Discovery and registration of components in multimodal systems distributed on the IoT

B. Helena RODRIGUEZ
W3C's MMI Working Group Editor
Paris, FRANCE
helena.rodriguez@shopedia.fr

Jean-Claude MOISSINAC
Institut Mines-Telecom
Telecom ParisTech CNRS LTCI
46, rue Barrault 75634 Paris CEDEX 13
moissinac@telecom-paristech.fr

Abstract— **One of the major gaps in the current HTML5 web platform is the lack of interoperable means for an application to discover services and applications available in a given space and network. This problem is shared by the multimodal applications developed with web technologies, for example, in smart houses or applications for the Internet of Things. To address this gap, we produced a SOA approach for the W3C's Multimodal Working Group that aims to allow the discovery and registration of components used in multimodal interaction systems in the web of things. In this approach, the components are described and virtualized in a dedicated module communicating with two dedicated events, and registering components in a Resources Manager to facilitate the fine management of concurrent multimodal interactions, and the interoperable discovery, registration and filtering of features provided by heterogeneous and dynamic components in the web of things.**

Keywords-component; Multimodal Interaction, Semantic Interaction, Interface Services, MMI Architecture and Interfaces, Pervasive computing, Context awareness

I. Introduction

The Multimodal Architecture and Interfaces (MMI-Arch) is a current Recommendation of the World Wide Consortium [1] introducing a generic structure and a communication protocol to allow the components in a multimodal system to communicate with each other. It proposes also a generic event-driven architecture and a general frame of reference focused exclusively in the control of the flow of data messages. This frame of reference has been proposed due to a lack of distributed approaches for multimodal systems "in the cloud". These approaches are mostly produced in ad-hoc solutions, as shown by a state of the art of 100 relevant multimodal systems where it was observed that more than 97% of the systems had little or no discovery and registration support [2].

At the time, even the W3C's MMI Architecture and Interfaces (MMI-Arch) and its runtime framework [3] failed to address: 1) the component's discovery and registration to support fusion (integration) and fission (composition) mechanisms, 2) the modality component's data model needed by this registry and 3) the modality component's annotation to facilitate the orchestration (and even the turn-taking) mechanism.

These three issues are addressed and to some extent resolved by our SOA proposal, which is now adopted as a W3C's recommendation.

Thus, our proposal has become an interoperable extension for the MMI-Arch's model, designed to support the automation of the discovery, registration and composition of multimodal semantic services. It is also designed to fulfill the requirements of high-level Quality of Service (QoS) like: the accurate selection of components when these are not available anymore, do not meet the expected functionality or disrupt the context of use.

With these goals in mind, our contribution was structured on three parts: 1) a new addressing method needed for the component's announcement at bootstrapping; 2) an architectural extension in order to support the handling of the state of the multimodal system using a virtual component approach for registration and 3) two new events for the messaging mechanism, to address the requirements of discovery and registration on distributed systems.

These three parts are currently completed by the creation of a common and interoperable vocabulary of states and generic features to allow the gross

DOI: 10.18293/DMS2015-39

discovery of modalities in large-networks over a concrete networking layer [4]

In the following sections we will present our contribution as follows: In Sec. 2 we will give an overview of the problem, followed by a study of the related work in Sec 3. In Sec. 4 we describe our work on Discovery and Registration and finally, we present a conclusion and some perspectives to continue this work.

II. PROBLEM STATEMENT

Historically, multimodal systems were implemented in stable and well-known environments. Its complexity demanded laboratory-like implementation and very few experiences were developed for real-time contexts or component distribution. But this situation has evolved. The web developer's community is progressively confronted with the problem of modality integration in large-scale networks which is expected to be huge in the years to come, when the Web of Things will attain a state of maturity.

The increasing amount of user-produced and collected data will also require a more dynamic software behavior with a more adequate approach 1) to handle the user's technical environment where the demand for energy supply is getting higher and higher, and 2) to encourage and improve the efficiency in consumption boosting the creation of systems compatible with smart-grid technologies.

For example, in Japan [5] (as in European countries) the distributed applications will play a very strategic role in the reduction of energy consumption, helping to evolve to an on-demand model. With this goal, the sustainable consumption in houses must be handled and analyzed distantly, using data collected by multimodal applications installed on multiple kinds of devices of the Internet of Things.

These applications must interact in a coordinated manner in order to improve the energetic efficiency of the application behavior, to collaborate in the home automation management and in some cases, even the user profiling; the whole with a distributed platform.

