IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 30 November 2022, accepted 10 January 2023, date of publication 6 February 2023, date of current version 13 February 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3242873

RESEARCH ARTICLE

Design and Evaluation of Protobject: A Tool for Rapid Prototyping of Interactive Products

ALESSIO BELLINO[1], GIORGIO DE MICHELIS[2], AND FLAVIO DE PAOLI[2]
[1]Facultad de Ingenieria y Ciencias, Escuela de Informatica y Telecomunicaciones, Universidad Diego Portales, Santiago 8370109, Chile
[2]Dipartimento di Informatica, Sistemistica e Comunicazione, Università di Milano–Bicocca, 20126 Milan, Italy

Corresponding author: Alessio Bellino (alessio.bellino@mail.udp.cl)

This work was supported in part by the Agencia Nacional de Investigación y Desarrollo (ANID), Chile, through the Startup Ciencia Program, under Grant SUC220048.

This work involved human subjects or animals in its research. The authors confirm that all human/animal subject research procedures and protocols are exempt from review board approval.

ABSTRACT Designing interactive prototypes involves multiple tools and skills. In addition, several design cycles are required to iterate through idea generation, evaluation of design alternatives, and development. Consequently, prototyping tools should offer flexibility and adaptability to allow designers to quickly test and evaluate different ideas, design alternatives, materials, interactions, etc. To meet these requirements, we designed Protobject – a rapid prototyping tool aimed at making the early stages of prototyping interactive products more flexible. Protobject allows designers to reinvent and reuse existing objects for prototyping purposes by making them interactive. After introducing the features of Protobject and discussing the differences with similar tools, we present a user evaluation through two workshop sessions held in Milan during Brera Design Days and attended by 22 people. The results suggest that Protobject facilitates cooperation between people with different skills by allowing them to envision interactive prototypes together.

INDEX TERMS Interactive systems, rapid prototyping, interaction design, physical product design, design tools, design cycle.

I. INTRODUCTION

Interactive products equipped with sensors and actuators (motors, servos, etc.) – commonly referred to as smart objects – are increasingly popular in the interconnected world in which we live. In addition to equipping smart objects with connectivity (techno-centric vision), designers should be concerned with discovering and inventing opportunities for interactions between objects and people (human-centric vision). To address this aspect, rapid prototyping plays a key role, as it allows designers to quickly evaluate ideas with users. Arduino and 3D printers are among the most common tools that have made rapid prototyping more accessible, affordable, and fast. However, Arduino and 3D printers better support the implementation stage rather than the early stages of design, where flexibility and adaptability are critical to accelerate prototyping.

The associate editor coordinating the review of this manuscript and approving it for publication was Eunil Park.

We argue that prototyping tools designed for the early design stage should be designed to provide flexibility and adaptability to support rapid prototyping [1], evolutionary change, and provide a solution that can eventually be engineered.

To meet these needs, we have developed Protobject, a rapid prototyping tool to support the early stages of design. First, we present and motivate Protobject. Then we discuss related work, the architecture of Protobject, and some usage scenarios. Finally, we present an evaluation performed in two workshop sessions held in Milan during Brera Design Days.

A. KEY FEATURES OF PROTOBJECT

Protobject [2] is a rapid prototyping tool that allows the observation of states and state changes of stationary objects (e.g., appliances, doors, lights, windows, curtains, umbrella stands in a house), which can be used as input events to trigger actions and model the behavior of interactive systems.

State changes can occur due to external events or user interaction with objects. For example, closing a door (interaction) changes the state of that door from ''open'' to ''closed''; taking an umbrella (interaction) from an umbrella stand changes the state of that umbrella from ''present'' to ''absent.'' An example that uses the change of state of an object to model the behavior of an interactive system might be the following: if John picks up the receiver to answer a phone call (state change event), then his Skype state can change to ''Don't disturb'' (triggered action).

