Abstract—Recently practical approaches for development of ubiquitous semantic applications have made quite some progress. In particular in the area of the ubiquitous access to the semantic data we recently observed a large number of approaches, systems and applications being described in the literature. With this survey we aim to provide an overview on the rapidly emerging field of Ubiquitous Semantic Applications (UbiSA). We conducted a systematic literature review comprising a thorough analysis of 48 primary studies out of 172 initially retrieved papers. We obtained a comprehensive set of quality attributes for UbiSA together with corresponding application features suggested for their realization. The quality attributes include aspects such as mobility, usability, heterogeneity, collaboration, customizability and evolvability. The primary studies were surveyed in the light of these quality attributes and we performed a thorough analysis of six ubiquitous semantic applications, six frameworks for UbiSA, three UbiSA specific ontologies, six ubiquitous semantic systems and nine general approaches. The proposed quality attributes facilitate the evaluation of existing approaches and the development of novel, more effective and intuitive UbiSA.

I. INTRODUCTION

Recently practical approaches for the development of UbiSA that allow access to the Web of Data have made quite some progress. On the backend side, a variety of triple stores were developed and their performance and maturity improved steadily. With increasing power of ubiquitous devices it has become possible to use some of the triple stores on devices to allow offline access to the semantic data. Similarly tools and algorithms for processing and presenting data on ubiquitous devices are progressing and applications are deployed for the use on the emerging Web of Data. The quantity and quality of semantic content being made available on the Data Web is rapidly increasing, mainly due to the use of automated knowledge extraction techniques or due to the semantic enrichment and transformation of existing structured data. Despite many interesting showcases (e.g. Sindice, Parallax, or PowerAqua), we still lack more user friendly and scalable approaches for the exploration, browsing and search of semantic data. However, the currently least developed aspect of access to the semantic data is, from our point of view, the user-friendly ubiquitous applications that provide access to rich semantic content.

To define UbiSA, we must first specify what we mean by ubiquitous applications and semantic documents.

A guiding principle of ubiquitous applications is to break away from desktop computing to provide computational services to a user when and where required [41]. Ubiquitous applications are characterized by two main attributes [57]:

- **ubiquity**: interaction with the system is available whenever the user needs it;
- **transparency**: the system is non-intrusive and is integrated into the everyday environment.

Semantic documents are documents that consist of semantic data and describe specific entities or collections of entities. Semantic data on the other hand is the data that is defined and linked in a way that it can be used by machines not just for display purposes, but for automation, integration, and reuse of data across various applications. Semantic data should provide a basis for coding, exchanging, and reusing structured metadata among applications exchanging machine understandable information on the Web.

Taking all of the above into account, we define **ubiquitous semantic application** as the computer software implemented specifically for ubiquitous devices and designed to help the user to perform specific tasks that satisfy the following requirements:

- the application is designed and developed specifically for (or with respect to) ubiquitous devices,
- the application utilizes semantic data during the work process in any way (e.g. executing SPARQL queries, reading or writing RDF triples).

A ubiquitous semantic application provides a human accessible interface with capabilities for reading, writing or modifying semantic documents. Semantic documents facilitate a number of important aspects of information management:

- For **search and retrieval**, enriching documents with semantic representations helps to create more efficient and effective search interfaces, such as faceted search (e.g. in [15]) or question answering. Ultimately, users are empowered to fight the increasing information overload and gain better access to relevant documents and answers related to their information needs.
- For **information presentation**, semantically enriched documents can be used to create more sophisticated ways of flexibly visualizing information, such as geospatial maps as described in [54], [6], [58].
• For information integration, semantically enriched documents can be used to provide unified views on heterogeneous data stored in different applications by creating composite applications such as semantic mashups, like ones presented in [58], [15].

• To realize personalization, semantic documents provide customized and context-specific information which better fits user needs and will result in delivering customized applications such as personalized semantic portals (e.g. [37], [50]).

• For reusability and interoperability, enriching documents with semantic representations (e.g. using the SKOS and Dublin Core vocabularies) facilitates exchanging content between disparate systems.

There are already many approaches, frameworks and tools available for ubiquitous devices which address different aspects of this tasks. Due to the wealth of different approaches emerging, it is crucial to obtain an overview on the advancement in this emerging field. Furthermore, having a holistic view on approaches and tools provides us with an exhausting set of quality attributes, which are important for conceiving guidelines for developing more effective and intuitive UbiSA. Since most of the current approaches and applications are developed for smartphones (e.g. [49], [54], [58], [5], [33]), these quality attributes will be especially important for new and upcoming ubiquitous devices such as application-enabled smart TVs, gaming consoles, wearable computers, etc. UbiSA development for tablet computers that are also gaining popularity and predicted to outnumber PCs in the next few years can also benefit from defined quality attributes.

In this article, we summarize the findings of a systematic literature review on UbiSA. We extract different types and properties of applications proposed for ubiquitous use. The results reveal a set of quality attributes which can be used for classification of UbiSA. Furthermore, we report on the suggested application types and features proposed in the literature to realize these quality attributes.

The rest of the paper is organized as follows. In Section II we explain our motivation. In Section III we describe the research method and the review protocol used for conducting the systematic review. In Section IV we first define the terminology of the paper then we elaborate on the results of the review by surveying the extracted quality attributes. In Section V we discuss three existing UbiSA and describe them in the light of the quality attributes. In Section VI we report on the gaps and open research issues revealed from the results of our systematic literature review. Finally, in Section VII we conclude and present some ideas for future work.

II. MOTIVATION

An increasing number of approaches, systems and applications is being presented in the literature for the area of the ubiquitous access to semantic data. It is important to provide a comprehensive overview of the existing approaches, systems and applications as well as to integrate them into a common conceptual scheme. Doing so will allow simple categorization of existing approaches and may help in the development of new approaches. Currently, there is no unified terminology or conceptual model for UbiSA. That makes categorization and evaluation of existing approaches difficult, since the used conceptual model and terminology has to be established separately for each approach.

With the plethora of new devices and device categories (e.g. phablets, smart watches, augmented reality glasses) the way people access the data changes. Usage shifts from traditional desktop computers to these new devices. This usage shift includes UbiSA access as well. Thus we need to identify challenges and promising research directions to have a complete vision of the research field comprising UbiSA. The main goal of this survey is analyzing existing semantic applications, approaches and systems for ubiquitous devices and providing a set of quality attributes which can serve as guidelines for designing suitable and effective semantic applications for ubiquitous devices as well as provide a simple but comprehensive model for the categorization of existing ones.