Thus, we face the raising of multiple issues, concerning the multimodal user interaction with very heterogeneous types of devices (some of them with low resources), protocols and messaging mechanisms to be synchronized in an interoperable way.

In this context, on one side, modality discovery and selection for distributed applications becomes a new working horizon giving new challenges for multimodal systems, user-centric design and the web research. And in the other side, generic and interoperable web approaches, using web technologies but capable of going beyond the browser model, will be unavoidable.

A. *Multimodal Discovery and Registration*

Multimodal systems are computer systems endowed with rich capabilities for human-machine interaction and able to interpret information from various communication modes. According to [6] the three principal features of multimodal systems are: 1) the fusion of different types of data; 2) real-time processing and temporal constraints imposed on information processing; 3) the fission of restituted data: a process for realizing an abstract message through output on some combination of the available channels.

On these systems, modality management is mostly of the time hard-coded, leaving aside the problem of a generic architecture respond to extensibility issues and the need of discovery, monitoring and coordination of modalities in real-time with context- awareness. Consequently, multimodal applications were manually composed by developers and shared via web APIs and embedded web technologies, in an ad-hoc and proprietary way.

To address this lack of a generic approach, the MMI Architecture proposes an architectural pattern for any system communicating with the user through different modalities simultaneously instantiated in the same interaction cycle. In this unique context of interaction the final user can dynamically switch modalities. This kind of bi-directional system combines inputs and outputs in multiple sensorial modes and modalities (e.g. voice, gesture, handwriting, biometrics capture, temperature sensing, etc) and can be used to identify the meaning of the user's behavior or to compose intelligently a more adapted, relevant and pertinent message.

Other important characteristic of the Multimodal Architecture and Interfaces specification is that it uses the MVC design pattern generalizing the View to the broader context of the multimodal interaction presentation, where the information can be rendered in a combination of various modalities. Thus, the MMI recommendation distinguishes (Fig. 1) three types of components: the Interaction Manager, the Data Component and the Modality Components.

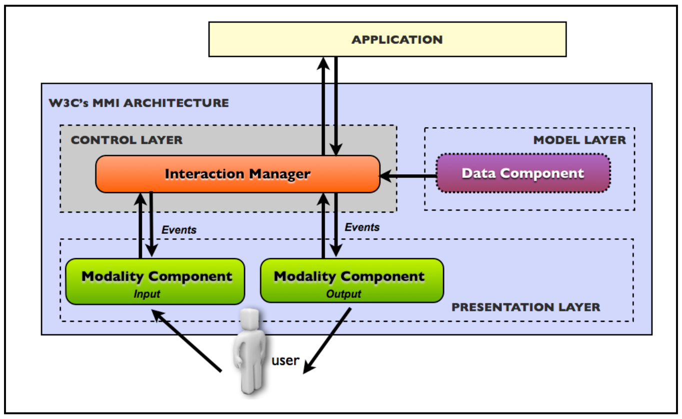

Figure 1. The W3C's Multimodal Architecture

The Interaction Manager is a logical component responsible of the integration and composition of the interaction cycles following multimodal rules. It handles all the exchanges between the components of the Multimodal System and the hosting runtime framework. To ensure some of these tasks, the Interaction Manager must access the data stored in a second component representing the Model, the Data Component, which is a logic entity that stores the public and private data of any module or the global data of the System.

Finally, in the MMI Architecture, the term Modality covers the forms of representing information in a known and recognizable rendered structure. For example, acoustic data can be expressed as a musical sound modality (e.g. a human singing) or as a speech modality (e.g. a human talking).

The component representing the presentation layer in the MMI Architecture is indeed, a Modality Component. This is a logic entity that handles the input and output of different hardware devices (e.g. microphone, graphic tablet, keyboard) or software services (e.g. motion detection, biometrics sensing).

Modality Components are also loosely coupled software modules that may be either co-resident on a device or distributed across a network. This aspect promotes low dependence between Modality Components, reducing the impact of changes and facilitating their reuse. In result, these components have little or no knowledge of the functioning of any other modules and the communication between modules is done through the exchange of events following a protocol provided by the MMI architecture.

Nevertheless, the architecture focuses only on the interaction cycle, leaving aside 1) the support of the lifecycle of Modality Components -from a system perspective-, 2) a more dynamic application behavior, and 3) the non- functional goals of some features to adapt the application to a particular space, device family and interaction type, using context-aware techniques.