The approach is consistent with the typical way of makers [3], which can teach Protobject the relevant states associated with objects (e.g., door ''open'' and ''closed''), how to detect state changes (e.g., from ''closed'' to ''open''), and what actions to perform when they occur (e.g., sounding an alarm).

Key features of Protobject are listed below.

1. Protobject is designed to sense objects without altering them or their surrounding environments; hence it is flexible and easily adaptable to changes.
2. Protobject supports visual training sessions to let designers visually define the regions of interest in the captured images and take snapshots of those regions to teach the system what states must be detected and what events are associated with them. At run time, Protobject can match the camera live stream against the stored images of states to detect the current state and trigger the associated events.
3. Protobject supports the detection of multi-value discrete states, either multi-value (e.g., closed-door/slightly-open-door/half-open-door/open-door), or binary (e.g., open/close, present/absent). Continuous state changes (e.g. the rotation degree of a knob) can be detected by adding physical markers on the observed objects (e.g., a multicolour band around the knob).
4. Protobject can use generic cameras to capture images of the state of objects. Since users frequently change smartphones, the use of older smartphones is encouraged since they are usually equipped with a camera and Wi-Fi connections and available at no cost. Any smartphone can serve the purpose since it can be easily turned into a Wi-Fi camera by simply installing a specific app: the current version of Protobject includes an app for Android smartphones.
5. Protobject supports both visual programming and code-based programming of interactive apps, i.e., where actions associated with state change events are defined. The former leverages a dedicated visual programming language, while a dedicated JavaScript framework supports the latter.
6. Protobject can be integrated with external platforms (e.g. Arduino, or IFTTT) to extend the sensing/actuation capabilities. Protobject is designed to detect visual states, but needs external tools to perform actions (e.g., send a message, activate the coffee machine, etc). By integrating tools such as Arduino, motors/actuators can be controlled, and physical properties that cannot be inferred from the visual properties of objects (e.g., internal states of appliances, temperature, etc.) can be sensed. By integrating tools such as IFTTT, moreover, a large amount of home automation devices can be detected/controlled.
7. Protobject has been conceived, designed, and implemented with a 'low tech' approach so that it can leverage widely available resources, such as smartphones and common objects, with basic technical skills.

Protobject is designed for developers, researchers, designers, and makers who aim to prototype novel products in the fields of home automation, Internet of Things, and tangible interfaces. In these fields, it is often necessary to detect interactions with objects and/or obtain their state.

B. MOTIVATING PROTOBJECT

According to Preece et al. [4], four main activities are carried out in the design process of interactive products:

1. establishing requirements for the user experience;
2. idea generation and designing alternatives that meet those requirements;
3. prototyping the selected alternative designs so that they can be deeply observed, communicated and assessed;
4. evaluating what is being built throughout the process and the user experience it offers.

These four activities are closely connected: ''*alternatives are evaluated through the prototypes and the results are fed back into the design or might identify missing requirements*.'' This iteration cycle is encouraged by Protobject, which in particular accelerates and supports prototyping (third activity) by allowing users to build design alternatives quickly and flexibly. Prototyping is supported by encouraging designers to reinvent the use of existing objects by seeing them from different perspectives. The result is the creation of ''things'' to put together to experiment with design alternatives. Moreover, Protobject allows designers to embed the ''*object of design*'' [5] in real-world contexts. This allows designers to have real-world experiences with objects, and to *reflect in practice* [6] on what they are designing. The opportunity to reflect while designing can also help generate innovative ideas. New insights, in fact, often emerge from practice [7]. Thus, the idea behind Protobject is to encourage the virtuous cycle of designing, prototyping and testing interactive products until they meet the desired requirements.

In the next section, we present related work that discusses Protobject in comparison with other rapid prototyping tools.

II. COMPARISON OF PROTOBJECT AND OTHER APPROACHES

The term ''rapid prototyping'', encompasses techniques for quickly fabricating physical products. Accordingly, we present an overview of relevant approaches and compare them with Protobject.

