m:Ciudad – Unleashing mobile user-provided services

Diego Urdiales
Rafael de las Heras
Telefónica Investigación y Desarrollo
{diegou, rafaelh}@tid.es

Marcin Davies
Anna V. Zhdanova
Telecommunications Research Center Vienna (ftw.)
{davies, zhdanova}@ftw.at

Milla Immonen
Juhani Heinilä
VTT Technical Research Centre of Finland
{Milla.Immonen, juhani.heinila}@vtt.fi

Maribel Narganes
Robotiker - Tecnalia
maribel@robotiker.es

Benoît Christophe
Linas Maknavičius
Alcatel-Lucent Bell Labs France
{Benoit.Christophe, Linas.Maknavicius}@alcatel-lucent.fr

Payam Barnaghi
University of Surrey
P.Barnaghi@surrey.ac.uk

ABSTRACT
This paper presents the concept of mobile user-provided services - small services that users can create and provide on the go from their mobile terminals. It addresses the technological challenges to apply this concept and describes an architectural approach to realize it. Ultimate personalization and access to the most relevant information are the main drivers for user-provided services, which have been materialized in the m:Ciudad project. By extending the widget model with advanced service aspects and semantics, m:Ciudad defines a novel architecture in the mobile terminal as well as in the framework.

Categories and Subject Descriptors
H.3.5 [**Online Information Services**]: Data sharing, web-based services.

General Terms
Algorithms, Design, Theory.

Keywords
User-provided services, user-created services, user generated content (UGC).

1. INTRODUCTION
Recent years have seen the wide spreading of the trend of accessing the web in diverse mobility situations, regardless of the user's physical location, network technology or device constraints. The reason behind the mobile web breakthrough is a combination of several factors: changes in user behaviour, availability of high-capacity networks and high-end handsets, proliferation of mobile content and web services, reduced cost of data access… In the move towards an open, collaborative mobile web, a need for unrestricted, customised information sharing and unleashing collective creativity emerged. Nowadays a user does not go to the web content – the content comes to the user. Mobile widgets, acting as a front end to web 2.0 services, play an important role in this revolution, since they present up-to-date information to the user without the need to browse the web. A mobile widget is usually a small front-end application with sharp focus and a thin, user-friendly interface.

As users become more eager to personalise their mobile widget workspace, and as they demand more accurate, relevant information than traditional web services can provide, the user-provided service concept emerges as a further step beyond mobile widgets. **Mobile user-provided services** (often also called *microservices*) are small, content-oriented mobile services which users not only consume but also *provide* from their mobile terminals. User-provided services extend the widget concept by providing not only a presentation layer, but also the mechanisms for the services to be fully executed within the terminal.

This paper describes the concept of mobile user-provided services and presents the proposal for a feasible user-provided service architecture which has been developed within FP7-funded STREP project m:Ciudad[1].

2. THE DRIVERS FOR MOBILE USER-PROVIDED SERVICES
Technological advances and user demands are converging as drivers for user provided services:

- User-created services are gathering momentum. Users are demanding more control of their services, and they are becoming more creative. *User-created services* provide an evolution beyond user-generated content. Several initiatives are trying to study new paradigms to allow users to create mobile and web-based services (Yahoo Pipes[2], Microsoft Popfly[3], OPUCE[4], SMS[5]…). The

WWW 2009, April 20-24, 2009, Madrid, Spain.

[1] Project website: http://www.mciudad-fp7.org

[2] http://pipes.yahoo.com

[3] http://www.popfly.com

[4] OPUCE FP6 European project: http://www.opuce.tid.es

[5] SMS FP6 European project: http://www.ist-sms.org

concept of user-provided services is a further step in the same direction, allowing users full control since they are not only creators but also providers of services from their mobile terminals.

- The user-generated content (UGC) market is becoming mature. However, an important portion of all UGC is not currently part of it. Short-lived, mobile sensor-based information, which can be extremely relevant even if only in small time and space intervals, has previously remained locked inside individual mobile terminals. By giving users the possibility to access and share this information, a new range of contents is put into the global UGC market.
- Crowdsourcing has proven to be a powerful alternative to centralised knowledge building, providing a reach and depth that traditional methods can hardly achieve. With cooperative user-provided services, multiple users feed contents to the same service, thus extending the crowdsourcing trend to the field of mobile services.

