Playful Supervised Smart Spaces (P3S) A framework for designing, implementing and deploying multisensory play experiences for children with special needs

Giovanni Agosta, Luca Borghese, Carlo Brandolese, Francesco Clasadonte, William Fornaciari, Franca Garzotto and Mirko Gelsomini
Politecnico di Milano – DEIB, Milano, Italy
name.surname@polimi.it

Matteo Grotto
IBT Solutions srl, Milano, Italy
matteo.grotto@ibtsolutions.it

Cristina Frà, Danny Noferi and Massimo Valla
Open Innovation Research JOL S-Cube, Telecom Italia S.p.A., Milano, Italy
name.surname@telecomitalia.it

Abstract

Our research explores novel forms of smart spaces that support full body interaction with smart objects instrumented with audio, light, and motion sensors and actuators, virtual worlds on medium-large displays, and smart lights, and are integrated with cloud services for remote supervision and analysis of user behaviour. In this paper, we present the architecture and coordination infrastructure created in the Playful Supervised Smart Space (P3S) project, as well as initial designs for the Smart Object and Smart Space Gateway Components.

Index Terms

Smart Spaces, Smart Objects, Wireless Sensor Node

1. Introduction

The *Playful Supervised Smart Spaces* (P3S) project aims at improving the quality of life of persons with special needs (in particular, children with motor and intellectual disability or inpatient children) by deploying novel multisensory and multimodal "smart spaces" that can be used at health centers, special education institutions and home. The project, supported by the European Institute of Technology (EIT), started in January 2015 and will be completed by the end of the year. This paper reports on the status of P3S at its intermediate milestone, consisting of a first prototype implementation of the smart space.

From a technological perspective, the general approach of P3S implements, and extends, Mark Weiser's vision of a *smart space* as "*a physical world that is richly and invisibly interwoven with the sensors, actuators, displays and computational elements embedded seamlessly in the everyday objects of our lives, connected through a network*" [1]. In addition, P3S is grounded on theoretical and empirical research in psychology and neuro sciences that pinpoint the relationship between physical activity and cognitive processes, with the formative role of "embodiment"[1] in the development of cognitive skills, and the potential of *light* in promoting people relaxation and well being [2].

P3S smart spaces are characterized by an innovative combination of *smart objects* (toys or everyday items digitally enriched with sensors and actuators), virtual worlds on large screens, smart lights, and cloud services that are orchestrated as a *continuous* space. Using multiple forms of *full-body interaction* — physical manipulation of objects and movement/air gestures [3], [4], [5] — children can control and play with smart objects and on-screen virtual worlds, while *smart lights* (either integrated in the ambient or embedded in the objects) dynamically adapt themselves to the user experience, providing engaging visual feedbacks and creating a relaxing, comfortable atmosphere. Examples of such experiences are shown in Figure 1.

Behavioural data and brain engagement signals of children interacting with P3S smart space elements are automatically gathered and stored on the cloud.

1. Embodiment is defined as the way an organism's sensorimotor capacities enable it to successfully interact with the physical environment.

978-1-4673-8035-5/15 $31.00
DOI 10.1109/DSD.2015.61

Figure 1: Examples of P3S Smart Spaces

Dedicated cloud services enable specialists to exploit these data for analysis and research purposes, to tune a therapy, and to remotely customize the characteristics of the smart spaces to the specific needs of each child.

The main targets for P3S technology are health and special education service providers, in particular those addressing children with intellectual disability (e.g., Autism Spectrum Disorder – ASD, Attention Deficit Hyperactivity Disorder – ADHD, Down Syndrome, or Schizophrenia). By adopting P3S smart spaces, these institutions can offer new forms of game-based behavioural therapy as well as new services that can mitigate the patients' burden of "going on-site" for treatment. By installing a P3S smart space in their living environment and subscribing to P3S integrated cloud services, families will be able to offer to their children therapeutic activities that can be performed at home, with a remote therapist's supervision. A number of smart spaces have been designed in cooperation with therapists from specialized centres in Italy and the Netherlands where prototypes are currently under evaluation involving low-medium functioning autistic children and their caregivers.

The rest of this paper is organized as follows. Section 2 presents the overall architecture of the Playful Supervised Smart Space and its main elements: game logic, smart objects and smart ambient gateway, and depicts the interaction model that rules their behaviour. Finally, in Section 3 some conclusions are drawn and future research directions highlighted.

