HAL
open science

A Generic Service Oriented Software Platform to Design Ambient Intelligent Systems

Stéphane Lavirotte, Gaëtan Rey, Gérald Rocher, Jean-Yves Tigli

To cite this version:

Stéphane Lavirotte, Gaëtan Rey, Gérald Rocher, Jean-Yves Tigli. A Generic Service Oriented Software Platform to Design Ambient Intelligent Systems. 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp 2015), Sep 2015, Osaka, Japan. pp.281-284, 10.1145/2800835.2800843 . hal-01297410

HAL Id: hal-01297410

https://hal.science/hal-01297410v1

Submitted on 5 Apr 2016

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution - NonCommercial - ShareAlike 4.0 International License

A Generic Service Oriented Software Platform to Design Ambient Intelligent Systems

Stéphane Lavirotte
Université Nice Sophia Antipolis
CNRS (UMR 7271), I3S
Sophia Antipolis, France
Stephane.Lavirotte@unice.fr

Gérald Rocher
Université Nice Sophia Antipolis
CNRS (UMR 7271), I3S
Sophia Antipolis, France
Gerald.Rocher@unice.fr

Gaëtan Rey
Université Nice Sophia Antipolis
CNRS (UMR 7271), I3S
Sophia Antipolis, France
Gaetan.Rey@unice.fr

Jean-Yves Tigli
Université Nice Sophia Antipolis
CNRS (UMR 7271), I3S
Sophia Antipolis, France
Jean-Yves.Tigli@unice.fr

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s). Copyright is held by the owner/author(s).
UbiComp/ISWC '15 Adjunct, September 7-11, 2015, Osaka, Japan.
ACM 978-1-4503-3575-1/15/09.
http://dx.doi.org/10.1145/2800835.2800843

Abstract
Smart devices or smart things are widely deployed within environments and have to work in concert to assist users in many domains. The interoperability between things is achieved by the help of Internet and Web of Things. Despite this progress, a main challenge remains to fully manage the heterogeneity of the smart things: handling their dynamicity at runtime. In this paper, we present a three layers platform to address this challenge. The first layer monitors the appearance and disappearance of smart things in the environment. The second one provides mechanisms to dynamically compose the services provided by smart things. The third layer offers an autonomic context-driven composition mechanism based on a new software paradigm: '*application schemas*'. This layer manage the interferences and conflicts that may occur during the autonomic composition process.

Author Keywords
Middleware; Internet of Things; Web of Things; Ubiquitous Computing; Context-aware computing

ACM Classification Keywords
I.2.2. Automatic Programming; D.2.6 Programming Environments

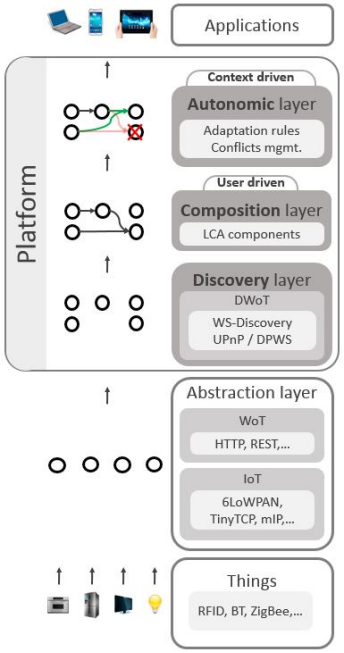

Figure 1: Three layers middleware to discover things, to compose them and to produce applications by an autonomic composition process

Introduction

Today, smart things are widely deployed within environments and aim at providing functionalities for users. A smart thing communicates with a user using a server/client model; an application running on a smartphone allows to get the data from one specific thing. Hence, the user need a specific application for each thing used.

But currently, there is a need for these things to work in concert to assist users in several domains (healthcare, smart houses, etc…). This cooperation requires a strong interoperability between smart things, firstly achieved by allowing them to communicate. Internet of Things (IoT) tries to provide a solution to the technological heterogeneity issue, but it is still challenging due to the large number of initiatives [1]. Among all the possible solutions, web services based approach (Web of Things, WoT) is now widely accepted [2]. Although we have an infrastructure composed of web services, a main challenge remains: the dynamicity of the environment corresponding to the appearance and disappearance of things has to be managed; "*the world is not static*". This implies to dynamically reconfigure, orchestrate or compose the available web services at runtime as we cannot totally anticipate all the possible configurations (like in software product lines). These dynamic reconfigurations can be managed by the user if and only if the changes are not too frequent, and reconfigurations of the system not too complex. Otherwise, the reconfigurations must be triggered on context changes, in an autonomic manner, reactively, and in a timely fashion.