A. PROTOBJECT VS. DIGITAL FABRICATION & SENSOR-BASED DEVICES

Rapid prototyping, also known as "additive manufacturing" in the digital fabrication industry, has gained popularity with the advent of affordable 3D printers and user-friendly applications such as *Thingiverse* and *Autodesk Tinkercad*. Thingiverse[1] is a repository of shareable 3D-printable things that can be easily edited, combined and remixed thanks to Tinkercad.[2] Moreover, researchers have developed a variety of digital fabrication tools with the most diverse purposes, such as the following: *Printed Optics* [8] uses 3D printed optical elements to make a device interactive; *Instant Inkjet Circuits* [9] allows users to print any electronic circuit; *Lamello* [10] focuses on acoustic sensing for creating tangible input components that recognize user's interaction; *Acoustruments* [11] is designed to give tangible functionalities to smartphones by exploiting their audio system; *Sauron* [12] modifies hollow 3D models to enable an embedded camera to sense interactions on physical controls like buttons, sliders and so on.

Another relevant branch of rapid prototyping tools is based on devices and sensors. The most popular industrial tool is *Arduino*, which is frequently used together with 3D-printers. In fact, makers are used to make 3D printed parts interactive by incorporating sensors and actuators connected to Arduino (e.g., light sensors, servos, and so on). Researchers have also developed tools such as *Phidgets* [13] and *Net Gadgeteer* [14], which provide packaged hardware/software modules that can be quickly assembled and programmed. This family of tools offer several predefined sensors, modules and behaviours that fulfil various and even unforeseen prototyping needs.

Sensor-based tools and digital fabrication are closely interrelated, since prototyping with the former usually involves the use of standard components (sensors/actuators) and requires the manufacture of 3D objects to accommodate these components. The use of 3D printed components and dedicated sensors is deeply rooted in the community [15], [16], [17], [18], [19], especially in the later stages of prototyping, where users should be provided with near-final objects robust enough to be tested and gather feedback.

Nevertheless, in the initial stages of prototyping, sensor-based approaches and 3D-printed components may have some drawbacks. As a matter of fact, the initial stages of interactive product design are based on trial and error [3], [20] and require frequent changes and redesign. Therefore, it could be necessary to move, add or replace sensors, reconfigure the system, reinstall wires and sensors in different contexts or environments with significant effort and waste of time. Moreover, the creation of objects with digital fabrication is expensive and often slow with tools like 3D printers. On the other hand, the installation of sensors in existing objects is error prone, and changes are often irreversible. For example, drilling into a door frame to install a potentiometer that detects the door's opening angle is an irreversible operation, so it becomes problematic if the user misplaces where to drill.

Protobject mitigates these issues by providing a simpler and more flexible approach to quickly reconfigure the environment without concern for sensors, wires and hardware parts. In fact, Protobject promotes the use of common objects that are reinvented to meet specific prototyping needs – without requiring irreversible changes. In addition, Protobject allows multiple objects to be detected with a single smartphone camera, while using sensors may require one sensor per detected object. Finally, Protobject does not require any specific hardware/software expertise, which is instead required to set up sensors or design 3D CAD models.

B. PROTOBJECT VS. TOOLS FOR ADDING INTERACTIVITY TO EXISTING OBJECTS

Protobject belongs to the family of prototyping tools, along with few other examples like *Touch-and-Activate* [21] and *Eyepatch* [22], that have been designed to add interactivity to existing objects.

Touch-and-Activate adds touch-sensing capability to any rigid objects exploiting their resonant properties, which change according the user's interactions (e.g. diverse ways to grasp). Resonances can be detected using a sensor, i.e., a pair of vibration speakers and a piezo-electric microphone attached to the target object. This approach requires to physically attach speakers and microphones to augment physical objects and sensing their resonance. The disadvantages of this solution are related to the exploitation of the resonance properties of the objects, which cannot be too large, and the fixing of the sensors, which may not always be possible. Consequently, some properties, such as the visual ones, cannot be sensed to retrieve the state of an object (e.g., light on/off).