2. P3S Overall Architecture

The overall system architecture underlying P3S smart spaces is depicted in Figure 2. The main components of the architecture are the *smart ambient*, the *smart objects*, and the *game logic*. The game logic encodes the characteristics of the desired User Experience (UX), whereas the smart objects and the smart ambient provide cyber-physical functionalities (sensing and actuation) to enable it. Specifically, the *smart ambient* consists of an ensemble of connected devices which are part of the play environment, such as lights, carpets, and wall installations as well as audio devices, whereas the *smart objects* are toys enhanced with sensors and actuators.

The three components interact by exchanging messages, which are specified using the *Game Description Language* (GDL). The GDL messages encode at a high level of abstraction the sensor information and the actuation commands exchanged among the three components. A *GDL Message Dispatcher* serves as the glue between the components, by collecting and distributing the messages.

Unique identifiers are used within the GDL messages to denote the smart objects, the smart ambient, and the game logic itself. GDL messages are divided into *events* and *commands*. Events can be originated by any component, and are composed of a set of properties (in the form of *key-value pairs*), among which must necessarily appear the originator identifier. Commands can only be generated by the game logic, and take a similar form to events, except that the identifier property is used to denote the destination, and must therefore be present. A more detailed description of the GDL and its implementation will be discussed in Section 2.4. The GDL Message Dispatcher serves as a central point of registration for all the components, as well as a message dispatcher for commands.

In the following, we describe the three main components of the P3S smart space in greater detail, as well as the interaction among them through the GDL.

2.1. Game Logic

The game logic can be seen as a (hierarchical) Finite State Transducer (FST), i.e. an automaton with both an input and an output tape. The hierarchical property, while not necessary per se, is a useful abstraction to allow modular composition of game fragments.

Each game or game fragment is therefore described as a tuple $< Q, \Sigma, \Gamma, I, F, \delta >$ where:

- Q is the set of states of the game;
- Σ is the input alphabet, which is composed of the available sensor inputs;
- Γ is the output alphabet, which is composed of the available actuator actions;
- I is the set of initial states (we may want to set $|I| = 1$, both for easier determinization and because there we do not foresee no game logics that can non-deterministically start from different states);
- F is the set of final states.
- δ is the transition function, which maps a pair in $Q \times \Sigma$ to a pair in $Q \times \Gamma$.

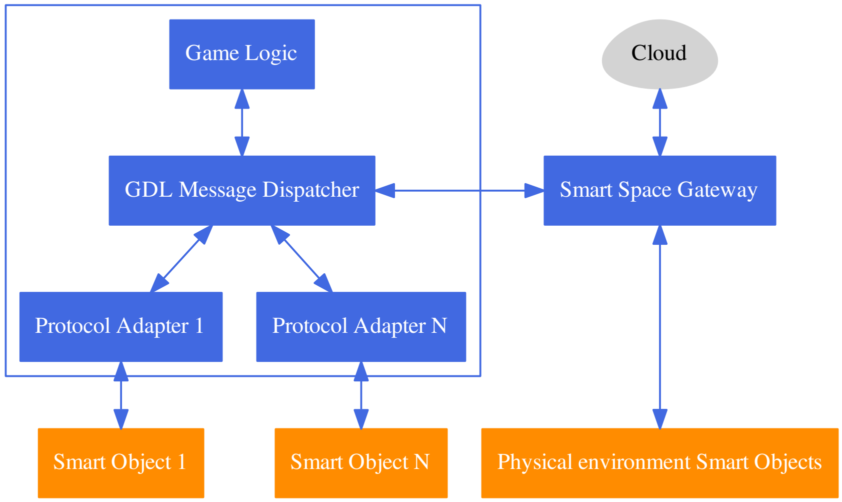

Figure 2: Overall Architecture of the Playful Supervised Smart Space

To avoid employing ϵ transitions[2], which tend to bloat the FST, for practical purposes we prefer to expand the definition of the input and output alphabets to include arbitrary sequences of sensor and actuator actions:

- Σ' the input alphabet, composed of sequences of sensing actions, $\Sigma' = \{(\sigma_1 \dots \sigma_n) | \sigma_i \in \Sigma \wedge n \leq |\Sigma|\}$;
- Γ' is the output alphabet, composed of sequences of actuator actions $\Gamma' = \{(\gamma_1 \dots \gamma_n) | \gamma_i \in \Gamma \wedge n \leq |\Gamma|\}$.

