physical and software based services. Physical services can be services provided as virtual representations of the functionality of physical objects, for example two similar fridge units in a large retail store providing information regarding the arrival of new products in discount. The software-based RESTful services can be a services provided by the local retail store and is domain specific. For example it can be a service regarding the positioning of certain products in certain areas of the interior map of the store. Finally the WS-* services can be external services from another publisher, that need to be modified in order to be implemented as service calls in the user-generated mashup. Following is an example of an application of our framework

Phase 1:
In the first phase we consider that a user inputs a list of functional requirements, based on her needs, on recommendations received and on joint advertising between stores, and receives a list of five WS_i services complying with her needs.

Phase 2:
The user is prompted to select the QoS criteria that she is interested in, along with the pairwise comparison weights for her choices, in the form of {($UsQoS_i$, $UsQoS_j$,PwC_{ij})…}

In this scenario the user chooses security ($UsQoS_1$), availability ($UsQoS_2$), reliability ($UsQoS_3$) as follows:

{(Security, Availability - 6), (Security, Reliability - 8), (Reliability, Availability – 2)}

A list of services that have values assigned by the provider is being produced
$ProvQoS_{WS1}$ = {(Security, 8), (Availability, 6), (Reliability,7), (Capacity,6)}
$ProvQoS_{WS2}$ = {(Security, 7), (Reliability, 6), (Performance,7), (Robustness,4)}
$ProvQoS_{WS3}$ = {(Security, 7), (Availability, 7)}
$ProvQoS_{WS4}$ = {(Security, 9), (Availability, 8), (Reliability,7)}
$ProvQoS_{WS5}$ = {(Availability,8), (Performance, 8)}

WS_5 will be omitted as it lacks values for the most important criteria.

Phases 3, 4:
To reduce complexity we opt not to enhance the previous selection with additional QoS criteria and proceed to the integration of user evaluations. In this case study the

end-users takes into account the weights for user-perceived Security (EvSecurity) and user-perceived Reliability (EvReliability).

Phase 5:

Step 1: After adding the ProvQoS and EvQoS criteria, the two-dimensional pairwise comparison vector must be completed by the end-user. In this scenario the matrix is as follows:

Table 4. Pairwise comparison matrix completed by the user

	$UsQoS_1$	$UsQoS_2$	$UsQoS_3$	$PrvQoS_1$	$PrvQoS_2$	$EvQoS_1$	$EvQoS_2$
$UsQoS_1$	1	6	8	7	9	3	8
$UsQoS_2$	1/6	1	1/2	3	7	1/2	7
$UsQoS_3$	1/8	2	1	5	7	2	5
$PrvQoS_1$	1/7	1/3	1/5	1	3	1/2	2
$PrvQoS_2$	1/9	1/7	1/7	1/3	1	1/2	1/2
$EvQoS_1$	1/3	2	1/2	2	2	1	3
$EvQoS_2$	1/8	1/7	1/5	1/2	2	1/3	1

Step2: By converting fractions to decimals and raising to the power of two we formulate the next table

Table 5. The matrix raised to the power of two

7.0006	38.7631	24.7861	85.0006	159	35.6672	125.5
2.644	7.001	5.5839	16.5007	43.0003	9.8339	28.8336
3.3674	12.1318	7.0003	25.7088	58.125	13.042	44.5
1.1164	3.6387	2.7886	7.0007	16.0199	4.1622	11.477
0.5407	2.4208	1.6627	3.8377	7.0007	2.0240	5.7704
1.9457	8.3817	5.953	17.0006	36.5006	7.0004	30.1672
0.7036	2.5551	2.0240	4.6373	9.6921	2.7633	7.0005

Using (1) and (2) the Eigenvector is being calculated, where:
Ev = [0.4962, 0.1183, 0.1709, 0.0482, 0.0243, 0.1115, 0.0306**]**

Step 3: In this step the user is prompted to enter values for seven matrixes regarding comparisons of alternative services in in regard to the each of the corresponding seven criterion

Table 6. An indicative criterion matrixes

UsQoS$_1$

WS_1	1	3	6	8
WS_2	1/3	1	3	6
WS_3	1/6	1/3	1	4
WS_4	1/8	1/6	1/4	1

UsQoS2

WS_1	1	4	3	5
WS_2	1/4	1	3	7
WS_3	1/3	1/3	1	4
WS_4	1/5	1/7	1/4	1

Following the procedure demonstrated above we calculate the seven eigenvectors

Step 4: Based on the information collected in the previous steps the final ranking table can be calculated based on (**3**):
AR = [0.564, 0.263, 0.125, 0.048]

The final ranking is presented in Figure 4, along with the overall inconsistency which is within the optimal range. As it can be seen WS_1 is the optimal service based on QoS criteria and the custom values and weights.