In this paper, we present our three layers middleware (Figure 1) which provides the ability to discover the available services in the environment and allows them to be interconnected in order to create an application. This composition can be user driven or context driven. The context, in this case, is all the available services in the infrastructure. The platform also provides a middleware for runtime autonomic composition based on application schemas (rules) instantiated depending on the services availability. It also permits to handle interferences and conflicts that may occur when applying concurrently multiples rules.

An Infrastructure Composed of Dynamic Web Services

Dynamic Web of Things

Ubiquitous Computing aims at integrating physical objects in the digital world. Developments in the field of embedded devices allow now to have smart things populating our daily life. The interconnection of these smart things is facilitated with the emergence of lower-power transport protocols like Bluetooth, ZigBee, X10... But the heterogeneity and the variety of these protocols are an obstacle for their interconnection. In order to facilitate the cross integration of these things into composite applications, the use of web services approach enables interoperability and loose-coupling like in business information systems. The use of WS-* but also REST approach and HTTP protocol (WoT) allows the integration of heterogeneous things into applications. Several researches focused on enhancing these web services with dynamicity in mind, making them more suitable for real world. This led to UPnP and its evolution to DPWS. One of the main advantage of these protocols is their ability to take into account the dynamicity of the physical infrastructure by adding research and discovery mechanisms. These mechanisms give services the ability to be dynamically

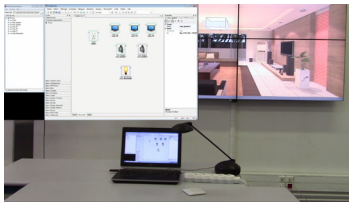

Figure 2: Virtual things can also be part of the dynamic web services infrastructure

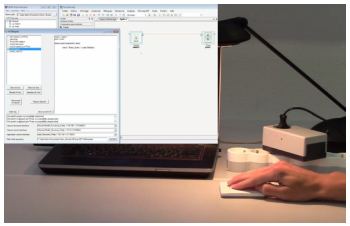

Figure 3: Dynamic composition layer to create application at runtime (user-driven composition)

Figure 4: Autonomic context driven composition (depending on things in physical environment)

managed (composition, orchestration …) into composite applications. As the world is not static, the dynamicity implied by the appearance and disappearance of smart things in the environment has to be managed by the first layer of the middleware.

Different Kind of Things

Moreover, the web service approach applied to smart things using protocols like WS-Discovery, can also be applied to digital things like: HMI fragments (as mashups or dedicated interfaces), business information services, or 3D virtual environments simulating worlds with virtual objects represented by 3D meshes (Figure 2) one can interact with through attached services (like a physical thing).

A Dynamic Service Composition Layer Managed by User

The second layer of the middleware is the dynamic software composition layer (Figure 3). Indeed, contrarily to business process languages for classical service composition like BPEL, Ambient Intelligent Systems are intrinsically interactive and need another software composition paradigm. Our programming model is inherited from Beans Assemblies. Main concepts of beans are input and output ports and properties. In our model, the discovery of new services associated to smart things is taken into account in this layer by generating a proxy component. This proxy component is a software component based on the Lightweight Component Architecture (LCA) model [3]. These components are called 'light' for several reasons. The first one is that they execute in the same memory address space, and in the same process, so their interactions are reduced to the simplest and the more efficient one, the function call. The second reason, which stems from the first, is that they do not embed nonfunctional code or other useless technical service in this local environment. Their memory footprint is then reduced and they are quickly instantiated and destroyed. To finish, they don't contain any reference to other components at design-time, and respect black box and late-binding concepts. The dynamicity of the model is thus maximal, components using events to communicate with the others, they are fully decoupled, and highly reactive. This runtime services composition model is well suited for interactive systems. Based on the available proxy components, a developer can manually create a composition by interconnecting events (output) from one component to methods (input) from another one. But this user driven composition is not suitable when dealing with dynamic composition.