2.1.1. Example

A simple example of game logic is shown in Figure 3. In this game, the smart space is first initialised to a specific ambient effect (in this case, a high level description, "beach", is used — the smart space gateway encodes the appropriate actions to emulate the requested environment), and the object used in the game (a cube with faces which can be actuated with multi-coloured LED lights) is set to emit red light on one of its faces. The game shows an initial image to prompt the player to respond, employing an icon specific to the object part. Transitions from the question state happen when the cube is moved, and lead to a correct or incorrect answer depending on whether the goal face is oriented upwards or not. In case of a correct answer, the lighting of the goal face is changed, and the smart space is requested to change the ambient effect to "jungle", signaling that the game is over. Otherwise, the game signals an incorrect answer and restates the request.

2. ϵ transition or *spontaneous moves* are transitions that are not triggered by reading an input symbol [6].

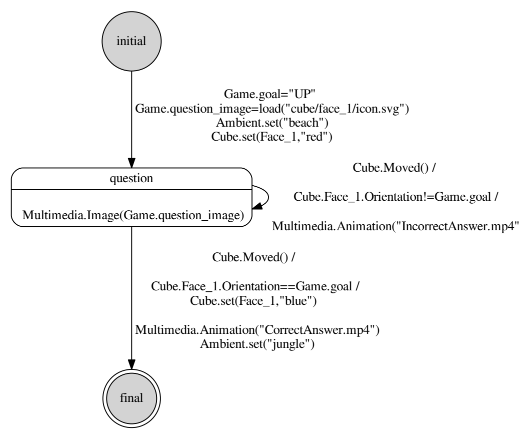

Figure 3: Example of Game Logic

2.1.2. Hierarchical and Weighted FSTs

The introduction of hierarchical FSTs, similar to complex states in UML state machines, enables the decomposition of game logic in scenes or other meaningful partitions. It also enables the reuse of the same

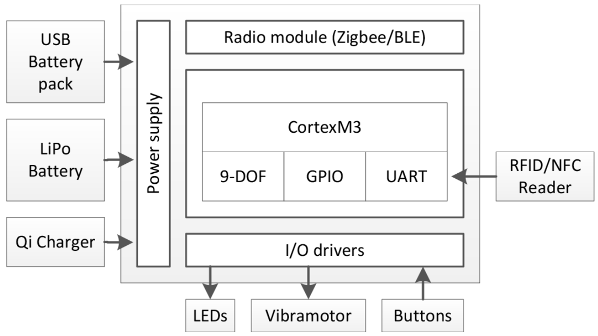

Figure 4: Smart objects hardware architecture

tools to compose a play experience out of a sequence (or other graph) of games. Additionally, it is possible to add weights to the transitions, which can be used to model scoring, both for feedback to the player and for later analysis.

2.2. Smart objects

Smart objects are simple objects and toys enriched with sensing and actuation capabilities. Objects need to be aware of their dynamic state (e.g. steady, moving, hit, orientation), possibly of their position in the environment — either absolute, e.g. coordinates within the room, or relative, e.g. close to another smart object — and shall provide actuation capabilities such as controlling LED lights, vibrating, or buzzing.

Being the sets of sensing/actuation capabilities of different smart objects very similar, a platform-based approach has been adopted. This approach led to the development of a core platform that can be easily extended with object-specific sensors (e.g. RFID/NFC readers) or actuators (e.g. vibramotor). The platform architecture, sketched in Figure 4 is based on the iNEMO-M1 Motion Sensing system-on-board [7] provided by STMicroelectronics.

This module integrates three inertial MEMS sensors — a triaxial accelerometer, a triaxial gyroscope and a triaxial magnetometer — and a CortexM3 core with 512KB of flash memory and 64KB of RAM, operating at a maximum frequency of 72MHz. The core provides several advanced mechanisms for power management, namely peripheral clock gating, frequency scaling and several low-power modes. Such features, combined with a suitable power management module tightly integrated with the RL-RTX microkernel [8], allow a dramatic optimization of the power consumption of the entire system-on-board.