User provided services satisfy basic user needs: the ability to share information and contents instantly with fellow users and, conversely, the need to access relevant information irrespective of time and location. In addition, they provide a number of key benefits for users:

- Personalisation reaches another level, since users are not only allowed to *skin* their services and move them around in a workspace, but also to decide exactly how each service looks, and how it is presented to consumers
- The user-provided service architecture provides direct access to mobile sensor data and embedded capabilities, which can be easily combined and plugged to service outputs.
- As for access to relevant information, the fact that each mobile terminal has a server embedded and is a potential service provider creates an ecosystem of millions of sources for relevant information, scattered throughout all coverage areas. This reach is nearly impossible to match by any centralised effort.

User-provided services also yield important business benefits that will drive their adoption by relevant actors in the mobile services sector. Among others:

- Firstly, they are a high quality source of user knowledge. Since users are pushed to share more of their data with others, user profiles become richer, and it is easier to identify user trends, which users are active creators as opposed to passive consumers, etc. This knowledge can be used, for instance, to deliver highly relevant advertising.
- Secondly, the possibility to leverage the new set of mobile contents that enter the UGC market is also attractive to actors in the mobile services value chain.

The user-provided service platform is an effective means of dissemination of information in a way adapted to user preferences. This is why the platform can also be deployed for large corporations who wish to use it for distributed teams or individual employees.

3. THE M:CIUDAD DESIGN METHODOLOGY

The concept of user-provided services has been the focus of the m:Ciudad project. During its first year, a reference architecture for user-provided services has been defined following a structured methodology, described in this section.

3.1 Use case and scenario work

In order to fulfil the m:Ciudad project vision, it was necessary to achieve a common understanding within the consortium about what user-provided services were, how they could be provided from mobile terminals and how users could manage the service lifecycle from their mobile terminal. In order to accomplish this, innovative examples of user-provided services were envisioned, in the form of narrative short stories, i.e., usage scenarios [1]. The main scenarios that were identified are listed here:

- *Traffic Jam Killer*: a microservice which allows users to publish their location and speed in order to collaboratively work out the traffic status of a road segment.
- *Translator*: a user who is expert in a foreign language provides a translation service from her mobile, responding to translation requests she receives on the go.
- *My Mobile Blog*: your personalised blog, provided from your mobile, and updated instantly on the go with your latest contents and information
- *Friend Locator*: share my location and presence details with my friends directly from my mobile terminal.

3.2 Requirement analysis

The use cases obtained in the scenario analysis established a sound basis to advance the requirements gathering and system specification steps of the project [2]. By grouping the use cases according to their functional properties (e.g., creating, searching, publishing, executing services), their number was reduced drastically. A detailed analysis of the combined use cases led to the final inventory of **user requirements** based on usage scenarios.

Additional sources of requirements were detected from the project definition (Description of Work), the business model analysis and individual partner expertise. The final list of m:Ciudad **system requirements** was worked out by grouping and combining the initial requirements with the system-related requirements based on the scenarios and use cases. Furthermore, the system requirement specification produced an analysis on how the system requirements would affect the architecture of the m:Ciudad system.

Some key requirements emerged as crucial enablers for user-provided services. Among them:

- There needs to be easy, direct access to terminal capabilities and other information sources, so that users will be able to use them in their services.
- New paradigms for service creation need to be envisioned, so as to allow impulsive service creation from mobile terminals.
- The impulsiveness that drives user-provided services requires novel strategies for context-based search so that only the most relevant mobile services are shown.

3.3 Service specification

The methodology for the specification of user-provided services basically follows the well-known Model-View-Controller (MVC) pattern [7], which is also frequently employed for widget design. Within m:Ciudad the three MVC components are used as follows:

- Model: refers to the content a service provides
- View: offers the actual representation (comparable to a style sheet of a widget)
- Controller: reacts to user inputs or events (similarly to JavaScript routines within a widget)

It should be noted that each of these parts is enriched with fine-grained semantic annotations in order to facilitate efficient service creation, re-usability and meaningful, accurate search results.

Another core aspect of the service specification is the definition of a comprehensive service lifecycle ranging from service creation over deployment and execution and, finally, to service deletion.

3.4 Architecture methodology

For the m:Ciudad Reference Architecture design, a conceptual perspective was adopted. Architectural descriptions define enablers, main functional building blocks (components) and relationships (interfaces/interactions) between components for all service lifecycle phases: launching, searching, creation, publishing, execution. Functional and conceptual views are key to the architectural methodology since they describe the supporting features, functions, properties and sequencing (behaviour) which fulfil user requirements. Section 4 describes the main building blocks of the proposed architecture.

