Context-Aware Collaborative Platform in Rural Living Labs

Olfa Mabrouki[1,2], Abdelghani Chibani[2], Yacine Amirat[2],
Monica Valenzuela Fernandez[3], and Mariano Navarro de la Cruz[3]

[1] CityPassenger SA, Avenue de l'Atlantique
Les Conquérents BP 903, 91976 Courtaboeuf Cedex, France
omabrouki@citypassenger.com
[2] Université Paris XII Val-de-Marne, LiSSi, E.A. 3956
120-122 rue Paul Armangot, 94400 Vitry sur seine, France
{olfa.mabrouki,chibani,amirat}@univ-paris12.fr
[3] Grupo Tragsa, Gerencia TIC ||ICT Division
Subdireccin de I+D+i ||Subdirectorate of Innovation and R&D
Julin Camarillo, 6 B; 4 planta Madrid Spain
{mvaf,mnc}@tragsa.es

Abstract. In this paper, we present a collaborative context-aware framework for rural living labs or rural innovation ecosystems as Social Spaces for Research and Innovation (SSRI). The proposed framework exploits seamless integration of standard ubiquitous computing technologies to support smart collaboration and knowledge sharing between rural communities. We underline an open collaborative platform based on context-aware components useful in any rural living lab area. This platform is focused on the bus component which acts as a connector for the different living labs. The bus architecture gives the advantage to facilitate the dynamic integration of living lab services using discovery and binding methods. It guarantees also the large scale interconnection of all the living labs. Building such framework requires resolving a main issue of the design approach. Moreover, we experiment the use of such platform in rural community of fishery sector. The work presented in this paper is one of the main results of the Collaboration@Rural (C@R) European research project: a collaboration platform for working and living in rural areas (FP6-2005-IST-5-03492) that aims to build a platform of networked living labs for context-aware collaborative working in rural areas.

Keywords: rural living labs, collaborative platform, context awareness, ubiquitous computing, orchestration.

1 Introduction

The term Living Lab (LL) was given at the first time in 2003 by ProfWilliam Mitchell from MIT, Media Lab and School of Architecture and city planning. He defines this new concept as a research methodology for sensing, prototyping, validating and refining complex solutions in multiple and evolving real life contexts.

This new concept is also represented as innovation environments where stakeholders form a partnership of enterprises, users, public agencies and research organizations. Since then, many definitions of LL have been proposed in the literature. In a LL, cooperation is established for creating, prototyping and using new products and services in real-life environments. Users are not seen as an object of innovation and customers but as early stage contributors and innovators [1]. Thus, we might view LLs as concrete implementations of user driven open innovation environments [2]. Consequently, we have agreed on the following definition: LL is a research methodology for innovation that challenges the whole research and innovation process in real-life conditions by human, social, cultural, organizational and institutional aspects, and has an impact on sustainable service,business and technology development.

Several researches in the field of LL were undertaken in the literature. The most of them concerns the proposition of ubiquitous technologies applications to impact socially and economically LLs environments. Other works deal with LLs in education [3] and home environment [4]. In the last five years, the European commission funded researches to build LLs as open innovation platforms. These LLs are intended to be used in a real-world environment for collaboration among stakeholders in the value of ICT (Information and Communication Technologies) production [5].

Hence, the paradigm of LL is used more and more in the area of ubiquitous computing as it involves user in the early phases of service innovation. Moreover, Rural Living Labs (RLLs) are a special kind of this paradigm that enhances ICT rural development and that are user-centered design. Ubiquitous collaboration in LLs targets to provide users with integrated context-aware services capable of exploiting all the facilities of both wired and wireless environments. These services allow creating flexible and effective virtual teams. Such integrated services placed in the rural context may be adapted to different users situations and environment characteristics.

The C@R LLs have been setting up to experiment advanced collaborative work and business innovations to enhance attractiveness of rural areas and strengthen rural development. Within the C@R project, a LLs vision and methodology have been developed and implemented to create rural and regional innovation environments. This project aims to propose and develop an innovative collaborative platform for rural communities and demonstrate the use of this platform by integrating various tools for various user communities. It promotes the development of an open collaborative architecture which enables the reuse and the contextualization of services and collaborative tools. Seven LLs have been launched; six in Europe and one in South-Africa, to establish and experiment collaborative platforms and applications enhancing Small and medium enterprises (SME) related work and business collaboration in specific sectors.