III. RESEARCH METHOD

We followed a formal systematic literature review process for this study based on the guidelines proposed in [14], [24]. A systematic literature review is an evidence-based approach to thoroughly search studies relevant to some pre-defined research questions and critically select, appraise, and synthesize findings for answering the research questions at hand. Systematic reviews maximize the chance to retrieve complete data sets and minimize the chance of bias. As part of the review process, we developed a protocol (described in the sequel) that provides a plan for the review in terms of the method to be followed, including the research questions and the data to be extracted.

A. Research Questions

The goal of our survey is analyzing existing semantic applications for ubiquitous devices and thereby providing a set of quality attributes, which can serve as guidelines for designing suitable and effective semantic applications for ubiquitous devices. To achieve this goal we aim to answer the following general research question:

What are the existing approaches for development of ubiquitous semantic applications?

We can divide this general research question into the following more concrete sub-questions:

• RQ1. How to classify existing approaches for development of ubiquitous semantic applications?

• RQ2. What type of applications are developed in each approach?

• RQ3. What are the features supported by the proposed application?

• RQ4. How is the application evaluated?

After doing some pilot searches and consulting experts in the field, we obtained a list of pilot studies which served as a basis for the systematic review.
B. Search Strategy

To cover all the relevant publications, we used the following electronic libraries:

- ACM Digital Library
- IEEE Xplore Digital Library
- ScienceDirect
- SpringerLink
- ISI Web of Sciences

Based on the research questions and pilot studies, we found the following basic terms to be most appropriate for the systematic review:

1) ubiquitous OR mobile
2) semantic OR linked data OR web of data OR data web
3) application OR software OR system

To construct the search string, all these search terms were combined using Boolean “AND” as follows:

1 AND 2 AND 3

The next decision was to find the suitable field (i.e. title, abstract and full-text) to apply the search string on. In our experience, searching in the “title” alone does not always provide us with all relevant publications. Thus, “abstract” or “full-text” of publications should potentially be included. On the other hand, since the search on the full-text of studies results in many irrelevant publications, we chose to apply the search query additionally on the “abstract” of the studies. This means a study is selected as a candidate study if its title or abstract contains the keywords defined in the search string. In addition, we limited our search to the publications that are written in English and are published after 2002 (when the first ISWC conference was held).

C. Study Selection

Some of the studies might contain the keywords used in the search string but might still be irrelevant for our research questions. Therefore, a study selection has to be performed to include only studies that contain useful information for answering the research question.

Peer-reviewed articles that satisfy all the following inclusion criteria are selected as primary studies:

- I1. A study that focuses on ubiquitous semantic applications
- I2. A study that either proposes an approach or a set of specific features for the purpose of accessing semantic content on ubiquitous devices.

Studies that met any of the following criteria were excluded from the review:

- E1. A study that does not focus on ubiquitous semantic applications but only mentions the term e.g. as an example or use case.
- E2. A study that does not propose any approach or specific features used in development of UbiSA.
- E3. A study that is not about semantic data (e.g. studies about semantics as the study of meaning).

The conduction of our search commenced in early September 2013. As a consequence, our review included studies that were published and/or indexed before that date. As shown in Figure 1, we first applied the search query on each data source separately. Subsequently, to remove duplicate studies, we merged the results obtained from the different data sources. To remove irrelevant studies, we scanned the articles by title and thereby reduced the number of studies to 172. Then, we read the abstract of each publication carefully and further decreased the number of studies to 92. Finally, we added a list of additional papers recommended by experts and then scanned the full-text of the publications. Experts recommended adding 3 papers that did not appear in the results during the search phase. We checked the full-text of studies to see if they fit with our predefined selection criteria. The result comprised 48 publications that represented our final set of primary studies.

D. Data Extraction and Analysis

The bibliographic metadata about each primary study were recorded using the bibliography management platform JabRef. In addition, we extracted the following information from each paper:

- used approach for UbiSA development
- type of application
- features supported by the application
- domain and type of user
- evaluation method used in the paper

To analyze the information appropriately, we required a suitable qualitative data analysis method applicable to our dataset. We used coding as our qualitative analysis method. A common method that is used for this purpose is the grounded theory method because the theories (the UbiSA approaches and application features) are “grounded” in the data [18].

Constant comparison method, one of the grounded theory techniques, has been often used in analyzing data and

---

Fig. 1. Steps followed to scope the search results.
generating categories of data. Although constant comparison method can be used on any set of data, it is particularly suitable for the data that are context sensitive \[43\] (i.e. data can be interpreted differently in different contexts). To interpret UbiSA approaches and application features correctly, one often needs to understand in which context the approach and feature is proposed and how it is addressed. For instance, consider one study that mentions “interoperability” as a feature for application. Without understanding the context of this feature, we cannot conclude whether this feature is about designing interoperable UIs or about supporting annotation/ontology interoperability.

Miles and Huberman \[30\] described coding as a procedure for the constant comparison method. Codes are tags or labels for assigning units of meaning to the descriptive or inferential information compiled during a study. Codes are efficient data-labeling and data-retrieval devices \[30\]. One method of creating codes (recommended in \[40\]) is that of creating a provisional “start-list” of codes prior to fieldwork. We created this list from our research questions and the pilot studies. To carry out the analysis systematically, we used the following coding procedures proposed by Lincoln and Guba \[30\]:

- **Filling-in**: we read each study carefully and added the codes for related fragments and items. As new insights or new ways of looking at the data emerged, we reconstructed our coherent coding schema.
- **Extension**: if needed, we returned to materials coded earlier and interrogated them in a new way, with a new theme, construct, or relationship.
- **Bridging**: if new or previously not understood relationships within units of a given category were found, we recorded that relationship.
- **Surfacing**: we identified new categories which contained the previously created codes.

We used the Weft QDA software\[7\] to record the codes. More detailed information on coding and usage of the Weft QDA software can be found in \[17\].

E. Overview of Included Studies

For quantitative analysis purposes, we performed some queries on the collected database of primary studies. The distribution of studies per year as shown in Figure 2 indicates an increasing intensity of research in the area of ubiquitous semantic applications. The remarkable rise after 2009 can be explained with the emergence and increasing adoption of new touch-enabled smartphones lead by Apple’s iPhone. The low number of publications in 2013 can be attributed by commencing our survey in September, when most conference proceedings were not yet published or properly indexed.

The primary studies included 29 conference papers, 11 journal articles, 4 workshop papers and 2 book sections. Among them, the following studies are survey papers \[40\], \[4\] which provide an overview of the approaches that unify semantic Web technologies with a number of different mobile operations.

http://www.pressure.to/qda/

IV. RESULTS

In this section we first define the basic concepts used in the paper and then elaborate on the results of our qualitative data analysis.