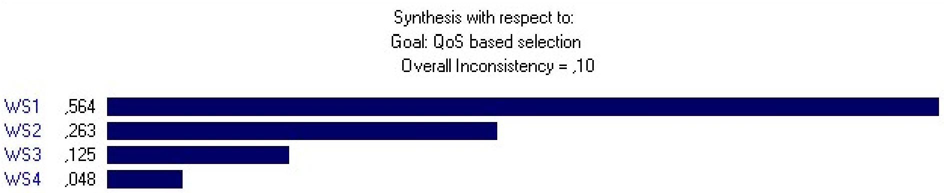

Fig. 4. Service ranking calculated by the Expert Choice Tool

7. Conclusion

The WoT provides an abundance of opportunities for enterprises and end-users. Nevertheless the ever-growing number of services available is exponentially increased as a result of the virtual representations of “smart objects”, that provide functions as services. As a result the selection of stand-alone services, or services to be included in value added compositions, is considered now, more than ever, a challenging task. In this paper we provided a framework for the creation of customized service mashups, by integrating the AHP method. As analyzed above, we chose to implement our framework using an MCDA method, as we intended to provide a methodology for developing more personalized service mashups, based on QoS characteristics. By using AHP, we enable end-users to rate and apply weights to QoS characteristics, based on their preferences. While similar approaches place their focus on the automation and the overall performance of the composition algorithm, they usually neglect the need for personalization in such compositions. In addition, by integrating user evaluations the novelty of our methodology is highlighted, which is the combination of selection methodologies that are rarely used in conjunction in modern literature. Namely these are the selection based on external evaluations and on user’s criteria. In addition, we chose AHP as the preferred MCDA method, as it is considered one of the most reliable MCDA methods, while the generated hierarchy model enables an easier monitoring of the overall decision problem. We strongly believe that apart from a stand-alone methodology, our approach can also be combined with performance-oriented techniques in order to enhance them. We are currently working on combining our approach with the notion of skyline service selection, which is proved to be an efficient methodology for performance based compositions. Thus we aim to add the element of customization without compromising the overall performance.

References

1. Agarski, B., Milankovic, D., Ilic, M., Budak, I., Vukelic, D., Hodolic, J.: Application of analytic hierarchy process for weighting of impact categories in life cycle impact assessment. In Proceedings of the 7th International Quality Conference. (2013)
2. Akalin, M., Turhan, G., Sahin, A.: The application of AHP approach for evaluating location selection elements for retail store: a case of clothing store. International Journal of Research in Business and Social Science, Vol.2, No.4, 1-20.(2013)
3. Akribopoulos, O., Chatzigiannakis, I., Koninis, C., Theodoridis, E.: A Web Services-oriented Architecture for Integrating Small Programmable Objects in the Web of Things. Developments in E-systems Engineering, 70–75.(2010)
4. Bouguettaya, A., Nepal, S., Sherchan, W., Zhou, X., Wu, J., Chen, S., Liu, D., Li, L., Wang, H., Liu, X.: End-to-End Service Support for Mashups. IEEE Transactions on Services Computing, Vol. 3, No. 3, 250-263.(2010)
5. Buyukozkan, G., Cifci, G.: A combined fuzzy AHP and fuzzy TOPSIS based strategic analysis of electronic service quality in healthcare industry. Expert System with Applications, Vol. 39, No. 3, 2341–2354. (2012)
6. Chatzigeorgiou, A., Stiakakis, E.: Combining metrics for software evolution assessment by means of Data Envelopment Analysis. Journal of Software Maintenance and Evolution: Research and Practice, Vol. 25, No. 3, 303–324.(2013)
7. Chen, J., Nairn, R., Nelson, L., Bernstein, M., Chi, E.: Short and tweet: Experiments on recommending content from information streams. In Proceedings the SIGCHI Conference on Human Factors in Computing Systems, 1185-1194.(2010)
8. Christophe, B., Verdot, V., Toubiana, V.: Searching the 'Web of Things'. In Proceedings of the Fifth IEEE International Conference on Semantic Computing, Palo Alto, CA, 308–313.(2011)
9. Duquennoy, S., Grimaud, G., Vandewalle, J.: The Web of Things: Interconnecting Devices with High Usability and Performance. In Proceedings of the International Conference on Embedded Software and Systems, 323 –330.(2009)
10. Duquennoy, S., Grimaud, G., Vandewalle, J.: Smews: Smart and mobile embedded web server. In Proceedings of the International Conference on Complex, Intelligent and Software Intensive Systems, 571– 576.(2009)
11. Eessaar, E., Soobik, M.: A Decision Support Method for Evaluating Database Designs. Computer Science and Information Systems, Vol. 9, No. 1, 81-106.(2012)
12. Fielding, R.T.: Architectural Styles and The Design of Network-based Software Architectures. Doctoral dissertation, University of California, University of California, Irvine.(2000)
13. Guinard, D., Trifa, V.: Towards the web of things: web mashups for embedded devices. In Proceedings of the International World Wide Web Conference, Madrid, Spain, 15.(2009)
14. Guinard, D., Ion, I., Mayer, S.: In Search of an Internet of Things Service Architecture: REST or WS-*? A Developers' Perspective. Mobile and Ubiquitous Systems: Computing, Networking, and Services, Springer Berlin Heidelberg, 326-337.(2012)
15. Guinard, D., Trifa, V., Karnouskos, S., Spiess, P., Savio, D.: Interacting with the SOA-Based Internet of Things: Discovery, Query, Selection, and On-Demand Provisioning of Web Services. IEEE Transactions on Services Computing, Vol. 3, No. 3, 223–235.(2010)
16. Guinard, D., Trifa, V., and Wilde, E.: Architecting a Mashable Open World Wide Web of things, Technical report, ETH Zurich,(2010). [Online]. Available: http://tinyurl.com/wotarchi (current December 2013)
17. Guinard, D., Trifa, V., Wilde, E.: A resource oriented architecture for the web of things. In Proceedings of the IEEE International Conference on Internet of Things, Tokyo, Japan, 1-8.(2010)