In this paper, we highlight our proposal for designing the collaborative context-aware platform for LLs and managing context-aware services. The methodology used is called Open Service Oriented Architecture (OSOA). The context-aware collaborative platform enhances social and economic conditions in rural environments through the introduction and experimenting of new ways of working and managing business. This platform is focused on the bus component which acts as a connector for the different LLs. The bus architecture gives the advantage to facilitate the dynamic integration of LL services using discovery and binding methods. The paper is organized in the following way: section II introduces the framework for context-aware collaborative platform in LLs. In the section III, we illustrate the use of this framework for rural LLs in the fishery sector.

2 Framework for Context-Aware Collaborative Platform in Living Labs

An efficient methodology to set up a framework for context-aware collaborative platform is indispensable. We present and argue the use of the OSOA approach for the design of such architecture. Then, we describe the framework with its several layers and highlight the different features of context-aware components in this framework.

2.1 Framework Modeling Methodology: OSOA Approach

Service Oriented Architecture (SOA) nowadays is a well known term providing principles of how to develop and integrate a system of loosely coupled services. SOA defines not only the low-level software architecture design principles but is a complete enterprise software concept including among others security, governance, deployment and integration. The term Open Service Oriented Architecture (OSOA) which is used to define the C@R SOA [6][7] approach also defines a set of principles to develop and integrate a system of loosely coupled services but also components. The differences to a traditional SOA approach are the concepts built upon and how they are combined to provide the ground of a service oriented collaborative working environment (CWE).

On the software architectural level there currently exist many different flavors and interpretations of service-oriented architecture (SOA) concepts, which are being promoted by different organizations. One of the most popular and active SOA developers group is the Open SOA (OSOA) Collaboration [8], which represents an informal group of industry leaders with a common goal. They work together on the definition of a language-neutral programming model able to meet the needs of enterprise developers who are developing software following the SOA principles.

The OSOA follows similar concepts and represents a system that exploits the SOA benefits using language-neutral concepts to build the ground for a service oriented CWE. Since the C@R architecture will be used in a broad community, among different LLs, and in a diverse set of use case scenarios it is essential to build the architecture upon well defined standards and to avoid proprietary concepts. The most important standards used in the C@R OSOA improve the quality of developed software and thus ensure the interoperability, maintainability and reusability of the individual components. This enables the utilization of advanced Collaborative Working Environments even in rural setups.

The C@R CWE System is certainly a different and wider concept than the OSOA Collaboration Group proposal and other SOA approaches, as it is not just considering software developers agreeing on a software architecture providing user services, but as a whole Open System to enable a CWE considering all available actors: users, equipment, service providers,software providers, CWE system designers and stakeholders.

Reference Architecture Design. The main purpose of the C@R project is to introduce and involve collaborative works environments which will raise rural development. Actually, the issue is related to the networked LLs where users need to collaborate, communicate and exchange information. Setting up an open collaborative architecture that enables the reuse, the contextualization and the personalized of services and tools is mandatory to cover the rural character of such LLs.

Architecture layers. A layered architecture design realizes decoupled components [9]depicted in the figure 1 to deal with different aggregation levels of business functionality, namely:

- **CCS. C**ollaborative **C**ore **S**ervices implemented as reusable software components that encapsulate distinctive core functionality. Such functionality provides basic services (e.g. 3 G connectivity, SMS delivery, order creation etc.). CCSs plug into the C@R Control BUS where they are registered. Every CCS provides a public API, implemented as a Web-service;
- **SCT. S**oftware **C**ollaboration **T**ools [9] comprise aggregated functionality, which can be integrated into a final RLL application, but is of such a degree of independence to be usable for various applications even across different Living Labs. Simple SCTs provide only one CCS and more sophisticated SCTs orchestrate several CCSs into a business process to the RLL application via a web service interface. SCTs can be defined using different languages. One of the basic objectives of C@R is to use as much as possible standard languages. Current implementations of the SCTs use BPEL [10] and/or BPMN [11] that allows the creation of scripts, which are executed by the orchestration engine;
- **OC. O**rchestration **C**apabilities [12] provide collaborative functions and libraries that will be used by executable scripts that define the composition of SCTs. Three orchestration capabilities are identified, namely Context Awareness, Distributed Workspace, and Advanced Services. A Collaborative situation may involve atomic functions from different OCs such as Messages Broadcasting, Shared Display, Videoconference systems, etc. categorized as the three identified orchestration capabilities. OCs can be implemented as Web Services (following the CCS design) or as static libraries deployed together with the Control BUS;
- **RLL. R**ural **L**iving **L**ab applications cover end user interactions (via a User Interface) with a system supporting collaborative workflows that overcome problems related to rural activities. These applications make use of underlying