As a result, we decided to extend the architecture's model in our work with the MMI Working Group in order to address the lack of an interoperable frame of reference to handle the runtime system's lifecycle that includes the dynamical discovery and registration of Modality Components, as we will see on Sect. 4.

B. *Beyond the Browser*

Today, there is an enormous variety and quantity of devices interacting with each other and with services in the cloud: 7 trillion wireless devices will be serving 7 billion people in the 5 years to come [7]. Web technologies are expected to be at the center of the Internet of Things (IoT), thanks to their universal adoption and huge scalability. Nevertheless the definition of a standardized programming model for objects beyond the page-browser mechanism has not been established yet, and the classical internet of documents or the internet of knowledge has being built with a series of architectural premises that could be inadequate and even a foundational obstacle to this new challenge.

To address it, device-centric technologies, proposals and protocols are spreading all over the current discussion around the Web of Things, assuming that the "infinite things problem" will be resolved by creating “virtual images” of this reality on IoT systems. But this solution will just transfer a real-world problem to a virtualized one, with the concurrency of policies, architectures, platforms, protocols and standards that such a transfer implies.

On one hand, browser vendors are advocating for browser-based solutions assuming that a model that works well for web pages (based on the document model) and web apps (mostly based on client- server models) in computers and mobiles, can be easily extended to any other kind of objects.

But, how can we model a rice cooker as a document? Is it really logical to communicate with an air conditioner as a "data resource" ? How to apply a client-server model to reflexive objects in the network, acting at the same time as server and clients of their own services? Are addresses registers of devices as stable as the addresses directories of web pages or web apps? How to express on, off or stand-by states with web technologies? And what will be the environmental and energetical price to this choices?

On the other hand device vendors are advocating for energy efficient and lightweight protocols fine-tailored for constrained devices; and willing to provide a web gateway to allow the communication between these devices and the web. After near 20 years of research, some of the industrial consortiums leaded by energy providers and device vendors built a series of low-level protocols and technologies supported by national policies: KNX [8], ZigBee SEP 2.0 [9], Z-Wave [10], Echonet [11], ETSI M2M [12], DLNA [13], UPnP[14], ZeroConf [15], etc.

As the above list shows, these concurrent protocols and technologies have to be evaluated and selected by a developer or a new device producer. If this panorama of exploded technologies continues, the situation that mobile developers endured during years will reappear in the web of objects: heterogeneous operating system, SKD's, app distribution circuits and developing models for an infinity of objects.

To sum up, there is a real and urgent need of a vendor- agnostic model of components and communication, to encompass the diversity of proposals and technologies in the Internet of Things, and the need of generic devices to reduce the effort of implementation for developers and app vendors.

Our effort in the MMI Working Group, has been always focused to evolve Web technologies from device-centric applications, to natural interaction experience and user-centered models that will extend the definition of an application to seamlessly encompass multiple heterogeneous devices collaborating and sharing resources and computational capabilities, both locally and across the web.

As an illustration of the problem, in a multimodal system devices may contain nested logical devices, as well as functional units, or services. A functional categorization of devices is currently defined by the UPnP protocol with 59 standardized device templates and a generic template profile, the Basic Device. With the same spirit, the Echonet consortium defines a number of 7 Device Groups for multiple Classes of Devices while Zigbee SEP 2.0 defines 20 device categories. In the three cases, the device specification defines explicitly the device's properties and access methods. In contrast, more generic protocols, like Z-wave use the Generic device approach and 3 abstract classes of Devices. Device classifications are provided also by The Composite Capability Preference Profiles Specification of the W3C or even with the User Agent Profile Specification extension of CC/PP [16] maintained by the Open Mobile Alliance (formerly the WAP Forum) with the Specification's Part 7: Digital Item Adaptation, in which Terminals and Terminals capabilities are described.

It is also possible to leverage the current work of the W3C's Device APIs Working Group [17], which is working on a set of heterogeneous deliverables going from the device object level to very specific features, browser extensions, HTML5 extensions and event networking issues: Vibration API, Battery Status API, HTML Media Capture, Proximity Events, Ambient Light Events, Media Capture and Streams, MediaStream Image Capture, Media Capture Depth Stream Extensions, Network Service Discovery (HTTP- based services advertised via common discovery protocols within the current network), Wake Lock API, Menu API and the sensor API to come.