A. Terminology

A generalized architecture of ubiquitous semantic applications is depicted in Figure 3. The presented generalized architecture was conceived on the basis of reviewed approaches, systems and applications to reflect all possible variations of UbiSA. In general, any existing UbiSA can be described using this generalized architecture. The opposite is true as well – it is possible to create any UbiSA by instantiating the presented generalized architecture. In the sequel we describe the research context terminology as well as individual components and concepts in more detail.

a) Human Computer Interaction (HCI): is crucial for the development of ubiquitous semantic applications and represents a research field that aims at improving the interactions between users and computers by making computers more usable and receptive to the user’s needs. This field is particularly relevant for ubiquitous semantic applications, since ubiquitous devices such as smartphones, tablet PCs and smart TVs offer a wide range of novel interaction paradigms.

b) Frontend: is an abstraction, which simplifies the usage of underlying components by providing a user-friendly interface for HCI. This area is especially important for UbiSA development since ubiquitous devices might have several different ways to interact with the user (e.g. touch screen input, voice control, gestures control, camera input, etc). There are currently several standard approaches used to develop front-ends such as:

- **Native**. Natively developed frontends provide users with familiar experience since usually all of the control elements are generated by operating system thus resulting in a consistent look-and-feel. Native frontend development is performed using tools provided by the platform manufacturer and is thus limited to a specific platform. Hence, portability and consequently development efficiency are a main disadvantage of native frontend development \[9\].
- **Web.** Web frontends are gaining popularity with the increase of data connection speeds and improvements of web browsers for ubiquitous devices. Web frontends are built using web technologies such as HTML, JavaScript and CSS. This results in exceptional portability. However, there are still many nuances (e.g. different screen sizes, connectivity issues, hardware limitations) that need to be considered while developing a web frontend (cf. for an in-depth discussion of such issues). One of their main issues is the look-and-feel of the frontend, since it is not possible to provide a native user experience for all platforms from a single web frontend.

- **Abstract.** Abstract frontend development approaches, that are inspired by the model-driven development paradigm gained recently popularity. These approaches define an abstract frontend description and to transform the abstract model into a (usually limited) number of platform-specific frontends which provide a native look-and-feel for each supported platform. The Unified Interface Markup Language (UIML) as described in is an example. While providing native UI support for several platforms, abstract development approaches are usually very limited with regard to the customization of the frontend elements.

  - **Context-Awareness:** is a term coined to describe applications that can passively or actively determine their context by utilizing on-board or peripheral device sensors. Especially in mobile and ubiquitous usage scenarios context-awareness is a crucial aspect, since the device is used in very specific and distinct situations (e.g. while walking or sitting in a restaurant). Also, mobile and ubiquitous devices have a variety of sensors not available in desktop computers, such as GPS, accelerometer, gyroscope, compass, etc. (a detailed overview is provided in ). Hence, for applications supporting ubiquitous usage scenarios, it is of paramount importance to employ the additional sensing techniques for optimally supporting the users’ experience. Context-awareness includes two dependent parts: sensors and logic. Logic is processing data from sensors and facilitates decision making about the current user’s context.. describe various approaches in that regard. A survey on context data distribution for mobile ubiquitous devices is also described in . While sensors can only be located on the ubiquitous device, the logic layer can as well be moved to the server application layer (e.g. in thin client approach )

- **d) Ontology:** is a formal, explicit specification of a shared conceptualisation that represents knowledge as a set of concepts within a domain as well as relationships between those concepts. Ontologies can be used in various scenarios in the context of UbiSA:
  - **Ontology authoring** the UbiSA is used for the creation of ontologies on the ubiquitous device. For example, OntoWiki mobile (as described in Section V-A) is a comprehensive ontology and knowledge base authoring interface.
  - **Ontology use** in UbiSA can happen at the server side (e.g. ) or directly at the ubiquitous device (labeled as mOntology in Figure 3). Examples for the use of ontologies directly at the device are presented in , .

- **e) Reasoning:** is the act or process of deriving logical conclusions from premises known or assumed to be true. Depending on the UbiSA development approach, reasoning can be performed on the ubiquitous device itself (mReasoning on e.g. ) or on the server (e.g. ). Reasoning approaches being deployed directly on the device have to cope with resource restrictions (e.g. available memory capacity and processing power). Hence, most mobile reasoning engines currently provide only simple rule processing through forward/backward chaining. There is also an increasing number of studies that aim to develop scalable semantic reasoning techniques that are useful for both ubiquitous and standard service selection algorithms .

- **f) Data Replication:** is the process of exchanging information so as to ensure consistency between different UbiSAs or a UbiSA’s client and server application layers. Replication is crucial for any UbiSA since the data connection might be limited, unstable or not available at all. Maintaining data consistency while replicating data to the client and synchronizing changes to the server is especially important in the area of Social Semantic Web. This is because social user interactions usually involves much collaboration aiming at creating and changing data. Examples of approaches include:
  - Frameworks for selective replication of data sets on mobile devices. The goal of such frameworks is to provide access to data sets in situations without network connectivity when communication with remote data sources is impossible. Most frequently used technique is adding intermediate components that handle queries
transparently, either by forwarding them to the actual data store if connectivity is up, or by answering them from a locally cached partial replica of the data set on the mobile device, if there is no connectivity [42].

- Conflict resolution approach based on a combination of distributed revision control strategies as well as the data evolution and ontology refactoring [15].
- Domain-specific approaches that tries to combine two or more techniques. For example, combination of multi-resolution spatial data structure and semantic caching, aimed towards efficient spatial query processing in mobile environments as described in [48].

**g)** Triplestore: is a specific database for the storage and retrieval of information adhering to the RDF data model, i.e. triples composed of subject-predicate-object (e.g. "Bob is 35" or "Bob knows Fred"). Triplestores can also be implemented either on the ubiquitous device itself (mTriplestore on Figure 3) or on the server. [6], [58], [52] describe UbiSAs accessing triples stores on the server side. With the emerging of Android OS several triplestores (mostly developed in Java) were ported to the Android system. List of the most popular triplestores for ubiquitous devices is provided in Table I.

**h)** Server Application Layer (SAL): is an abstraction, describing the underlying logic of the UbI SA server. SAL can be used in both thin and fat client approaches (e.g. [45], [20]). However, SAL is more commonly used in thin client approaches to execute most of the required operations on the server. A fat client approach might as well use SAL to outsource expensive operations (e.g. reasoning) in order to decrease the usage of ubiquitous device resources.