An Autonomic Context Driven Composition

Dynamically reacting, at runtime, to the smart things appearance and disappearance cannot be handled automatically by the second layer. A new paradigm for autonomic software composition must be introduced based on an autonomic reasoning to deduce a software composition from context observations, at each time. The third layer allows to automatically and dynamically compose multiple applications sharing common services according to the context evolutions (Figure 4). At a first stage, the context is limited to all the available smart things and their associated services at a time.

Our new paradigm describes application as an '*Application Schema*' (AS) defined as a set of rules. The new hypothesis of these AS is that it cannot be known at design time what devices will be available at runtime. Thus, a first subset of rules is used to search

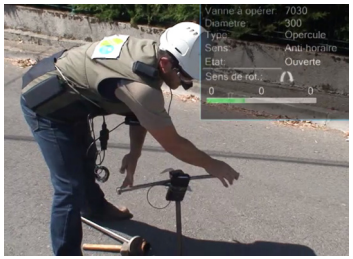

Figure 5: Evaluated in the field with industrial partners: Suez Environment, EDF R&D

and select required services to apply an AS. A second subset of rules defines how to create the application with available services. We introduced various algorithms [4], [5] to manage interferences and conflicts between multiple applications produced by multiple AS sharing services. This part can thus be considered as a middleware between AS paradigm to design an ambient system and the runtime composition of services. We also evaluated the response time of the automatic composition process according to numerous system parameters [6]. This allowed to assess the adequation between the ambient system complexity and its responsiveness to context changes.

Conclusion and Perspectives

In this paper we presented a three layers WoT software platform for ambient intelligent systems. This approach first takes into account the dynamicity of the underlying infrastructure populated with web services attached to physical smart things. A second layer provides runtime user-driven composition mechanisms while the third one provides a new paradigm enabling autonomic context-aware composition to produce application. This allows to preserve the overall consistency of the application in an ambient system. We are now working on enhancing the middleware (third layer) to provide it a smarter mechanism to select the services (with semantically annotated smart things and semantic reasoning) and new algorithms to resolve interferences and conflicts between multiple applications.

Acknowledgements

This work was supported by French Research Agency (Continuum project, ANR-08-VERS-005) and EDF R&D.

We would like to thank persons who contributed to the development of the platform: S. Weibel, V. Hourdin, D. Cheung-Foo-Wo, N. Ferry, C. Andral, A. Bourcet, L. Chungue, J. Fuchet, G. Joulie, and N. Pandolfo for his work on demonstrator and video.

References

1. Atzori, L., Iera, A., & Morabito, G. (2010). The internet of things: A survey. *Computer networks*, 54(15), 2787-2805.
2. Zeng, D., Guo, S., & Cheng, Z. (2011). The web of things: A survey. *Journal of Communications*, 6(6), 424-438.
3. Hourdin, V., Tigli, J.-Y., Lavirotte, S., Rey, G., & Riveill M. (2008). SLCA, composite services for ubiquitous computing. In *Proceedings of the 5th Int. Conf. on Mobile Technology, Applications and Systems* (Mobility'08).
4. Cheung-Foo-Wo, D., Tigli, J.-Y., Lavirotte, S. & Riveill, M. (2007) Self-adaptation of event-driven component-oriented middleware using Aspects of Assembly. In *Proceedings of the 5th Int. Workshop on Middleware for Pervasive and Ad-Hoc Computing* (MPAC'07), 31-36.
5. Fathallah, S., Lavirotte, S., Tigli, J-Y., Rey G. & Riveill M. (2011) MergeIA: a service for dynamic merging of interfering adaptations in ubiquitous system. In *Proceedings of the Fifth Int. Conf. on Mobile Ubiquitous Computing, Systems, Services and Technologies* (UBICOMM'11).
6. Tigli, J.-Y., Lavirotte, S., Rey, G., Ferry, N., Hourdin, V. Fathallah, S., Vergoni C. & Riveill, M. (2012). Aspects of Assembly: from theory to performance. *LNCS Transactions on Aspect Oriented Software Development* (TAOSD), volume 7271.