4. AN ARCHITECTURE FOR USER-PROVIDED SERVICES

The m:Ciudad architecture approach on which this user-provided services platform is being built is based on a modular design where the different modules that conform the overall architecture can be split into two main groups: terminal side components and framework side components, as shown in Figure 1. The full m:Ciudad architectural specifications can be found in [3].

As can be seen, the platform architecture is mainly focused on the user mobile terminal where services are hosted and executed, both for provision and consumption, whereas framework side modules play the roles of a central repository and registry. The main components are explained in the following sections.

4.1 Terminal-side components

The **Service Creation Kit (SCK)** is a mobile application running in the user mobile terminal, which performs as a service creation tool for non-technically skilled users, i.e. end users. The SCK provides an interface for the user to create services regardless of time or location. Two possibilities have been considered for the service creation process, namely a) using templates (incomplete services with some logic and interface/layout implementation, but with specific parameters/data stripped off to allow further customization) or b) using Simple Service Elements (SSE) as building blocks for creating a brand new user-provided service from scratch. SSEs must be small and easy to use, manipulate and describe. SSEs can be either content elements or functional elements. The SCK offers a basic selection of SSEs, but more can be downloaded from the Service Warehouse (see next section).

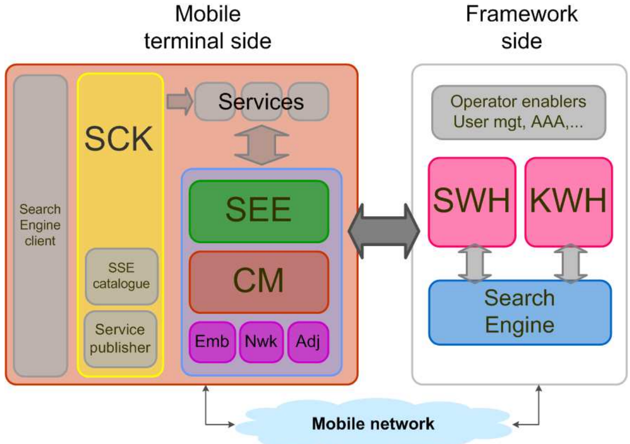

Figure 1. Main building blocks of the m:Ciudad architecture

This approach allows also tech-savvy users to build their own SSEs and to provide them to others [3].

The **Service Execution Environment (SEE)** is the mobile application running in the mobile terminal which controls the execution of all user-generated services deployed by the user on his terminal. m:Ciudad user-provided services are coded using a custom service description language which the SEE is able to interpret. Together with the Capabilities Manager (see next section), the SEE supports the simultaneous execution of all service instances being provided and consumed by the same user; the execution of each particular service is mainly driven by the service logic defined at the service creation stage, where the service behaviour is described. The SEE requires the ability to:

- capture incoming requests from consumers and push content to them;
- interact with the capabilities manager both for updating the status of the capabilities and for invoking the execution of the required and available capabilities;
- create, transform and manage user generated content; and
- infer knowledge and push it into the Knowledge Warehouse.

As opposed to SSEs, which are the building blocks for user-provided services from the user's perspective, **capabilities** are the main building blocks from the system's perspective, since they map the diverse sources of information that a user-provided service uses. Capabilities provide context information by using

- the sensors built-in in the mobile terminal, such as GPS, thermometer, camera, acceleration sensors, etc. – these are called **embedded capabilities**. In general, these capabilities do not have standardised interfaces in the terminals and the interfaces have to be mapped differently for each platform;
- devices located in the surroundings of the mobile device, such as an external GPS receiver; these are called **adjacent capabilities**; or
- functions provided by the operator network, such as video-calling, multi-conferencing, etc; these are called **network capabilities**.

Capability management within the scope of m:Ciudad consists in allowing both communication capabilities, embedded capabilities and adjacent capabilities to be used in order to realise the execution of an m:Ciudad service.

From the m:Ciudad mobile platform point of view, each capability is seen as an interface and can then be called and accessed in a common and uniform way. Such an approach attempts to focus on the functional aspect a capability delivers in order to better decouple layers. Indeed, it allows a service to use a capability wherever it is (embedded or external to the terminal). Capabilities are divided into two categories, namely functional (like taking a picture or making a phone call) and non-functional (like opening a Bluetooth connection). Such a choice has been made in order to address users with a broad variety of technical skill levels. Capabilities are wrapped into Small Service Elements (SSE) in order to be composed by the Service Creation Kit (SCK).