**i)** Social Semantic Web: is a very general technology field triggered by the advent of Web 2.0. It aims at bringing a social novelty, rather than a technical one by providing user-friendly tools to facilitate broad user participation in the process of creating semantic content. Examples can be found in [24], [15]. The Social Semantic Web vision comprises many of the aforementioned domains and techniques. The Social Semantic Web is a crucial application domain for UbiSA.

### B. Possible User Roles

One more important aspect that needs to be considered in process of development of the UbiSA is possible user roles. User roles describe main focus in user interaction with the UbiSA.

**j)** Professional user: is a user that has extensive knowledge of how ubiquitous devices and semantic web applications work. Professional users usually have deep understanding of the UbiSA they use. Main purpose of using UbiSA is usually data gathering for research purposes (e.g. for the knowledge management project Caucasian Spiders [15]).

**k)** User seeking entertainment or social interactions: is a user that has little or no knowledge on semantic web applications and has only basic skills in using ubiquitous devices. Gamification, research field that is currently gaining popularity, can be used to engage such users into useful process that is also entertaining for them.

### C. Ubiquitous Semantic Applications Development Approaches

There are already a number of different approaches proposed for ubiquitous applications development but for non-semantic content (see [32], [28]) and a smaller number of approaches specifically for semantic content (e.g. [53]). These approaches aim at user gratification in the form of useful visualizations and interesting data aggregation but do not focus on using shared vocabularies and formal ontologies which ultimately facilitate customizability, portability and reuse. With regard to ubiquitous applications development recent approaches can be roughly classified into two categories: Fat Client and Thin Client. As demonstrated in Figure 3 the classification is based on the provided functionality independent of the server-side.

Overview of the comparison between thin and fat client advantages with regard to provided benefits is provided in Table II.

1) **Fat Client Approaches:** Fat client approaches, which are also called “rich client approaches”, aim to allow access to existing or create new semantically enriched data while using as many required features as possible on the ubiquitous device itself. The basic parts of an UbiSA are triple store, ontologies, reasoner, context-awareness service and front end that provides the user access to underlying components. In case of fat client approach UbiSA can have all of the described components on the device itself. A fat client still requires periodic connection to a network, but is often characterised by the ability to perform many functions without that connection. Fat client approach has the following advantages:

- **Fewer server requirements.**
  A fat client server does not require high level of performance since the fat client itself does the most of the application processing.

- **Offline working.**
  Fat clients have advantages over thin clients in that a constant connection to the server is often not required.

- **Better multimedia performance.**
  Fat clients have advantages in multimedia-rich applications that would be bandwidth intensive, if fully served.

- **More flexibility.**
  On some operating systems software products are designed for ubiquitous devices that have their own local resources.
More flexibility

μSesame Core was ported to Android as part of blueBill Mobile project. RDF On The Go is the project to build a persistent RDF store and query processor on Android.

Offline working

Using existing infrastructure

Higher server capacity

Fat Client

Androjena is a porting of Hewlett-Packard’s Jena semantic web framework to the Google Android platform.

RDF On The Go

TriplePlace

OpenSesame

μJena

Androjena

http://code.google.com/p/androjena/

RDF On The Go

http://code.google.com/p/rdfonthego/

TriplePlace

https://github.com/white-glasses/TriplePlace

OpenSesame

http://bluebill-idatalabствие/mobile/

μJena

http://poseidon.ws.dei.polimi.it/ca/?page_id=9

μJena is a reduced, lightweight porting of the Jena API for Android.

TABLE I. LIST OF TRIPLESTORES FOR UBIQUITOUS PLATFORMS.

<table>
<thead>
<tr>
<th>Triplestore</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Androjena</td>
<td>Androjena is a porting of Hewlett-Packard’s Jena semantic web framework to the Google Android platform.</td>
</tr>
<tr>
<td>RDF On The Go</td>
<td>RDF On The Go is the project to build a persistent RDF store and query processor on Android phone.</td>
</tr>
<tr>
<td>TriplePlace</td>
<td>TriplePlace a light weight and flexible Triple Store for Android. It uses a indexing structure similar to the one in Hexastore. TriplePlace uses TokyoCabinet as persistent storage system.</td>
</tr>
<tr>
<td>OpenSesame</td>
<td>OpenSesame Core was ported to Android as part of blueBill Mobile project.</td>
</tr>
<tr>
<td>μJena</td>
<td>μJena is a reduced, lightweight porting of the Jena API for Android.</td>
</tr>
</tbody>
</table>

The downsides of doing all the work on the servers are high resource consumption (and as result decrease in device’s battery life) and depending on task demand for better hardware.

2) Thin Client Approaches: Thin client approaches aim to allow access to existing or create new semantically enriched data on the remote servers while using ubiquitous device as an input-output terminal for user interaction and sensors. In case of thin clients, application on ubiquitous device can have only frontend for the user, context-awareness sensors and means to exchange data with the server, thus being more independent from device’s resources and hardware but more sensitive to data connection quality. The exact roles assumed by the server may vary, from providing data persistence (for example, for diskless nodes) to actual information processing on the client’s behalf. Thin client approach has the following advantages:

- **Using existing infrastructure.**
  As many people now have very fast ubiquitous devices, they already have the infrastructure to run fat clients at no extra cost.

- **Higher server capacity.**
  The more work that is carried out by the client, the less the server needs to do, increasing the number of users each server can support.

The downsides of doing all the work on the ubiquitous device are high resource consumption (and as result decrease in device’s battery life) and depending on task demand for better hardware.

The following features are proposed for improving the ubiquity of the applications:

- **Cross-device Compatibility.**
  Application should be able to work on the device most relevant to the user, dynamically adjusting to the device’s specific features.


- **Device-dependent UIs.**
  Device-dependent UIs allow the generation of different views on the same data and aggregations of the knowledge base based on the ubiquitous device parameters, personal preferences, and local policies of the intended users. Such views can be either generic or domain specific. Generic views provide visual representations of data according to certain property values (e.g., map view or calendar view). Domain specific views address the requirements of a particular domain user (e.g., chemists need specific views for visualizing the atomic structure of chemical compounds).

  2) **Usability:** Usability is a measure of the quality of a user’s experience in interacting with an application. In [27], usability is defined as consisting of the six factors:

  (a) **Fit for use (or functionality).** The application can support the tasks that the user has in real life.

  (b) **Ease of learning.** How easy is the application to learn for various groups of users?