Another key component of the architecture is the wireless radio transceiver. In the current prototype implementation the selected module is the XBEE Zigbee module produced by Digi International. Though more power-efficient and flexible solution exists for radio communication, this module has the advantage of its ease of use and the availability of pin-to-pin compatible modules offering alternative protocols such as DigiMesh, Bluetooth/BLE, and 802.11 bgn. This allows adapting the specific smart object to different environments with possible different electromagnetic constraints or noise levels. Furthermore, all these modules can be configured in different power modes, from the active mode (high consumption) to the deep-sleep mode with a very limited power consumption.

Finally, the smart object platform provides circuitry for power supply control (voltage regulators), battery charging and high-voltage and/or high-current I/O drivers. This allows to control several actuators (LEDs, LED strips, motors) through the microcontroller GPIOs.

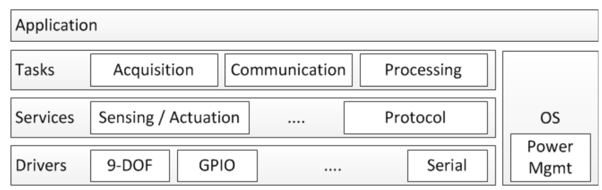

Figure 5: Smart objects software architecture

From the software point of view, a platform-based approach has also been adopted. In particular, a layered architecture provides different abstractions of the key functions of the system, namely: raw dynamic data sampled by the devices, semantic interpretation of dynamic data (e.g. state of movement, orientation), semantic-level communication protocol. Similarly, for actuation, a first abstraction layer allows controlling the individual devices while a more abstract layer provides high-level actuation based on semantic actions (e.g., a smile composed of blinking LEDs).

In addition to these functional abstractions, a module devoted to power-management has been developed. This module operates at two different levels of time granularity. At the finer level it implements a fast duty-cycling (typical period is in the order of few milliseconds) on the microcontroller operating modes and a slow duty-cycling (period of 1-5 seconds, depending on the configuration) on the radio module. These techniques allow reducing the overall power consumption from a nominal value (i.e. without power management) around 110mA to approximately 10mA. It is important noting that the current implementation of the power management does not exploit all the available features (e.g. DMA) that may enable further optimizations.

2.3. Smart Ambient

In the P3S scenario the goal of the Smart Ambient is to enhance the interaction with the users, extending the game logic to the physical space, integrating actuators (ambient lights, sound, lights carpet) and sensors (proximity and presence, lights color and level, air quality).

The Smart Space Gateway (Figure 6) is a software framework built upon some core functionalities and several add-ons, developed on an hardware platform based on popular communication protocols such as Wi-Fi, Bluetooth and ZigBee. It integrates and allows the control of the Smart Objects available in the environment. The whole system works in an *if-this-then-that* manner, based on commands, events and reactions. The core part of the software has to ensure some functionalities common to all add-ons such as a rule-engine for the event-based automations, a messaging bus for the events and commands-forwarding and virtualization of the real objects in the environment. On top of the aforementioned software, several services are realized through dedicated add-ons installed in a modular architecture. Add-ons can both provide northbound (i.e. http or vocal interaction) and southbound (i.e. communication to the physical layers) functionalities, thus allowing the design of services that can perform actions on the Smart Objects in the environment based on some information gathered from users living the space or internet.