This example showing the device description proposals, illustrates the concurrency of concerns, approaches and proprietary interests around the "thing" indexing and registration problem. This work can be made more extensible and less driven by the specific capabilities of today's mobile devices by aligning it with the generic, device-independent Multimodal Interfaces API. It would also be very useful to integrate these proposals with the taxonomic efforts already made by consortia like Echonet during the last 20 years in a common and standardized vocabulary and generic API.

We can imagine that the horizon opened by the web of things is as exponential as the technical solutions currently available. This situation explains and supports the MMI Working Group generic approach and, as we will present on the following sections, defines our proposal for discovery and registration of Modality Components.

III. OVERVIEW OF RELATED WORK

In a previous work [2] we studied a sample of 16 multimodal architectures that were selected from a previous analysis of a larger set (100) of multimodal implementations. The selection criteria has being the amount of information provided by the authors about the architectural facets of the implementation, its completeness and its representativeness of three domains of research: distribution, the modality description and the use of semantic technologies.

In the following section we will present a first group of emerging trends directly related to the criteria of discovery and registration, and later, a second group transversal to the same criteria.

A. *Emerging trends related to the criteria*

1) *EVENT HANDLING.*

The first recurrent topic is event handling. Seven architectures tried to address the management of events, which is normal in the human computer interaction research because user interfaces are highly event-oriented.

The event management concerns are resolved with seven different techniques. In OAA [18], triggers provide a general mechanism to express conditional requests. Each agent in the architecture can install triggers either locally, on itself, or remotely on its facilitator or peer agents. There are four types of triggers: communication triggers; data triggers; task triggers; and time triggers.

GALATEA [19] uses macro-commands while an Agent Manager that possesses a macro-command interpreter expands each received macro-command in a sequence of commands and sending them sequentially to the designated modules. With task control layers in OPENINTERFACE [20], communication paradigms (event- based, remote procedure call, pipe, etc) are implemented with adapters/connectors using rules for instantaneous events and persistent events. In MEDITOR [21], events are handled with three specialized managers: the input messages queue, the input messages generator and the output messages generator. The temporal order is ensured and disambiguation is handled with a routing table and predefined rules. Hardwired Reactions are the tool in REA [22], for quick reactions to stimuli. These stimuli then produce a modification of the agent's behavior without much delay, as predefined events.

In DIRECTOR [23] events are handled at the level of pipeline execution –continuous- and at the level of scripting –discrete-. In HEPHAISTK [24], events are handled by the Event Manager, which ensures the temporal order of events. The client application is a client, but also is another input source, and consequently the Event Manager is needed also as a recognition agent, which communicates through a set of predefined messages.

In contrast, the MMI Architecture responds to the same concern with the Interaction Life-Cycle Events, and the proposal of a dedicated component: the Interaction Manager. This solution provides a clear separation between the interaction control and the interaction content data, but hardwired mechanisms are not envisioned, neither the transport queue mechanism implemented in MEDITOR, GPAC [25] and HEPHAISTK that can be an important support for the fusion / fission of modalities. In consequence, these mechanisms were detected as possible extensions to the W3C's Architecture to provide some complementary resources to handle multimodal events in an interoperable way.

2) *STATE MANAGEMENT*

The second key topic, recovered from 5 of the sampled architectures is the state management. It corresponds also to the session management. This feature is oriented to register the evolution of the interaction cycle and provides the information about any modification of the state of the system and the components. It is designed as a monitoring process in support of the decision layer (SMARTKOM [26], HEPHAISTK), as a display list manager in support of the fusion and fission mechanisms (DIRECTOR), as a blackboard (OAA, HEPHAISTK), a central place where all data coming from the different sources are standardized, and other interested agents can dig them at will.

Finally, the states are handled by an object manager -for decoding and rendering purposes- (GPAC), and even as a routing table (MEDITOR). Concerning this subject the MMI Framework recommends a specific component to handle the multimodal session and the state of components; yet, it does not give details about the interfaces needed to use this component or about its role in the management of the interaction cycles. As a result, an extension to the MMI Architecture can be conceived to complete this generic description with specific details about the eventual implementation, behavior and responsibilities of this state manager.

B. *Emerging trends transversal to the criteria*

1) *GENERIC MODELS*

The first transversal key topic is the definition of models: 12 of our architectures proposed interesting approaches concerning the modeling of the entities that participate in the multimodal interaction. However, only SMARTKOM addresses the modeling task with a proposal coming from web semantic technologies.