layered business functionality encapsulated in SCTs but also linking directly to CCS and OC functionality.

Additional to these layers a Control BUS has been conceptualized and implemented in order to centrally deal with component registration and brokerage.

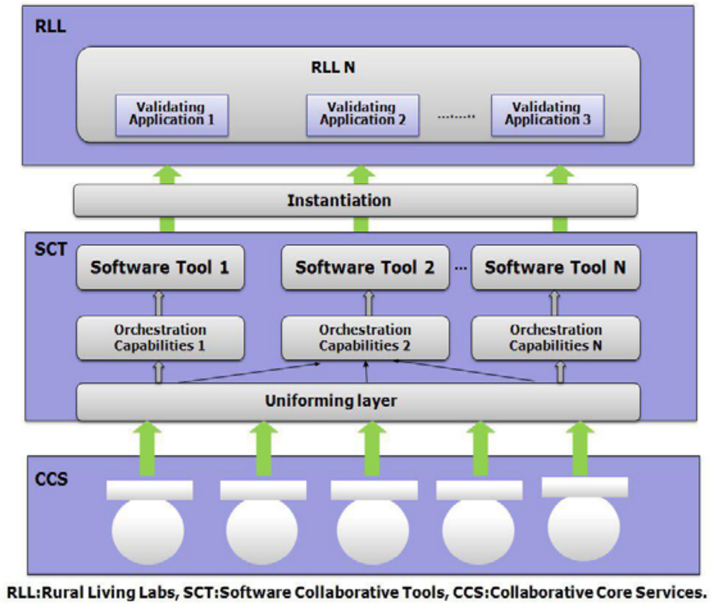

RLL
RLL N
Validating Application 1
Validating Application 2
Validating Application 3
Instantiation
SCT
Software Tool 1
Software Tool 2
Software Tool N
Orchestration Capabilities 1
Orchestration Capabilities 2
Orchestration Capabilities N
Uniforming layer
CCS
RLL:Rural Living Labs, SCT:Software Collaborative Tools, CCS:Collaborative Core Services.

Fig. 1. Framework for context-aware networked RLLs

Control Bus. Control functions of the elements of the C@R Architecture are encapsulated in the Control BUS. It acts as a resource broker, where signaling information about resources is exchanged, enabling the system to search for resources, managing their interconnection and supporting collaboration among different CWEs. The BUS acts as an informing middleware that is a conceptual inter-layer space designed for CCS component harmonization, homogenization and adaptation to standards. It makes the C@R architecture more powerful and flexible allowing an easy integration of proprietary or new standard CCS components. This key piece of C@R architecture consists of five modules (see figure 2):

– **Bus maintenance.** This module is responsible for keeping the logs of all the BUS activities, and for all tasks related to the management of the BUS itself. Furthermore, the module offers configuration files and interfaces for the administrators to control the behavior of the BUS;
– **Registrar.** This module is responsible for keeping a database of all components (CCS and Orchestration Capability) connected to the system. Furthermore it implements search functionality, allowing any element to look for other elements (CCS, Orchestration Capability) connected to the platform;

- **Connector.** This module allows interconnection of components and the management of release and monitoring of connections among them;
- **Instantiation management.** This module serves to support the instantiation process;
- **Bus Inter-working.** This module is responsible for negotiating the communication with other C@R platform BUSes, or even control layers of other platforms. It provides collaboration among different BUSes and enables resources sharing.