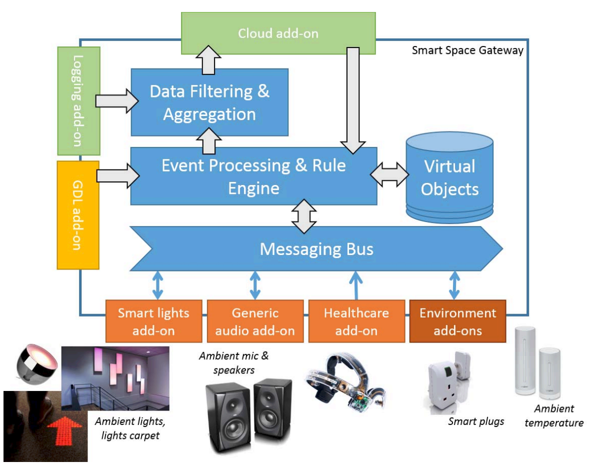

Figure 6: Smart Space Gateway Architecture

In the overall P3S architecture, the Smart Ambient represents the physical space where the users' activities are performed and is seen as a special Smart Object that can provide data and can be controlled according to the game logic. In the P3S installations the Smart Space Gateway connects and controls all the Smart Objects strictly related to the physical environment such as smart lights, ambient audio speakers, light wall panels, smart carpets, etc. It provides a GDL interface able to receive GDL messages describing how the ambient should be modified; these messages can address specific smart objects (e.g. to change the color of a given carpet tile) or be more complex requesting for a composed scenario. For example, the game logic can request to change the ambient to *jungle* scenario and the Smart Space Gateway acts on the Smart Ambient according to a predefined combination of lights, sounds, colors, etc.

As represented in Figure 6, the Smart Space Gateway is also the interface to the cloud components. It collects through dedicated APIs the data coming from the Smart Objects, even from those not directly involved in the game logic. The data are synchronized, aggregated and then sent to the cloud platforms in order to be available by therapists for monitoring. The messages received on this interface (JSON format over HTTP) have an envelope part including a set of common attributes and a variable payload whose format depends on specific data sent.

2.4. Interaction Model

The components of the P3S architecture interact through messages described using the Game Description Language (GDL). As stated in Section 2, the GDL specifications encode messages from the Game Logic to the devices (Smart Objects and Smart Space) and vice versa. To this end, every device needs to register to the GDL Message Dispatcher to advertise its presence in the system. Figure 7 shows an example of advertise message for the Smart Space. The message includes two items of information: *identity*, the URL to which messages should be sent to issue commands to the device itself (both for sensing and actuating); and *class*, the URL at which the description of the capabilities of the device can be retrieved.

The *class* describes the structure of the object (which can have several *parts*, or sub-objects associated with a position in a 3D space, as well as sensors and actuators). Additional properties can be exposed, e.g. a name and an icon, for integration in the Game Logic. These properties may have cumbersome representations — e.g., high-resolution pictures of a Smart Toy may be used to provide a faithful rendering of it in the virtual world. Thus, while for the Smart Space, the Smart Space Gateway fulfills both roles, for Smart Toys typically the capabilities are described in one or more files stored on a server. Once a Smart Object

```
REQUEST
POST http://p3s.elet.polimi.it/gdl
Content-Type: application/json
Body:
{
 "identity" : "http://192.168.1.43:8080/
     gdl/status/ambient",
 "class" : "http://192.168.1.43:8080/gdl
     /parts/ambient"
}
```

Figure 7: Advertise message sent to the dispatcher to register a device (in this case, the Smart Ambient)

is registered, it can receive messages from the Game Logic through the GDL Message Dispatcher.

Figure 8 shows an example of interaction between the main components of the P3S architecture. In the example, we assume a simple game logic where the Smart Space is set initially (in this case, the ambient "beach" is used), then the child is asked to put the Smart Object — a cube with die markings and internal lighting — in a specified position (in this case, with the face showing a single dot upwards). When the correct condition is reported, the game logic changes the colour of the cube face from red to blue.

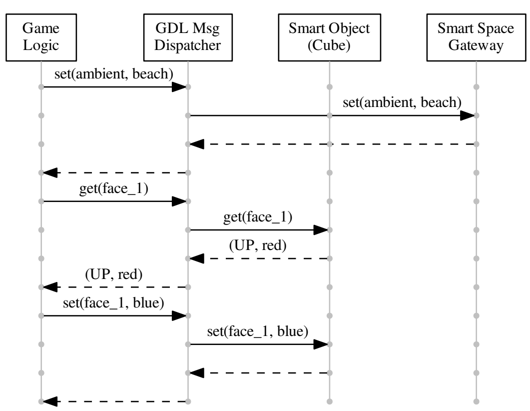

Figure 8: Example of interaction between the main components of the P3S architecture, assuming a single Smart Object