In addition, depending on the modeled entity, the models are more or less expressive or homogeneous, and consequently, usable. The modeling of the multimodal interaction phenomenon (SMARTKOM, HEPHAISTK, MEDITOR), the task (GALATEA, OPENINTERFACE, SQUIDY [27]), the dialog interaction (REA, GALATEA, SMARTKOM), and the devices (SMARTKOM) is more extensive, tested and advanced than the modeling of the user (REA), the application (OAA, SMARTKOM, ELOQUENCE, GPAC, HEPHAISTK) or the environment & context of usage (SMARTKOM) conceived to support and enrich the multimodal interaction.

This growing and common interest on models - expressed in SMARTKOM as a foundational principle, opens the way to reinforce the MMI recommendation with an effort to address this issue and to see how the MMI Framework & Architecture can respond to data modeling needs.

2) *DISTRIBUTED ARCHITECTURES*

The second transversal topic is distribution. It is tackled with solutions like the remote installation of triggers (OAA), the distribution of the fusion-fission mechanisms into nodes and components (OPENINTERFACE, SQUIDY) that can even be external to the multimodal system, the management of inputs as “sensed” data (input sensors) or as broadcasted media containing behavior (and interaction) information in the distributed streams (GPAC); and finally, the distribution of application services (SMARTKOM, HEPHAISTK). This topic is also reflected on the service-oriented proposals of application services and services advertisement (OAA, SMARTKOM) and the networking services layer to manage the broadcasted input and output data of a rich application (GPAC). The MMI Framework & Architecture reflects this topic in its distributed nature based on web standards. Nevertheless, there are few current implementations using the web services or a service-oriented approach from a distributed perspective.

The current implementations are oriented to prototype mobile interfaces (Orange Labs), to provide a multimodal mobile client for health monitoring (Openstream), to test an authoring tool (Deutsche Telekom R&D) and to complete JVoiceXML, an open source platform for voice interpretation (TU Darmstadt). We believe that it is possible that interesting extensions arise from a fully SOA implementation of the MMI Framework & Architecture standard according with its distributed nature ant the needs that are appearing with the Internet of Things.

3) *CONTROL DELEGATION*

A final transversal topic is the delegation of the interaction management by a client application. It is present in the form of application agents (CICERO, OAA) or application services (SMARTKOM, HEPHAISTK). The MMI Framework & Architecture does not deal with this subject because the application is meant to be the concrete implementation of the architecture. A delegation approach supposes that an external functional core can delegate the management of the interaction to a multimodal system built in accordance with the standard, and providing multimodal functionalities to the client application installed on devices with low processing capabilities.

This approach is not currently addressed, even if it could be the type of requirement of a multimodal browser, a home gateway virtualization or an IoT web application. Our current work on the W3C

MMI Working Group addresses the possible extensions that such approach could bring and how the MMI Architecture standard can support this type of future implementation.

In short, the study of the related work allow us to structure and define our collaboration in the W3C Multimodal Working Group, to extend the MMI Architecture with a proposal oriented to facilitate the distributed implementations coming from the Internet of Things.

IV. DISCOVERY & REGISTRATION FOR MMI SYSTEMS

To the best of our knowledge, there is no standardized way to build a web multimodal application that can dynamically combine and control discovered components by querying a registry build based on the modality states. At the same time -as we showed in Sect. 3 - research efforts also lack of this distributed perspective. Based on this previous analysis, we decide to focus on three complementary extensions to the MMI Architecture: 1) we propose to complete the current addressing method in order to evolve from a client-server model to an anycast model. 2) We propose to reinforce the management of the “multimodal session”, and more precisely, a dedicated component to handle the system’s state and support the system’s virtualization of components. And 3) we propose to extend the transport layer with two new events designed to complete and reinforce the interaction Lifecycle Events.

A. *Extending the MMI addressing methods*

To inform the system about the changes in the state of the Modality Components, an adaptive addressing mechanism is needed. We consider that the combination of push/pull mechanisms is crucial to extend the MMI Architecture to the Web of Things. For example, in the case of the unavailability of a given Modality Component, it needs to communicate with the control layer. This situation is not necessarily related to the interaction context itself, but it can affect it, because the interaction cycle can be stopped or updated according to this change on the global state of the system.

In the current state of the Multimodal Architecture Specification [1], interaction events like Prepare or Start, must be triggered only by the Interaction Manager and sent to the Modality Components. In result, a Modality Component cannot send messages to the Interaction Manager other than the message beginning the interaction cycle: the newContext event. Any other event originated by an internal command or like in our example, by a change on the component’s state cannot be raised. Nevertheless, to start an interaction cycle the Modality Component needs to be already part of the system and to be registered. The registration process is part of a previous phase, when even the presence of the user is not mandatory and the communication must be bidirectional.