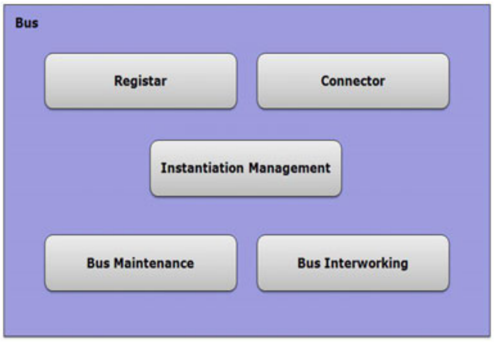


Fig. 2. Control bus

BUS implementations require the establishment of information channels with the resources that it pretends to manage and interconnect. Those Control Communications are centralized by the BUS and use Web Services as transport technology, while Data Communications are P2P and may use any kind of transport technology.

Orchestration capabilities. The Orchestration Capabilities (OC), as defined by C@R [12], provides collaborative functions and libraries that will be used by executable scripts that define the composition of SCTs. Three orchestration capabilities have been identified:

- Distributed Workspaces includes the minimum necessary methods and data structures to build high-level collaborative functions;
- Context Awareness adds functions providing environmental status and methods for context change reaction;
- Access to Advanced Services includes SIP/IMS & Security capabilities.

C@R has analyzed current collaboration activities performed at Rural Areas (using the seven Rural Living Labs included in C@R. This analysis concluded on the identification of 45 services that where common to two or more RLL and 60 services that are specific for only one RLL. The identified common services are the candidates for being including in any of the OCs of C@R in order to provide the basic support for creating complex collaboration services.

Context-aware components. Context awareness was introduced for the first time by Schilit and Theimer [13]. They defined context of an entity as a set of information concerning the identity of the entity, its location, identities of nearby objects and changes to those objects. Ryan et al. [14] present context of an entity as its environmental information, such as location, time, temperature and its identity. Dey [15] considers context of an entity as its physical, social, emotional, and mental (focus-of-attention) environments, location and orientation, date and time of day, other objects in the environment.

The majority of these researchers share a common vision of context as it represents a set on information about location, time and activity of a person. In the C@R architecture, CCS components are contextaware components as they provide contextual information about the user. Actually, these components provide information about user location, user profile, spoken languages and Web sensors, namely the following components have been implemented and used in the individual Living Labs (see figure 3):

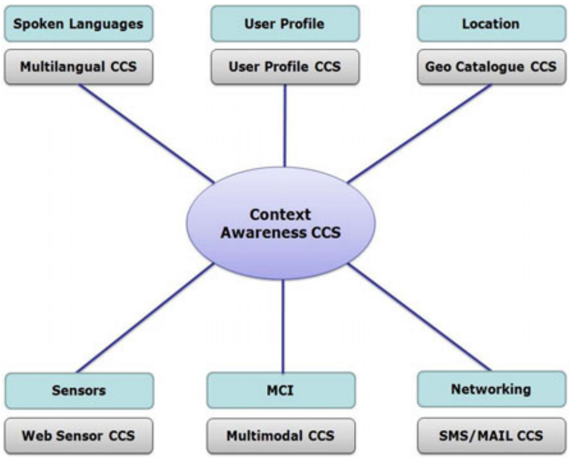


Fig. 3. Context-aware components

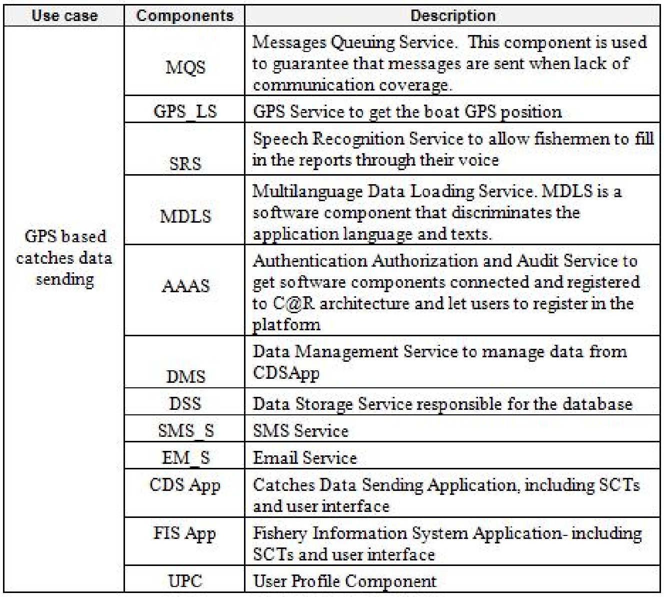
- **User profile:** the User Profile Component (UPC) describes the user personal information, preferences and role. Once the user is authenticated, it is possible to load his particular profile and particular preferences. In this way, available components and their settings are customized on startup. The context awareness API related to UPC allows full access with appropriate authorization to context information component to allow technicians the reuse of the UPC by simply invoking some methods. This component is for its nature distributed and cross-LLs as potentially. It is expected to run on the top of different databases which are implemented into several and heterogeneous products both open source and commercial licenses;