The GDL messages can be encoded in different formats, depending on the actual software implementation of the GDL Message Dispatcher and the protocol adapters that provide the interface to each object. Two main encodings are currently employed — an internal representation as C# data structures, typically used for communication between components residing within the same process, and an external representation as JSON messages used for HTTP-based communication

```
REQUEST
GET http://192.168.1.44:8080/gdl/status/
    face_1

RESPONSE
Content-Type: application/json
Body:
{
 "identity" : "http://192.168.1.44:8080/
     gdl/status/face_1",
 "colour": "red",
 "orientation":"UP"
}
```

(a) Status information request and response for the die object, specifically for the face showing 1 dot

```
REQUEST
POST http://192.168.1.43:8080/gdl/status/
    ambient
Content-Type: application/json
Body:
{
 "identity" : "http://192.168.1.43:8080/
     gdl/parts/ambient",
 "environment": "beach"
}
```

(b) Request for actuation of the "beach" environment

Figure 9: Example of GDL messages

between components residing on different processes or hosts. Figure 9a shows the HTTP request and response to perform the interrogation regarding the orientation of the cube face, whereas Figure 9b shows the HTTP request to set the environment.

3. Conclusions

Our research explores novel forms of smart spaces that can be used to support game-based interventions for children with intellectual disability. Our smart spaces support full body interaction with smart objects instrumented with audio, light, and motion sensors and actuators, virtual worlds on medium-large displays, and smart lights, and are integrated with cloud services for remote supervision and analysis of user behaviour.

This scenario involves a number of challenges. From a technological perspective, the developer must master the complexity of the communication among and behaviour of a mix of heterogeneous components and devices, and of their smooth orchestration in the context of each specific user activity.

This paper provides a contribution in this respect by presenting the general architecture and coordination infrastructure created in the Playful Supervised Smart Space project P3S, as well as initial designs for the

Smart Object and Smart Space Gateway Components. From an interaction design perspective, the challenge is to exploit this blend of technologies to offer a fluid, a usable and meaningful user experience, mastering the complexity of orchestrating the user actions upon, and the perceivable behaviour of, tangible elements (smart objects and smart lights) and virtual worlds that are conceived as a continuum space. In the domain of children with intellectual disability, UX design complexity is increased by the need to optimally calibrate the actions required by the various tasks, the stimuli provided, and their temporal distribution with respect to the special requirements of the children. During the remaining part of the P3S project, both type of challenges will be addressed, working along three main lines: 1) we will extend and refine the technological components of the P3S architecture, also to support the adaptation of P3S smart space and smart objects according to the evolution of the UX; 2) we will enrich the specification language GDL and its implementation, to better support the UX design process; 3) we will perform an intense on-the-field evaluation of current P3S smart spaces prototypes, in cooperation with the therapeutic centers involved in the project, with whom we will also design new UX to be implemented using the P3S platform.

4. Acknowledgements

This work was supported in part by EIT Digital Activity 15257 Playful Supervised Smart Spaces (P3S).

References

[1] M. Weiser, "The computer of the twenty-first century," *Scientific American*, vol. 265, no. 3, pp. 94–104, 1991.

[2] P. Dourish, *Where the action is: the foundations of embodied interaction*. MIT press, 2004.

[3] F. Garzotto, M. Gelsomini, L. Oliveto, and M. Valoriani, "Motion-based touchless interaction for asd children: A case study," in *Proc. 2014 Int'l Working Conf on Advanced Visual Interfaces*, ser. AVI '14. ACM, 2014, pp. 117–120.

[4] L. Bartoli, F. Garzotto, M. Gelsomini, L. Oliveto, and M. Valoriani, "Designing and evaluating touchless playful interaction for asd children," in *Proceedings of the 2014 Conference on Interaction Design and Children*, ser. IDC '14, 2014, pp. 17–26.

[5] N. Pares, P. Masri, G. van Wolferen, and C. Creed, "Achieving dialogue with children with severe autism in an adaptive multisensory interaction: the "mediate" project," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 11, no. 6, pp. 734–743, Nov 2005.

[6] J. E. Hopcroft and J. D. Ullman, *Introduction To Automata Theory, Languages, And Computation*, 1st ed. Addison-Wesley Longman Publishing Co., Inc., 1990.

[7] STMicroelectronics, "iNEMO-Inertial Modules," http://www.st.com/inemo, Apr 2015 (retrieved).

[8] ARM, "RTX Real-Time Operating System," http://www.keil.com/rl-arm/kernel.asp, Apr 2015 (retrieved).