As Modality Components are reflexive objects in the network acting at the same time as server and clients, they need to communicate and to receive messages as well. The flow of messages always initiated by the Interaction Manager is not sufficient to address use cases evolving in dynamic environments, like personal externalized interfaces, smart cars, home gateways, interactive spaces or in-office assistance applications. In all these cases, Modality Components enter and quit the multimodal system dynamically, and they must declare their existence, availability and capabilities to the system in some way.

To address this need, we proposed our first extension, which is a bidirectional flow of messages to support a complete number of addressing methods and to preserve a register of the system’s global state. One of the results of this new flow of messages is the capability to produce the advertisement of Modality Components. It allows the Multimodal System to reach correctness in the Modality Components retrieval and also affects the completeness in the Modality Component retrieval. To return all matching instances corresponding to the user's request, the request criteria must match some information previously registered before the interaction cycle starts.

For this reason, the MMI Architecture should provide a means for multimodal applications to announce the Modality Component’s presence and state. This was the first step to address the distribution requirement: Modality Components can be distributed in a centralized way, a hybrid way or a fully decentralized way.

For the Discovery & Registration purposes the distribution of the Modality Components influences how many requests the Multimodal System can handle in a given time interval, and how efficiently it can execute these requests. Even if the MMI Architecture Specification is distribution-agnostic, with this extension Modality Components can be located anywhere and communicate their state and their availability to new a dedicated component: the Resources Manager.

B. *Extending the MMI Architecture's modules*

The new flow of messages between the Modality Components and the control layer needed a mechanism tracing the relevant data about the session and the system state. This is the first of the responsibilities for the second extension, the Resources Manager. This manager is responsible for handling the evolution of the "multimodal session" and the modifications in any of the participants of the system that could affect its global state. It is also aware of the system's capabilities, the address and features of modalities, their availability and their processing state. Thus, the Resources Manager is nested in the control layer of the multimodal system and keeps the control of the global state and resources of the system. And the extended control layer encompasses the handling of the multimodal interaction and the management of the resources on the multimodal system. In this way, with our extension, the architecture preserves its compliance with the MVC design pattern.

The data handled by the Resources Manager can be structured and stored in a virtualized manner. In this way, the Resources Manager can be calibrated for mediated discovery -and federated registering-. The Resources Manager uses the scanning features provided by the underlying network, looking for components tagged in their descriptions with a specific group label. If the discovered component is not tagged with a group label, the Resources Manager can use some mechanism provided to allow subscriptions to a generic group. In this case, the Modality Component should send a request using the new flow of messages and using one of the new discovery events to the Resources Manager, subscribing to the register of the generic group.

In this way, the Resources Manager translates the Modality Component's messages into method calls on the Data Component, like the MVC pattern proposes but also, the Resources Manager broadcasts to the Modality Component the changes on the system's state or notifies it following a subscription mechanism. Upon reception of the notification, the Modality Component updates the user interface according to the information received.

The Resources Manager supports the coordination between virtualized distributed agents and their communication through the control layer. This enables to synchronize the input constraints across modalities and also enhances the resolution of input conflicts from distributed modalities. It is also the starting point to declare and process the advertised announcements and to keep them up to date and the core support for mediated and passive discovery and it can also be used to trigger active discovery using the push mechanism or to execute some of the tasks on fixed discovery [4]. The Resources Manager is also the interface that can be requested to register the Modality Component's information. It handles all the communication between them and the registry. The flow of discovery queries transit through it, which dispatches the requests to the Data Component and notifies the Interaction Manager if needed. These queries must be produced using the state handling events presented on the next Section.

To summarize, the Resources Manager delivers information about the state and resources of the multimodal system during and outside the interaction cycle.

C. *Extending the MMI Event model*

With a new flow of messages and a new component handling the state of the system, a Modality Component can register its services for a specific period of time. This is the basis for the handling of the Modality Component's state. Every Modality Component can have a lifetime, which begins at discovery and ends at a date provided at registration. If the Modality Component does not re-register the service before its lifetime expires, the Modality Component's index is purged. This depends on the parameters given by the Application logic, the distribution of the Modality Components or the context of interaction.