- **Geo-Catalogue:** catalogue services are the key technology for locating, managing and maintaining distributed resources. With catalogue services, client applications are capable of searching for resources in a standardized way (i.e. through standardized interfaces and operations). Catalogue component developed for C@R architecture automatically collects CCS component metadata and provides their registration into metadata catalogue. This internal component gives to users/developers possibility to register metadata records for new developed CCS as well as to know which CCS are currently available in the C@R architecture. Once registered metadata can be searched, extended or updated using some of metadata applications (Metadata editor or Metadata Catalogue System);
- **Web sensor:** this component presents many opportunities for adding a real-time sensor dimension to the Internet and the Web. This has extraordinary significance for science, environmental monitoring,transportation management, public safety, facility security, disaster management, utilities Supervisory Control and Data acquisition (SCADA) operations, industrial controls, facilities management and many other domains of activity;
- **MDLC:** the **M**ultilanguage **D**ata **L**oading **C**omponent is a context-aware component which allows user applications to retrieve a set of configuration files that contain the localized texts needed to interact with the end users. This software component enables user applications to adapt their interfaces to a particular user. Hence,the communication happens in terms of their preferred language. The component aims to empower user applications with the capability to perform the loading of different languages as a set of localized texts at the interface level stored in configuration files which are particular to a specific application.

3 Experimenting C@R Platform in the Fishery Sector

Among the different rural regions of LLs, we have chosen as an experiment example the Cudillero living lab (in Asturias, Spain). This LL has been developed in collaboration with public administrations, the local authorities and the fishery guilds. Applications developed for the fishermen try to enhance current business processes in order to make fishing production more profitable, e.g. helping on the day-by-day activity of the users in the vessels and in the auction process via the transmission of reports on the catches (arrival hour, sizes of the catches, total weight of the catches, etc.) and thereby contributing with a significant time and workload saving. The applications also contribute to improve the safety of fishermen in case of accidents or health emergencies, providing an immediate response from the health authorities. Furthermore, the collaboration between vessel and port will serve to optimize the organization of the port activities.

The following use cases have been implemented in the Cudillero RLL: GPS based catches data sending; Weather reports; Alerts management service and safety on board; Messages delivery service by instant messaging and presence. Based on mockups and prototypes validated by the user community, the software platform is implemented according to the principles of the proposed reference architecture. Once prototypes are developed, the basic software components and their interactions are determined. As a result the three layered architecture is mapped onto the individual components (see Figure 4, for use case Catches data sending). From bottom to top, CCS (Collaborative Core Services) components are the atomic resources which are orchestrated thanks to a control middleware, by service scripts and collaborative functions in the SCT layer of the architecture. These collaborative core services (CCSs) in layer 1 are registered in the resources broker (Bus) enabling the system to search for resources and managing their interconnection. A homogeneous layer (BusCCSOperations library) registers and connects CCSs to the Bus, in order to make each identified CCS available to the C@R platform.