Acting as the central piece that resides within the mobile terminal, the **Capabilities Manager** (CM) module (see Figure 2) uses the capability interface definitions in order to allow the realization of user-provided services by creating an ecosystem where capabilities can be called, executed or aggregated. Based on a plug & play architecture, the CM module allows for a strong capability management by controlling capabilities through a dynamic registering process. Another key concept of the CM module is its ability to be aware of both context situation (like 3G network availability) and context of a service (like reception of a new SMS) and to share this information between capabilities through a strong event mechanism system. The CM module is also able to query the knowledge module to retrieve a capability pointer according to metadata. Finally the Capabilities Manager takes into account interdependencies between several capabilities (like external GPS access requiring Bluetooth turned on) thanks to a dependency graph which is dynamically rebuilt each time a capability is added in the mobile terminal.

4.2 Framework-side components

The **Knowledge Warehouse** (KWH) is a repository to store common specifications of user generated contents and application domain concepts. Ontology-based annotations and structured information representations are used in the KWH to provide interoperable and machine-processable metadata descriptions for contents. The Knowledge Warehouse works closely in collaboration with the Search Engine. The original contents are stored on user platforms and users only publish the content descriptions (i.e. annotations) in the KWH. The annotation ontology defines the main attributes of contents such as media dependent, temporal and spatial characteristics of the contents as well as attributes to link the contents to external domain ontologies. The domain ontologies specify common vocabularies and define relationships between concepts in the domains of application (e.g. tourism, traffic, and entertainment). The domain concepts are defined through external ontologies as well as (shallow) ontologies emerging from user provided keywords for content descriptions (i.e. community-driven ontologies).

The **Service Warehouse** (SWH) is the framework-side component which stores the full service code in the form of a service bundle. These bundles are downloaded by users who want to become providers of services which they have not originally created. In addition, the Service Warehouse keeps a matching table of users and services which they have downloaded, used to complement search results. Finally, the Service Warehouse also

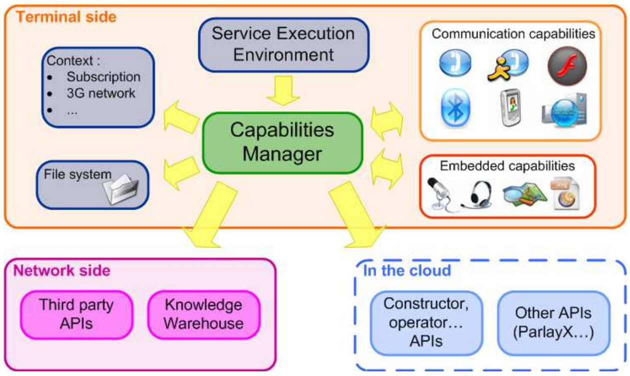

Figure 2. The Capabilities Manager and its relationship with other terminal-side components

acts as a repository for Simple Service Elements, which a user terminal can check periodically to obtain the latest available components.

The **Search Engine** component enables users to search for services; it also has mechanisms to search for ontologies, used by the rest of the modules in the architecture. The ontology search and selection mechanism is used to find available and relevant vocabularies to describe a service or content during the publishing process. The search engine also provides search mechanisms on metadata stored in the SWH and KWH. Metadata search is performed on content annotations and service descriptions to facilitate content/service selection based on a submitted query and emerging semantics from the provided annotations.

5. RELATED WORK

The related project OPUCE allows users to create services (mash-ups) by providing the necessary tools to allow everyone to build his/her own communication services without the need for a specific background in computing. However, the focus is on graphical mobile and Web editors for service creation, based on drag-and-drop operations, while m:Ciudad has a stronger focus on knowledge management and a general, semantically-enabled architecture for user-generated services. In addition, services created in OPUCE are provided from the network, and not from user terminals.

5.1 Description and Modelling of User-Generated Services

Semantic Web Services (SWS) are commonly based on rather complex service models such as OWL-S or WSMO. Although more lightweight approaches such as SAWSDL and SA-REST [5] exist as well, they are not suited for services with a potentially short life-span created by end-users.

Additionally, there are different modelling approaches for various aspects of mobile services; e.g. utilizing semantic modelling for Location Based Services [4] or for mobile service platforms in general [8]. Such approaches may not be widely accepted or standardized yet, however the main ideas introduced in these related work can be used to enhance and extend the foreseen user-provided service model.