When the lifetime has no end, the Modality Component is part of the multimodal system indefinitely. In contrast, in more dynamic environments, a limited lifetime can be associated with it, and if it is not renewed before expiration, the Modality Component will be assumed to no longer be part of the multimodal system. Thus, by the use of this kind of registering, the multimodal system can implement a procedure to confirm its global state and update the «inventory» of the components that could eventually participate in the interaction cycle. Therefore, registering involves some Modality Components' timeout information, which can be always exchanged between components and, in the case of a dynamic environment, can be updated from time to time. For this reason, a registration renewal mechanism is needed. We proposed a registration mechanism based on the use of a timeout attribute and two new events: the checkUpdate Event and the UpdateNotification, used in conjunction with an automatic process that ensures periodical requests.

The checkUpdate Event is provided a) to verify if there are any changes in the system side; b) to recover the eventual message; c) to adapt the request timeout if needed and d) to trigger automatic notifications about the state of the Modality Component, if the automaticUpdate field in the response is true. If a Modality Component is waiting for some processing provided by other distributed component, the checkUpdate Event allows the recovery of progressive information and the fine-tuning of requests by changing the timeout attribute. This enhances input/output synchronization in distributed environments.

On the other hand, the Update Notification is proposed a) to periodically inform the Resources Manager about the state of the Modality Component; b) to help in the decision making process (on the server side, for example).

For notification of failures, progress or delays in distributed processing the Update Notification (Fig. 4) ensures periodical requests informing other components if any important change occurs in the Modality Component's state. This can support, for example, grammar updates or image recognition updates for a subset of differential data (the general recognized image is the same but one little part of the image has changed, e. g. the face is the same but there is a smile)

The use of the timeout attribute helps in the management of the validity of the advertised data. If a Modality Component's communication is out-of-date, the system can infer that the data has the risk of being inaccurate or invalid. The checkUpdate Event allows the recovery of small subsets of the information provided by the interaction manager, to maintain up to date the data in the Modality Components as in the Resources Manager.

V. CONCLUSION

The work on standardization produced by the MMI Working Group in the last two years and its focus on distribution has been fruitful.

Today, the first step needed to allow the component's discovery and registration helping for a more adaptive fusion and fission mechanisms is done. The Components have now a means to announce its capabilities and states through different addressing modes. Second, the modality component's data model needed as a building block for a multimodal registry is founded, starting by a common taxonomy of generic states (out of the scope of this document, but available in [5]) (Fig.8) and the construction of a generic classification system for devices and groups of modes and modalities. This premises of classification will allow facilitate the orchestration mechanism with the Modality Component's annotation. A mechanism that is now possible, thanks to the extension of the MMI Architecture's event model with two events specified for discovery and registration needs.

These three issues are covered by our results and now are entering on the W3C's recommendation processes to be available to the community of web developers. From the requirements extracted from the analysis of the state of the art and a series of use cases provided by the industry [4], we produce three pertinent extensions.

To handle multimodal events (See Sect. 3-A-1) in an interoperable way, we extend the MMI Architecture by completing the current addressing method in order to evolve from a client-server model to an anycast model using bidirectional communication.

To ensure the handling of states (See Sect. 3-A-2), we proposed to support the management of the "multimodal session" by a dedicated component using a virtualization of components to reflect the current state of the system.

To allow distribution (See Sect. 3-B-2), we proposed to extend the transport layer with two new events completing and reinforcing the interaction Lifecycle Events. An finally, to support the delegation of control (See Sect. 3-B-3) and to use generic models (See Sect. 3-B-1), we proposed a virtualization mechanism used to create and store the registry, based on generic multimodal properties, a generic model of states needed for keeping the registry up-to-date and a generic vocabulary for the description of Modality Components.

The MMI's Modality Component is an abstraction flexible enough for any implementation of the Internet of Things and networking model, while keeping an interoperable structure. The MMI Architecture is built around the management of continuous media and their states not only as outputs (presentations) but also as inputs. This means that the architecture is fine-tuned to handle issues derived from very dynamic environments needing session control and recovering with all kinds of medias and interaction modes.

In this paper we presented our current work on Discovery and Registration of Modality Components from a generic and interoperable technology that will allow us to face the infinity created by the web of things. From an extensive study of the state of the art, we produced a series of requirements and evaluation criteria that founds the proposal presented on Sect. 4, which is now a W3C's Recommendation.

In a future activity the W3C working group will produce an annotation vocabulary and the support of the semantic annotation in the "info" dedicated attribute on the new discovery and registration events. This vocabulary is a first step on the direction of a more expressive annotation of the interaction with Modality Components using ontologies and a more intelligent composition of semantic web services for multimodal applications with rich interaction features.