18. Guinard, D., Trifa, V., Mattern, F., Wilde, E.: From the internet of things to the web of things: Resource oriented architecture and best practices. Architecting the Internet of Things, Springer Berlin Heidelberg, 97-129. (2011)
19. Hristoskova, A., Volckaert, B., De Turck, F.: The WTE+ Framework: Automated construction and runtime adaptation of service mashups. Automated Software Engineering, Vol. 20, No. 41–4, 499-542.(2012)
20. Hoyer , V., Stanoevska-Slabeva, K., Kramer, S., Giessmann, A.: What are the business benefits of enterprise mashups? In Proceedings of the 44th IEEE Hawaii International Conference on System Sciences, 1–10.(2011)
21. Kahraman, C., Cebeci, U., Ruan, D.: Multi-attribute comparison of catering service companies using fuzzy AHP: The case of Turkey. International Journal of Production Economics, Vol. 87, No. 2, 171–184. (2004)
22. Karim, R., Ding, C.: An Enhanced PROMETHEE Model for QoS-Based Web Service Selection. In Proceedings of the 8th IEEE International Conference on Services Computing, 537-543. (2011)
23. Karpischek S., Michahelles, F.: My2cents- Digitizing consumer opinions and comments about retail products. In Proceedings of IEEE International Conference on Internet of Things, 1–7. (2010)
24. Li, Q., Qin, W., Han, B., Wang, R., Sun, L.: A Case Study on REST-Style Architecture for Cyber-Physical Systems: Restful Smart Gateway. Computer Science and Information Systems, Vol. 8, No. 4, 1317-1329. (2011)
25. Luckenbach, T., Gober, P., Arbanowski, S., Fokus, F., Kotsopoulos, A., Kim, K., Advanced, S., Sait, T., Box. P.: TinyREST - A Protocol for Integrating Sensor Networks into the Internet. In Proceedings of REALWSN, 101-105. (2005)
26. Makelainen S., Alakoski, T.: Fixed-mobile hybrid mashups: Applying the rest principles to mobile-specific resources. In Proceedings of the International Workshop on Web Information Systems Engineering, Springer-Verlag, Berlin Heidelberg, 172–182. (2008)
27. Mayer S., and Guinard, D.: An extensible discovery service for smart things. In Proceedings of the 2nd ACM International Workshop on the Web of Things, San Francisco, USA, No. 7. (2011)
28. Novak, J.:Ubiquitous computing and socially aware consumer support systems in the augmented supermarket. In Workshop on Social Implications of Ubiquitous Computing at CHI, Portland, Oregon, US. (2005)
29. Ostermaier, B., Romer, K., Mattern, F., Fahrmair, M., Kellerer, W.: A Real-Time Search Engine for the Web of Things. In Proceedings of Internet of Things IEEE International Conference, Tokyo, Japan, 1-8. (2010)
30. Omkarprasad, S.V., Sushil, K.: Analytic hierarchy process: An overview of applications. European Journal of Operational Research, Vol. 169, No. 1, 1–29. (2006)
31. Pautasso, C., Zimmermann, O., Leymann, F.: RESTful Web Services versus 'Big' Web Services: Making the Right Architectural Decision. In Proceedings of the 17th ACM International Conference on World Wide Web, 805-814. (2008)
32. Pautasso, C.: RESTful web service composition with BPEL for REST. Data & Knowledge Engineering, Vol. 69, No. 9, 851–866. (2009)
33. Pautasso, C,: Composing RESTful Services with JOpera, In Proceedings of the 8th International Conference on Software Composition, Springer-Verlag, Berlin Heidelberg, 142– 159. (2009)
34. Rümpel, A., Tietz, V., Wagner, A., Meißner, K.: Modeling and Utilizing Quality Properties in the Development of Composite Web Mashups. Current Trends in Web Engineering. Springer International Publishing, 54-65. (2013)
35. Saaty, T. L.: A scaling method for priorities in hierarchical structures. Journal of Mathematical Psychology, Vol. 15, No. 3, 234-281. (1977)
36. Sheng, Q.Z., Qiao, X., Vasilakos, A.V., Szabo, C., Bourne, S., Xu, X.: Web services composition: A decade's overview. Information Sciences, Vol. 280, 218-238. (2014)