  (c) **Task efficiency.** How efficient is it for the frequent user?

  (d) **Ease of remembering.** How easy is it to remember for the occasional user?

  (e) **Subjective satisfaction.** How satisfied is the user with the application?

  (f) **Understandability.** How easy is it to understand what the application does? This factor is particularly important in unusual situations, for instance error situations or system failures. Only an understanding of what the application does can help the user out.

  **Ease of use** (or user friendliness) is defined as the combination of factors (b) to (f).

  Simplicity is the main prerequisite of usability. An UbiSA should, as a rule, hide technical concepts and ontologies from the end users as well as provide (if possible) native for the user’s device way of interaction with data. It is crucial to provide end-users with easy to use interfaces that simplify the interaction process and place it in the context of their everyday usage. More attention needs to be paid to decrease or blur the gap between the normal interaction process and the interaction process with the semantic content. Ubiquitous semantic applications should focus on users main task. Usually, a user wants to perform the task of browsing or writing data with regard to current context.

  The following features are proposed for improving the usability of UbiSAs:

  - **Single Point of Entry Interface.** It means the environment in which users interact with the semantic data should be integrated (or adjusted to be as close as possible) with the one in which they usually interact with any other data. So, there is no added user effort involved in interacting with a semantic content versus a conventional approach.

  - **Faceted Browsing.**
    Faceted browsing is a technique for accessing a collection of information represented using a faceted classification, allowing users to explore by filtering the available information. In the UI which implements this technique, all property values (i.e., facets) of a set of selected instances are analyzed. If for a certain property the instances have only a limited set of values, those values are offered to further restrict the instance selection. Hence, this way of navigation through data will never lead to empty results [2]. This feature is useful when searching for available resources or vocabularies especially for ubiquitous devices with limited screen sizes.

  - **Inline Resource Editing.**
    Inline editing allows editing items by clicking on them. Such behaviour allows to minimize amount of information on screen which is extremely important on ubiquitous devices.

  3) **Customizability:** Customizability is the ability of an application to be configured according to users’ needs, preferences or context. Instead of being a static form strictly dependent on a given schema, UbiSA should provide a mechanism to tailor its functionalities based on the user’s needs or context.

    The following features are proposed for improving the customizability of UbiSA:

    - **Living UIs.**
      A Living UI is a user interface that configures itself to automatically display the information most relevant to the user, dynamically adjusts to changing data, and still allows single users to customize according to their preferences and context [40]. End-user development techniques like personalized UIs allow inferring user intents as well as present context in real interactions and according to that providing customized outputs.

    - **Providing Device-dependent UIs.**
      Device-dependent UIs allow the generation of different views on the same data and aggregations of the knowledge base based on the ubiquitous device parameters, personal preferences, and local policies of the intended users. Such views can be either generic or
Adaptivity reduces the costs of supporting new schemata considerably, by valuable it is for different contexts and users. A generic UbiSA fact, the more flexible and adaptable an application is, the more support a wide range of metadata schemata in a flexible way. In classification to adapt to different situations or use cases. UbiSA may change the ontology prior to distribution. In most adoption assumes changes of underlying ontology during the ontology at any time during UbiSA execution. Compile time heterogeneity of UbiSA: is in opposition to Usability of an application. For instance, adding more and more editing possibilities counteracts ease of use for UbiSA.

The following features are proposed for improving the heterogeneity of UbiSA:

- **Supporting Multiple Ontologies.** A domain is usually described by several ontologies. For example, in a medical context there may be one ontology for general metadata about a patient and other technical ontologies that deal with diagnosis and treatment. UbiSAs need to be able to support multiple ontologies. In a heterogeneous UbiSA, the user interface must be completely decoupled from the ontological models. It should be possible to add models at runtime and become immediately accessible to the users.

- **Supporting Ontology Modification.** A heterogeneous UbiSA should provide users with user-friendly interfaces to navigate or modify the structure (classes and properties) of ontologies. In this case, the application also needs to deal with consistency issues which might arise between ontologies and annotations with respect to ontology changes (a.k.a. Ontology Maintenance).

5) **Collaboration:** Collaboration refers to the ability of a application to support cooperation between different users of the system. UbiSA can support collaborative semantic authoring, where the authoring process can be shared among different authors at different locations. This is a key requirement of knowledge sharing between users from different fields who are contributing to and reusing intelligent documents. Web applications and related technologies provide incentives to their users for collaboration and lead to rapidly growing amounts of content. Triggered by the success of the Web 2.0 phenomenon, the Social Semantic Web idea has gained momentum yielding tools that allow collaboration and participation incorporating semantics by lay users (e.g. [1]). As a result, many collaborative and community-driven approaches to semantic content creation have been proposed. Examples are SemanticWikis and Semantic Tagging Systems (e.g. OntoWiki Mobile) which exploit Web 2.0 principles and technologies to facilitate broad user participation and collaboration in the process of creating semantically enriched or annotated content.

Access control and supporting standard formats are two additional independent prerequisites of collaboration in a ubiquitous semantic applications. The ubiquitous semantic application should allow to distinguish between writeable and non-writeable content based on the users permission level. It also needs to support standard formats which promote the collaboration and make it possible to share and re-use the generated content.

To realize collaboration, UbiSA should provide appropriate UI elements for meta-level interactions around different types of semantically created content such as rating, tagging and discussing. Supporting social networking features such as following other authors, watching the evolution of content as well as reusing and re-purposing of content are also important to increase the collaboration in UbiSA.

6) **Accessibility:** Accessibility describes the degree to which an application is available to as many people as possible. It can be viewed as the ability to access and benefit from some application. Accessibility is often used to focus on people with disabilities or special needs and their right of access the application. As mentioned in [21], papers discussing accessibility are clearly lacking in the context of Semantic Web UIs. Accessibility is especially important in area of ubiquitous devices since they can provide Multimodal User Interfaces (e.g. voice recognition, text to speech, gestures) that might be suitable for different contexts or people with disabilities.

7) **Evolvability:** Evolvability is defined as the capacity of a system for adaptive evolution. UbiSA should support evolution of the used data. To achieve this goal, it should take into account the following consistency constraints:

- **Resource Consistency.** In cases where several users may edit the same resources, replication issues may occur. It is especially important if the UbiSA allows offline work without synchronization in between client-server data exchange sessions. To address this, UbiSA should use data replication techniques (e.g. [42], [15]) to safely merge data from different users. Otherwise, data from some of the clients might be lost in the process of synchronization.