Use case	Components	Description
GPS based catches data sending	MQS	Messages Queuing Service. This component is used to guarantee that messages are sent when lack of communication coverage.
	GPS_LS	GPS Service to get the boat GPS position
	SRS	Speech Recognition Service to allow fishermen to fill in the reports through their voice
	MDLS	Multilanguage Data Loading Service. MDLS is a software component that discriminates the application language and texts.
	AAAS	Authentication Authorization and Audit Service to get software components connected and registered to C@R architecture and let users to register in the platform
	DMS	Data Management Service to manage data from CDSApp
	DSS	Data Storage Service responsible for the database
	SMS_S	SMS Service
	EM_S	Email Service
	CDS App	Catches Data Sending Application, including SCTs and user interface
	FIS App	Fishery Information System Application- including SCTs and user interface
	UPC	User Profile Component

Fig. 4. Prototypes in basic components, in this case catches data sending

Services for Cudillero operate in a main domain where two sub domains are distinguished: the fishing boats sub domain and the fishery sub domain (see figure5). Each sub domain relies on one C@R bus. Different sub domains are registered in the Cudillero domain through the bus internetworking capability. This module also enables the reutilization of basic resources or the information exchange with other domains as other ports (i.e. Aviles port). Layer 2 in Cudillero utilizes a specific component as an Orchestration Capability: the authentication Authorization and Audit Service (AAAS). AAAS acts as a transversal service that needs to be preregistered in the bus to let the rest of CCSs to be authenticated to the C@R platform.

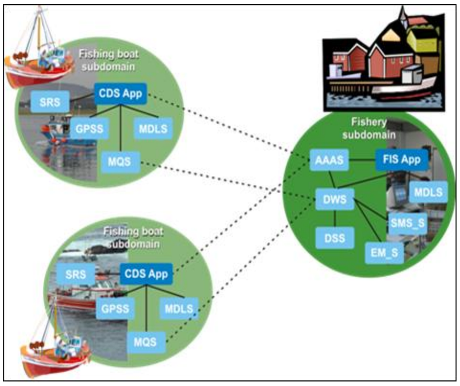
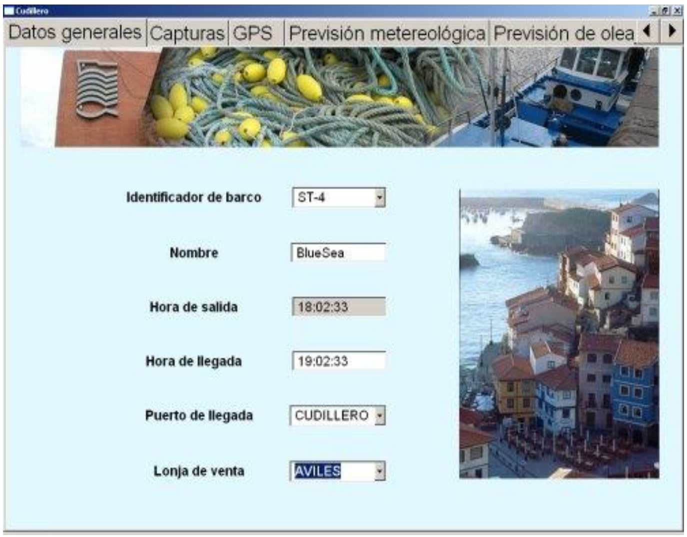


Fig. 5. Software components in Cudillero sub domains



Identificador de barco	ST-4
Nombre	BlueSea
Hora de salida	18:02:33
Hora de llegada	19:02:33
Puerto de llegada	CUDILLERO
Lonja de venta	AVILES

Fig. 6. CDS APP main data screen

Layer 3 defines the Collaborative services instantiation process. SCTs or software collaboration tools are the key elements to instantiate the collaborative platform relying in each bus. The SCTs deal with the modeling of the business processes of each sub domain. This piece of software is first compiled to get a BPEL script. These scripts contain information about all the necessary elements and basic services to be connected and started to run the platform. This BPEL script is uploaded to the SCT scripts repository. When the instantiation process begins, the SCT is downloaded to a server (composition engine) configured with some instantiation parameters (additional code). As a result, CCSs defined in the SCT are deployed to the server, registered to the bus and started. Most of the software components are identified as Collaborative Core Services (CCSs) except the AAAS that acts as a transversal service that needs to be preregistered in the bus to let the rest of CCSs to be authenticated to the C@R platform.

Two special CCSs were distinguished in each sub domain: 1) CDS App - Catches Data Sending Application, see Figure 5, and 6) FIS App - Fishery Information System Application. These specific CCSs (CCS applications) consist of the graphic user interfaces and the service framework. These are main threads that use and orchestrate the basic services (rest of CCSs) to compose end user applications.