37. Stirbu, V.: Towards a RESTful Plug and Play Experience in the Web of Things. In Proceedings of the IEEE International Conference on Semantic Computing. Washington, DC, USA, 512–517. (2008)
38. Su, H., Cheng, B., Wu, T. Li, X.: Mashup Service Release Based on SOAP and REST. In Proceedings of IEEE International Conference on Computer Science and Network Technology (ICCSNT), Vol. 2, 1091 – 1095. (2011)
39. Tran, V. X. Tsuji, H., Masuda, R.: A new QoS ontology and its QoS-based ranking algorithm for Web services. Journal on Simulation Modelling Practice and Theory, Vol. 17, No. 8, 1378-1398. (2009)
40. Trifa,V., Wiel, S., Guinard, D., Bohnert, T.:. Design and implementation of a gateway for web-based interaction and management of embedded devices. In Proceedings of the 2nd International Workshop on Sensor Network Engineering (IWSNE). (2009)
41. Vesyropoulos, N., Georgiadis, C.K., Ilioudis, C.: Analyzing the selection and dynamic composition of web services in e-commerce transactions. In Proceedings of the 5th Balkan Conference in Informatics, Novi Sad, Serbia, 130-135. (2012)
42. Xu, H., Song, M., Luo, X.: A QoS-oriented Management Framework for Reconfigurable Mobile Mashup Services. In Proceedings of 11th IEEE International Conference on Advanced Communication Technology. Gangwon-Do, South Korea, Vol. 3, 2001-2005. (2009)
43. Yan, B., Huang, G. W.: Supply Chain Information Transmission Based on RFID and Internet of Things. In Proceedings of ISECS International Colloquium on Computing, Communication, Control and Management, Sanya City, China,166-169. (2009)
44. Yang, W., Zhang, C., Mu, B.: An Approach to Optimize Mashup Based on QoS-Skyline and Transactional Properties. In Proceedings of International Conference on Information Science and Technology (ICIST), 878 – 882. (2013)
45. Zhang, Y., Liu, S., Wu, L., Jiao, Y.: Service-oriented Enterprise Interoperability in Automobile Supply Chain Management. Computer Science and Information Systems, Vol. 7, No. 1, 31-49. (2010)
46. Zhong, N., Ma, J.H., Huang, R.H., Liu, J.M., Yao, Y.Y., Zhang, Y.X., Chen, J.H.: Research Challenges and Perspectives on Wisdom Web of Things (W2T). The Journal of Supercomputing, Vol. 64, No. 3, 862-882. (2010)

Nikolaos Vesyropoulos received the B.Sc. and M.Sc. degrees in Computer Science from the Aristotle University of Thessaloniki, in 2008 and 2010, respectively. He is a Ph.D. candidate in the Department of Applied Informatics at the University of Macedonia under the supervision of Dr. Christos K. Georgiadis. In the context of his PhD he is investigating novel approaches for the discovery and composition of Web Services to support e-Commerce transactions.

Christos K. Georgiadis is an associate professor of e-commerce technology in the Department of Applied Informatics at the University of Macedonia, Thessaloniki, Greece. He received the BSc degree in Mathematics and the PhD degree in Computer Science from the Aristotle University of Thessaloniki, Greece, in 1987 and 2002 respectively. He has also completed post-graduate studies in Computer Science at the University of Pisa, Italy, in 1989. His research interests include the areas of e-commerce and m-commerce technologies, e-security, web programming and web services supporting technologies. He has published more than 60 articles in international journals, edited volumes and international conferences and has received more than 490 citations. He is a senior member of the ACM and SIGEcom.

Received: January 08, 2014; Accepted: June 30, 2014