- **Document and Annotation Consistency.** One of the important issues for the design of a semantic authoring environment is to determine how changes should be reflected in the knowledge base of annotated documents and whether changes of ontologies create conflicts with existing annotations [15]. Ontologies change sometimes but some documents change many
times. So, it is crucial for a ubiquitous semantic application to track data evolution.

UbiSA should provide appropriate UIs and frameworks for versioning and change tracking to deal with data evolution.

8) Interoperability: Interoperability is the ability of an application to work and interact with other systems. A ubiquitous semantic application could provide mechanisms to interoperate together with other systems which generate or consume the semantic content created. The following features are proposed for improving the interoperability of ubiquitous semantic applications:

- **Support of Standard Formats.**
  To minimize the problems of interoperability the ubiquitous semantic application should be built on standards. There are already many standards for semantic content serialization (e.g. typical RDF serializations and particular JSON-RDF), representation (e.g. RDF/RDF-S/OWL/RIF and established vocabularies such as SIOC, SKOS, FOAF, rNews, etc.) and exchange (e.g. Linked Data, Web Services, REST). Supporting standard formats and avoiding proprietary formats are essential for compatibility of data with other systems [15], [49].

- **Semantic Syndication.**
  Semantic syndication supports the distribution of information and their integration into other applications by providing mechanisms such as Semantic Atom [34] and Semantic Pingback[2].

9) Scalability: Scalability refers to the capability of an application to maintain performance under an increased work load. UbiSA should support scalability as, for example, the number of users, data or annotations increase. Support of caching and implementing a suitable storage strategy play an important role in achieving a scalable UbiSA [2], [15], [49]. Most of the current UbiSA adopt a variety of replication frameworks. In this case, replication framework handles replication of required data to the ubiquitous device for later (e.g. offline) use. A replication framework sometimes poses a redundancy but allows information from heterogeneous resources to be queried centrally and, if it is supported by the framework, even in offline mode.

E. Quality Attributes Dependencies

The aforementioned quality attributes are not completely isolated and independent from each other but have overlaps and relations with each other. Figure 4 shows an overview of these quality attributes with their inter-relations. Customizability will improve the usability of UbiSA. Customizable applications are configured based on the user needs thereby increasing the overall usability of the application.

Scalability will enhance the level of collaboration since scalable application will support more users and data thereby more collaboration in the system. Interoperability will also enhance the collaboration support of an application, since an interoperable application supports users of different devices. It can also support importing user’s data from other devices or systems which will play a positive role in enhancing the customizability. Mobility will also enhance the collaboration support of an application, since an ubiquitous application supports users of different devices.

Evolvability and heterogeneity are directly related. The more evolvable to change an application is, the more heterogeneous it will be and vice versa. Customizability and heterogeneity share a reciprocal relation. A heterogeneous application will decrease its customization and a customizable application needs to focus on specific user needs and thus lacks heterogeneity.

F. Applications Evaluation

In this section we briefly outline various methods for applications evaluation and report about their usage in the surveyed papers. Table IV lists existing methods for application evaluation adopted from [10].

Among the primary studies, the majority of studies (24) were using an *Example Application* as their evaluation method. Discussion method was used by variety of papers (13) mostly related to the ontology creation topic. Other papers (11) were focused on algorithms development and thus used *Experiment with Software Subjects* method. The following UbiSA were described in the primary studies: mSpace Mobile [58], BuddyAlert [19], OntoWiki Mobile [15], MSSW [49], myCampus [44], PhotoMap [54] and csxPOI [6]. These studies were selected to cover as many quality attributes and parts of the generalized architecture as possible.

V. APPLICATIONS

In this section we look at six available UbiSA and compare them according to the quality attributes defined in Section IV-D. Among the applications five (i.e. Ontowiki Mobile, csxPOI, mSpace Mobile, myCampus and Bottari) follow the thin client approach (cf. Section IV-C2) and one (i.e. MSSW) follows the fat client approach (cf. Section IV-C1) for UbiSA. Figure 5 summarizes the assessment of the applications according to the defined quality attributes.

A. OntoWiki Mobile

OntoWiki Mobile[10] [15] is an application that provides support for agile, distributed knowledge engineering scenarios in ubiquitous environments. Ontowiki Mobile facilitates the visual presentation of a knowledge base as an information
The result has been used on real examples, but not in the form of case studies or controlled experiments, the evidence of its use is collected informally or formally.

The application is implemented as a HTML5 thin client aimed to run in web browser of a ubiquitous device. The backend is developed in PHP using the Zend framework. The backend is powered by the Erfurt framework provides support for MySQL database and the Virtuoso triple store as storage backends. The user interface is built using jQuery Mobile framework.

![Figure 6](http://caucasus-spiders.info/)

![Figure 7](http://jquerymobile.com/)

The architecture of csxPOI is shown in Figure 8. The application is implemented as a native Android thin client aimed to deliver access to database of collaboratively created POIs from the ubiquitous device. The backed of csxPOI is organized in a two-tier server architecture. The server consists of an abstraction layer for the collaborative ontology of POIs and provides user management. It also provides a POI revision engine to improve the quality of collaboratively created POIs. The server is implemented as an Apache Tomcat servlet handling the communication with the mobile clients and the triplestore over HTTP. The triplestore is realized as a Sesame web application on top of the same Apache Tomcat web server.

![Figure 8](http://isweb.uni-koblenz.de/Research/systeme/csxPOI)

The csxPOI, as an application with a single point of entry UI, adopts the thin client approach for UbiSA. It provides a semantic search feature with support for faceted browsing. It also supports two complementary knowledge base authoring strategies: a) Inline editing, which enables users to edit small information chunks (i.e. statements). b) View editing, which enables users to edit common combinations of information (such as an instance of a distinct class) in one single step. In order to do so, OntoWiki Mobile uses the version of RDFAuthor specially adapted for ubiquitous devices to make generated RDFa views editable. Regarding the customizability, OntoWiki Mobile supports different semantic views of the knowledge base which can be generic or domain-specific. It also supports editing multiple ontologies including both the instances and structures of the ontologies. As a Web-based application, it provides cross-browser compatibility but does not provide adoption of UI for different ubiquitous operating systems.

OntoWiki Mobile also provides versioning and evolution features to track, review and selectively roll-back changes which is really important for collaborative ubiquitous applications that might have problems while merging resources from different users. It also supports semantic syndication (employing Semantic Pingback and Linked Data interfaces) to interoperable with other systems. OntoWiki Mobile is backend independent to some extent and supports two different types of storage engines. It also provides a caching component to optimize the performance of the system.