4 Conclusion

One of the key objectives of C@R is the development of a reference architecture reflecting advantageous concepts that overcome a variety of challenges and pain points typical for rural CWEs. Deriving common characteristics of such architecture turned out to be difficult due to the limited capabilities of end users to reflect on technical needs and to the differences in target sectors of the 7 Living Labs involved.

C@R found out overlaps between architectural needs if not between all Living Labs at least between some of them. These overlaps have been translated into several principles (decoupling, open standard compliancy, flexible infrastructure support, service orchestration, interoperability, etc.) that drove the architectural design and the implementations in the individual Living Labs.

The common principles of the reference architecture have been realized exemplary and subsequently validated in terms of added value. Such common principles include the usage of most important standards (e.g. web services, BEPL), component representation (e.g. BPMN), tools (e.g. Intalio Designer), reusable, encapsulated functionality (OC services, CCSs), security models (e.g. AAS) or service brokerage (e.g. BUS). Besides commonalities the flavored implementations in the different Living Labs also showed distinctive differences that reflect the local specifics, e.g. the usage of the sub domain concept in Cudillero (fishing boats) or the limited usage of the BUS in Sekhukhune due to network impediments. The full potential of architectural benefits couldnt be leveraged during the lifetime of the project. Nevertheless the validation of architecture implementations in distinctive experiments provided promising results: In particular the C@R reference architecture is capable to facilitate the reuse of collaboration services, concepts and components across design and runtime environments of different CWEs.

The required degree of flexibility to develop and operate software collaboration tools has been assured through the usage of the most relevant standards in the fields associated to services. Openness and interoperation of CCS components for SCT orchestration coming from different platforms have been showcased. The performance of SCTs using the base components of the architecture is satisfactory and cost and effort required to develop software collaboration tools are competitive.

References

1. Hippel, E.V.: Democratizing Innovation. MIT Press Books, vol. 1. The MIT Press, Cambridge (2006)
2. Guzman, J.G., Schaffers, H., Bilicki, V., Merz, C., Valenzuela, M.: Living labs fostering open innovation and rural development: Methodology and results, Asia-Pacific Tech. Monitor (September 2007)
3. Abowd, G.D.: Classroom 2000: An experiment with the instrumentation of a living educational environment. IBM Systems Journal 38, 508–530 (2000)
4. Kidd, C.D., Orr, R., Abowd, G.D., Atkeson, C.G., Essa, I.A., Macintyre, B., Mynatt, E., Starner, T.E., Newstetter, W.: The aware home: A living laboratory for ubiquitous computing research, pp. 191–198 (1999)
5. Oliveira, A., Fradinho, E., Caires, R.: From a successful regional information society strategy to an advanced living lab in mobile technologies and services. In: Hawaii International Conference on System Sciences, vol. 4, p. 83a (2006)
6. D2.1.3, C.P.D.: C@r and osoa design (July 2008)
7. D2.5.1, C.P.D.: C@r and osoa design (July 2008)
8. Edwards, M.: Open service oriented architecture (June 2006), `http://www.osoa.org`
9. D2.1.5, C.P.D.: Workflows for rural activities (October 2009)
10. OASIS: Bpel web services business process execution language version 2.0., `http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpelv2.0-OS.html`
11. Group, O.M.: Bpmn, `http://www.bpmn.org`
12. D2.5.2, C.P.D.: Integration of distributed workspaces, localization and context awareness and access to advanced service components into sct for rural environments (October 2009)
13. Schilit, B., Theimer, M.: Disseminating active map information to mobile hosts. IEEE Network 8, 22–32 (1994)
14. Ryan, N.S., Pascoe, J., Morse, D.R.: Enhanced reality fieldwork: the context-aware archaeological assistant. In: Gaffney, V., van Leusen, M., Exxon, S. (eds.) Computer Applications in Archaeology 1997, British Archaeological Reports, Oxford, Tempus Reparatum (October 1998)
15. Dey, A.K.: Context-aware computing: The cyberdesk project. In: AAAI 1998 Spring Symposium on Intelligent Environments, Palo Alto, pp. 51–54. AAAI Press, Menlo Park (1998)