[1] MMI-Arch:http://www.w3.org/TR/mmi-arch/ Visited at :01/04/2015

[2] B. H. Rodriguez. "A SOA model, semantic and multimodal, and its support for the discovery and registration of assistance services". PhD Thesis, Institut Mines-Télécom, Telecom ParisTech, Paris, 2013.

[3] Multimodal Interaction Framework http://www.w3.org/TR/mmi-framework/ Visited at :01/04/2015

[4] B.H. Rodriguez (Ed)., D.Dahl, R. Tumuluri, P. Wiechno and K. Ashimura. Registration & Discovery of Multimodal Modality Components in Multimodal Systems: Use Cases and Requirements. W3C Working Group Note 5 July 2012. Available at: http://www.w3.org/TR/mmi-discovery/ Visited at :01/04/2015

[5] International Symposium on Home Energy Management System -Joint discussion with the W3C MMI WG -, Keio University Shonan Fujisawa Research Institute, 25-26 Feb., 2015

[6] J. Coutaz, L. Nigay, D. Salber, A. Blandford, J. May and R. Young "Four easy pieces for assessing the usability of multimodal interaction: the CARE properties" In: Proceedings of INTERACT'95Lillehammer, June 1995

[7] W3C Workshop on the Web of Things Enablers and services for an open Web of Devices, 25–26 June 2014, Berlin, Germany http://www.w3.org/2014/02/wot/

[8] KNX network communications protocol for intelligent buildings (EN 50090, ISO/IEC 14543) http://www.knx.org/knx-en/index.php

[9] The Smart Energy Profile 2 http://www.zigbee.org/zigbee-fordevelopers/applicationstandards/zigbeesmartenergy/

[10] Z-Wave http://www.z-wave.com

[11] ECHONET Energy Conservation and HomecareNetwork. http://www.echonet.gr.jp/

[12] ETSI Machine to machine communication. http://www.etsi.org/technologies-clusters/technologies/m2m

[13] Digital Living Network Alliance.http://www.dlna.org

[14] Universal Plug and Play http://www.upnp.org

[15] Zero Configuration Networking https://developer.apple.com/bonjour/index.html

[16] Composite Capabilities/Preference Profiles http://www.w3.org/Mobile/CCPP/

[17] Device APIs Working Group. http://www.w3.org/2009/dap/#roadmap

[18] D. Martin, A. Cheyer, and D. Moran, "The Open Agent Architecture: A Framework for Building Distributed Software Systems," Applied Artificial Intelligence, Volume 13, Number 1-2, January-March 1999, pp. 91-128.

[19] NITTA. Activities of Interactive Speech Technology Consortium (ISTC) Targeting Open Software Development for MMI Systems

[20] M. Serrano, L. Nigay, J-Y. Lawson, L. Ramsay, and S. Denef "The openInterface framework: a tool for multimodal interaction" In: CHI '08 extended abstracts on Human factors in computing systems -CHI EA08. ACM, New York, NY, USA. p.p.3501-3506.

[21] Y. Bellik Interfaces Multimodales : Concepts, Modèles et Architectures. PhD Thesis, University Paris-South 11, Orsay, 1995.

[22] J. Cassell "Embodied Conversational Agents.Representation and Intelligence in User Interfaces" In: AI Magazine, 2001. Vol. 22. No.4.

[23] http://en.wikipedia.org/wiki/Adobe_Director Visited at :01/04/2015

[24] B. Dumas, D.Lalanne, and R. Ingol. Démonstration: HephaisTK, une boîte à outils pour le prototypage d'interfaces multimodales, 2008.

[25] J. Le Feuvre et al., Experimenting with Multimedia Advances using GPAC, ACM Multimedia, Scottsdale, USA, November 2011 http://dl.acm.org/citation.cfm?doid=2072298.2072427

[26] Herzog, Gerd and Reithinger, Norbert. "The SmartKom Architecture: A Framework for Multimodal Dialogue Systems" In: SmartKom: Foundations of Multimodal Dialogue Systems, 2006. Springer Berlin Heidelberg, p.p. 55-70. http://dx.doi.org/10.1007/3-540-36678-4_4

[27] Werner A. König, Roman Rädle, and Harald Reiterer. 2009. Squidy: a zoomable design environment for natural user interfaces. In Proceedings of the 27th international conference extended abstracts on Human factors in computing systems (CHI EA '09). ACM, New York, NY, USA, 4561-4566. DOI=10.1145/1520340.1520700