As a drawback, OntoWiki Mobile does not provide any UI elements to facilitate accessibility and automation. It supports only the editing of structured content thus lacking UIs for the annotation of unstructured or semi-structured content.

### B. csxPOI

csxPOI (short for: collaborative, semantic, and context-aware points-of-interest) is an application that allows its users to collaboratively create, share, and modify semantic points of interest (POIs) in ubiquitous environments. It is made for engaged users who want to collaboratively create and share location-based data.

The csxPOI, as an application with a single point of entry UI, adopts the thin client approach for UbiSA. It provides a way to collaboratively create, share, and modify semantic POIs. While working with the POIs, the users implicitly and collaboratively modify and improve an ontology of POI categories underlying the application.

Since collaboratively created semantic POIs inevitably introduce a significant amount of inaccuracy, inconsistency, and redundancy, the csxPOI application provides a revision engine that clusters POIs with combinations of spatial, linguistic, and semantic similarity measures in order to identify and clean duplicate POIs.

---

As a drawback, csxPOI does not provide any compatibility with other devices except Android-based ones and there is no support for customizability. csxPOI does not provide any UI elements to facilitate accessibility and automation. It also works only in one specific domain.

C. mSpace Mobile

mSpace Mobile ([58]) is a Semantic Web application that lets people explore the world around them by leveraging contexts that are meaningful to them in time, space and subject. Especially applicable to those unfamiliar with their surroundings, the application provides information about topics of chosen interest, based upon the location, as determined by an optional GPS receiver.

Architecture of mSpace Mobile is shown in Figure 10. The application is implemented as native Windows Mobile thin client aimed to deliver access to database of topics of interest from ubiquitous device. mSpace Mobile is organized in a three-layer architecture: the mSpace Mobile application (MA) queries the mSpace Query Service (MQ), which is connected to RDF triplestore knowledge interfaces (MK). The architecture is designed to query multiple triplestores, as well as support incorporation of resources, which may not yet be referenced in triplestores. It also abstracts the internal concepts of query generation and triplestore querying, taking load away from ubiquitous devices. Figure 11 shows a screenshots of mSpace Mobile.
Fig. 7. Screenshot of the OntoWiki Mobile. (a) instance view, (b) inline editing, (c) device camera access (from [15]).

Fig. 8. Architecture of csxPOI (from [6]).

The mSpace Mobile, as an application with a single point of entry UI, adopts the thin client approach for UbiSA. It provides a way to access the location-based information while on the move. The mSpace Mobile interface is designed to let users of ubiquitous devices run complex queries through direct manipulation without typing.

The five key features of the mSpace Mobile interaction model are:

- **A spatial browser**, maintaining persistent display of domain dimensions, for browsing information within a domain.
- **User-determined organization** of dimensions presented: they can be added, removed and swapped.
- **Information area**, providing contextual information about selected items in the column browser.
- **Preview cues**, which provide typical examples of information within a domain.
- **Triage area**, allowing the user to save items from the domain that are of particular interest for further exploration in the future.

As a drawback, mSpace Mobile does not provide any compatibility with other devices than Windows Mobile-powered ones as well as there is no customizability support provided in any way. mSpace Mobile also works only in one specific domain and does not allow any editing of the data (except for rating the results).

**D. myCampus**

myCampus [44] is a context-aware semantic web environment aimed at enhancing everyday campus life. In myCampus, users can, over time acquire or subscribe to a variety of different sets of task-specific agents that assist them in context of different tasks (e.g. scheduling meetings, sharing documents, organizing evenings out, filtering and routing incoming messages, etc.). It is made for users aiming to learn or access information depending on the context.

The architecture of myCampus is shown in Figure 12. In myCampus, sources of contextual information are represented as Semantic Web Services. This means that each source of contextual information is described by a profile that describes its functional properties in relation to one or more ontologies. Service descriptions also include details about how to invoke a service (e.g. input, output and preconditions).

myCampus, as an application with a single point of entry UI, adopts the thin client approach for UbiSA. It provides an environment, where relevant sources of contextual information about a user can automatically be discovered and accessed in
support of different queries. This approach makes it possible to accommodate users that rely on different sets of contextual resources (e.g. different calendar systems, different sources of location information, etc.) and to adapt to situations where sources of contextual information for a user may change over time (e.g. different location tracking services depending on where the user is). myCampus agents can range from simple agents that rely on one or more sources of contextual information about their users to more complex agents that are capable of dynamically building plans in response to requests from their users.

As a drawback, myCampus does not provide any UI elements to facilitate accessibility and automation as well as there is no customizability support provided in any way. myCampus also works only in one specific domain and does not allow any editing of the data.

E. MSSW

MSSW\footnote{\url{http://aksw.org/Projects/MobileSocialSemanticWeb}} is an Android-based Social Semantic Web client as well as a contacts provider, which integrates the...
distributive FOAF/WebID social network into a ubiquitous device. It is made for users aiming to access information from their social graph.

The application is implemented as a native Android fat client. The overall MSSW architecture is depicted in Figure 14. It integrates a number of vocabularies, protocols and technologies. The semantic representation of personal information is facilitated by the WebID protocol and vocabulary. FOAF+SSL allows the use of a WebID for authentication and access control purposes. Semantic Pingback facilitates the first contact between users of the social network. The subscription services based on PubSubHubbub allow obtaining specific information from people in ones social network as near-instant notifications. The MSSW application itself consists of two application frameworks, which are built on top of the Android runtime and a number of libraries. In particular, androjena is one of those libraries, which itself is a partial port of the popular Jena framework to the Android platform (cf. Table I). MSSW uses the included Jena rule engine to transform the retrieved WebID statements into Android-specific data structures which are well suited for a straightforward import into the Android contacts provider.

Figure 15 shows screenshots of MSSW. MSSW provides a way to traverse a user’s FOAF network, synchronize the user’s contact with contact book on the device, add/remove relations or search for profiles using the Sindice search engine. As a drawback, MSSW is not compatible with non-Android devices. Also, there is no customizability support provided in any way. MSSW works only in one specific application domain but it is possible to adjust the data with custom Jena rules.

F. Bottari

Bottari is an Android-based application that exploits social media and context to provide point of interest (POI) recommendations to user in a specific geographic location. It is made for users aiming to discover nearby points of interest on the basis of user’s tastes and influencing people’s opinion.