5.2 End-User Aspects and Widgets

Although user-oriented prototype toolkits for the description of services exist [6], they are not feasible for mobile end-users while on the go due to their complex nature. Real-life present examples of commercial as well as user-created widgets include Nokia's WidSets [6] and Apple's Dashboard [7]. However, most of these widgets are still a mere data fetching or aggregation facility, and need to be found and installed manually. In comparison to these developments, the m:Ciudad platform for user-provided services explores "widgetising" of services with arbitrary functionality, fully resident in the mobile terminal, as well as easy semantically-enabled search and discovery.

6. CONCLUSIONS

Nowadays, mobile users demand more and more control over their services and applications, and mobile devices have become omnipresent in users' everyday life. Thus, a new mobile user role arises from both these trends: the user as provider of his/her own mobile services. With the aim of exploiting this potential, a new platform consisting of a new service architecture and a set of mobile tools to allow mobile users to create small, sharply focused and knowledge-based mobile-provided services has been elaborated and is presented in this paper.

Firstly, an overview has been given of the methodology followed in the m:Ciudad project for the design of the intended platform. Even if users were not directly involved, scenario-based work has proven to be a valuable tool to materialize the concept of user-provided services. Next, the architectural approach has been depicted together with an explanation of the main components and their functionalities. The architecture implies a novel functionality split between the mobile terminal, which hosts the service creation and execution environment, and the framework side, which provides centralised repositories for services and ontologies (but not contents). Finally, the user-provided service initiative led by the m:Ciudad project has been put in context by analysing the main related work and technologies.

The m:Ciudad project provides a foundation for the concept of user-provided services. It has defined an architecture and a service description language, and it will provide a reference implementation for the whole platform, both in the terminal and the framework. Future work will need to validate the concept with real mobile users and tackle the main barriers needed to move user-provided services beyond the prototype phase – limits in mobile inbound bandwidth and battery life, and inter-network interoperability, to name a few. The involvement of key players in the mobile services market, as well as the alignment with the strategic agendas of major technological platforms and institutions will surely help in this way forward.

7. REFERENCES

[1] The m:Ciudad consortium. "D1.1: m:Ciudad Scenarios and Use Case Analysis" (public deliverable).

[2] The m:Ciudad consortium, "D1.3: m:Ciudad User and System Requirements" (public deliverable).

[3] The m:Ciudad consortium, "D1.4: m:Ciudad Architecture Design" (public deliverable).

[4] Kim, J.-W., Kim, J.-Y., Kim, C.-S., 2006. "Semantic LBS: Ontological Approach for Enhancing Interoperability in Location Based Services". In Proc. On the Move to Meaningful Internet Systems 2006: OTM 2006 Workshops, LNCS 4277, Springer, pp. 792-801.

[5] Lathem, J., Gomadam, K., Sheth, A.P.: SA-REST and (S)mashups : Adding Semantics to RESTful Services. International Conference on Semantic Computing (ICSC) (Sep 2007) 469-476.

[6] Kerrigan, M., Mocan, A.: The Web Service Modeling Toolkit. In: The Semantic Web: Research and Applications. Volume 5021 of LNCS, Springer (2008) 812-816.

[7] Trygve Reenskaug, 1979. Models-Views-Controllers.

[8] Villalonga, C., Strohbach, M., Snoeck, N., Sutterer, M., Belaunde, M., Kovacs, E., Zhdanova, A.V., Goix, L.W., Droegehorn, O. "Mobile Ontology: Towards a Standardized Semantic Model for the Mobile Domain". In Proc. of the 1st International Workshop on Telecom Service Oriented Architectures (TSOA 2007) at the 5th International Conference on Service-Oriented Computing, 17 September 2007, Vienna, Austria, Springer (2007)

8. APPENDIX: COMPLETE AFFILIATION DETAILS

Telefónica Investigación y Desarrollo

Emilio Vargas 6

28043 Madrid, Spain

Tel. +34913379000

Telecommunications Research Center Vienna (ftw.)

Donau-City-Strasse 1/3

1220 Vienna, Austria

Tel. +4315052830

Robotiker – Tecnalia

Parque Tecnológico, Edificio 202.

E-48170 Zamudio (Bizkaia) (SPAIN)

Tel. +34 94 600 22 66

Alcatel-Lucent Bell Labs France

Centre Villarceaux, Route de Villejust

91620 Nozay, France

Tel. +33 130 772 777

VTT Technical Research Centre of Finland

Kaitoväylä 1,

Po.Box 1100, FI-90571 Oulu, Finland

Tel. +358 20 722 2168

University of Surrey

CCSR, Guildford, Surrey, GU2 7XH, UK.

Tel. +44 1483 68 9485

[6] http://www.widsets.com

[7] http://www.apple.com/downloads/dashboard/