The application is implemented as a native Android thin client. The overall Bottari architecture is depicted in Figure 16. It integrates a number of vocabularies, protocols and technologies. Semantic information retrieval is used to get interesting POIs close to user’s location. Sentiment analysis of posts from social media is used to get nearby POIs that are popular among other people. Stream reasoning is used to get emerging POIs that are getting a lot of traction. Finally, inductive
reasoning on social media is used to compute personalized recommendations. The Bottari application itself consists of an application framework that is built on top of the Android runtime and a number of libraries. In particular, androjena is one of those libraries, which itself is a partial port of the popular Jena framework to the Android platform (cf. Table I).

Figure 17 shows screenshots of Bottari.

Bottari provides a way to get personalized POI recommendations on Android tablets, to help users find their way when they are in a specific location. As a drawback, Bottari is not compatible with non-Android devices. Also, there is no customizability support provided in any way. As well Bottari works only in one specific application domain.

VI. RESEARCH AND TECHNOLOGY CHALLENGES

The results of our systematic review revealed several research and technology gaps and corresponding challenges with regard to the development of UbiSA. To the best of our knowledge, none of the challenges outlined in this section was so far addressed in existing research in any way.

a) Accessibility: Addressing accessibility issues during the design of UbiSA UIs and providing special UIs that utilize different input methods available on ubiquitous devices are very important aspects of UbiSA development. Addressing these issues can be beneficial not only for people with disabilities and special needs, but also for people that access UbiSA in special contexts (e.g. using voice control while driving).

b) Semantic disambiguation: Usage of Linked Data as background knowledge for disambiguation can simplify user interactions with ubiquitous devices in several ways. For example, search can be significantly improved and simplified by mapping keywords (in text or voice input) to URIs and use them for creating formal queries (e.g. in SPARQL).

c) Facilitating entertainment: Serious games on ubiquitous devices can bring entertainment to people seeking it while solving real problems. The challenge is to define game templates, which are tailored to mobile devices, employ or producing semantics and can be played in a casual way (e.g. while waiting for a bus). For example, a paper chase game could use background knowledge about points-of-interest such as LinkedGeoData, for generating verification tasks (e.g. location, opening hours etc.).

d) Making complex algorithms accessible: Bridging between complex algorithms and easily accessible functionality is one more important research field. The amount of complex algorithms developed by various research communities to solve different problems is increasing. But there is only a small amount of research on how to make those complex algorithms
easy for users to interact with. One possible approach for addressing this issue, could be a market place for algorithms and UIs. The market place could provide clearly defined interfaces for interacting with information in certain application scenarios. Different information object recognition algorithms (e.g. specialized on faces, buildings etc.), for example, could be dynamically plugged-into a UbiSA UI providing image annotation to a user.

e) Augmented reality: Augmented reality is one more gap in the research. With advancements of ubiquitous device’s camera sensors and image recognition technologies it is becoming possible to detect the context and display the information about surroundings in better way. Projects like Layar\(^2\) or Google Goggles\(^3\) can benefit from semantic datasets like LinkedGeoData\(^4\) or Europeana\(^5\) for augmenting reality on ubiquitous devices with semantic background information.

VII. CONCLUSIONS

In this paper we reported the results of a systematic literature review on ubiquitous semantic applications comprising

\(^2\)http://www.layar.com/
\(^3\)http://www.google.com/mobile/goggles/
\(^4\)http://europeana.eu/
Fig. 17. Screenshots of BOTTARI: (a) augmented reality display of recommended POIs, (b) POI selection and (c) visualization of the selected POI details, (d) trends in user sentiment about the POI.

initially of 172 papers. The review aimed to answer the four research questions defined in Section III-A by thorough analysis of the 48 most relevant papers. Before addressing the defined research questions, we drew a terminology which defines the basic concepts used in the literature as well our survey. To answer the RQ1, we classified existing approaches for ubiquitous semantic applications into two categories namely Fat Client and Thin Client approaches discussed in Section IV-C. Furthermore, Our data analysis revealed a set of 9 quality attributes for ubiquitous semantic applications together with the corresponding features which are suggested for their realization. These quality attributes plus the features are used to answer the RQ2 and RQ3. In order to answer RQ4 we extracted the main user roles as well as evaluation methods discussed in the primary studies and reflected the results in Section IV-F. Additionally, to show the applicability of the results, we performed an in-depth comparison of six existing ubiquitous semantic applications according to the defined quality attributes and described their strengths as well as their weaknesses.

Essentially, foundational quality attributes for a ubiquitous semantic applications are, in particular, mobility, usability, heterogeneity, customizability and evolvability. A basic ubiquitous semantic application should fulfill a reasonable level of user-friendliness and adopt to different situations or use cases while providing mechanisms to tailor its functionality based on specific user needs. It should also take into account issues such as resource consistency over different clients as well as cross-device compatibility. Support of collaboration, interoperability and scalability are quality attributes required when UbiSA is employed in a community-driven environment with large amount of users, systems and interactions. A UbiSA should support standard formats and provide appropriate UI elements for social networking including both human-to-human as well as system-to-system interactions. Additionally, it should maintain performance under an increased work load by supplying appropriate storage and caching mechanisms. Also, it should take into account possible issues with data connection and hardware related ubiquitous device limits. Accessibility is, as our survey indicates, not well addressed by the literature so far. Furthermore, providing accessible UIs for people with disabilities or special needs is another requirement which should be taken into account when designing ubiquitous semantic applications.

While there are many benefits of systematic reviews, they also bear some limitations and validity threats originating from human errors. The main threats to validity of our systematic review are twofold: correct and thorough selection of the studies to be included as well as accurate and exhaustive selection of quality attributes together with their corresponding features. With the increasing number of works in the area of ubiquitous semantic applications, we cannot guarantee to have captured all the material in this area. The scope of our review is restricted to the scientific domain. Therefore, some tools or approaches employed in the industry might have not been included in our primary studies. Furthermore, since the review
process was mainly performed by one researcher, a bias is possible. In order to mitigate a potential subjective bias, the review protocol and results were checked and validated by a senior researcher and other colleagues experienced in the context of Semantic Web.

We see this effort and in particular the identification of a comprehensive set of quality attributes as a crucial step towards developing more effective and user-friendly ubiquitous application for accessing the Social Semantic Web. New approaches and applications can be evaluated in the light of these quality attributes, thus revealing additional aspects to be taken into consideration. As a result, more user-friendly applications will enable more people to interact with the Semantic Web thereby facilitating the realization of the intelligent Web vision.

VIII. ACKNOWLEDGMENTS

We would like to thank our colleagues from AKSW research group for their helpful comments and inspiring discussions during the conducting of this systematic review. This work was supported by a grant from the European Union’s 7th Framework Programme provided for the project LOD2 (GA no. 257943).

REFERENCES