Eyepatch is more similar to Protobject, since it also has been designed for sensing interactions and objects by cameras. Eyepatch allows programmers to extract interactions from live video and to stream those interactions to other rapid prototyping tools to simplify the development of vision-based prototypes. Eyepatch requires a training phase to teach the tool how to classify, hence recognise, the target objects and situations (e.g., the detection of walking people). Different kinds of classifiers are supported, including the ones to detect shapes, colours, motions, etc.

Although Eyepatch can address a variety of situations, it has not been designed to detect state and state change of objects, therefore the association of behaviour with the physical position of an object is not supported. Another difference between Protobject and Eyepatch is that the latter cannot sense different regions of a video stream simultaneously as Protobject does.

C. PROTOBJECT VS. WIZARD-OF-OZ APPROACHES

The Wizard-of-Oz [23] method is "*based on the observation of the user when operating an apparently fully functional*

[1] https://thingiverse.com
[2] https://tinkercad.com

system whose missing features are supplemented by a hidden human wizard.'' Therefore, hidden humans replace the use of sensors for the rapid prototyping of interactive systems.

From an HCI perspective, the approach of Protobject to rapid prototyping is much similar to the one of the Wizard of Oz. In fact, Protobject works somehow like a ''hidden human'', i.e., observing and sensing the environment. From the design practice perspective, Wizard-of-Oz tools also let designers use, reuse and reinvent existing objects without any modifications.

According to a survey, most of the Wizard-of-Oz tools have been developed for prototyping and testing specific scenarios or systems [24]. On the other hand, just few tools (e.g., [25]) are general purpose and customizable, while the flexibility and configurability of Protobject make it natively general-purpose.

D. PROTOBJECT VS. PAPER PROTOTYPING

In paper prototyping [1], [26], system components are represented by paper artefacts and their behaviour is simulated by people. Despite their primitiveness, the approaches to prototyping on paper have distinctive characteristics that make them interesting for the simplicity and versatility. They are generally used for the initial phase of prototyping to deeply involve both designers and users in the simulation of the behaviour of the future system, so that while they are performing it, they have the chance to reflect on what they are doing and reconsider it until the definition of a stable set of requirements.

Paper prototyping is used to animate objects in a very creative way, but paper objects are not interactive. In contrast, the real interactivity allowed by Protobject lets designers work in a more structured way by identifying the significant states – and how to combine them – to obtain the desired behaviour. The identification of the relevant states allows a first definition of a state machine that defines the algorithm that will govern the desired interactive behaviour. Therefore, compared to paper prototyping, our approach encourages designers to build more realistic prototypes that are closer to what can be achieved in an engineered final system.

III. PROTOBJECT DESIGN

Protobject is composed of two main complementary apps: a smartphone app that turns a smartphone into a Wi-Fi camera; and a desktop app that allows the designer to develop the prototype. The latter supports (i) the acquisition of images from the smartphone camera; (ii) the definition of object states; (iii) the detection of events (i.e., state changes); and (iv) the broadcasting of events through an integrated Web Socket server connected to the prototype under development.

A. THE SMARTPHONE APP

The smartphone app turns an Android smartphone into a Wi-Fi camera that sends a video stream in M-JPEG format to the desktop app through the IP address displayed on the screen (Figure 1). Protobject can work with any cameras.

FIGURE 1. A smartphone turned into a Wi-Fi camera that displays the IP address to connect.

Anyway, the resolution has been limited to 1280 × 960 to reduce the bandwidth and ensure fluency of images while guaranteeing a satisfactory quality.

B. THE DESKTOP APP

Protobject desktop app (Figure 2) is written in NW.js, and run on Windows, Linux and Mac OS.

1) USER INTERFACE

Protobject desktop app works in design and detection mode. The design mode lets users define and record the possible states of involved objects to build a database of states. The detection mode lets Protobject detect the current state of an object and its state changes.

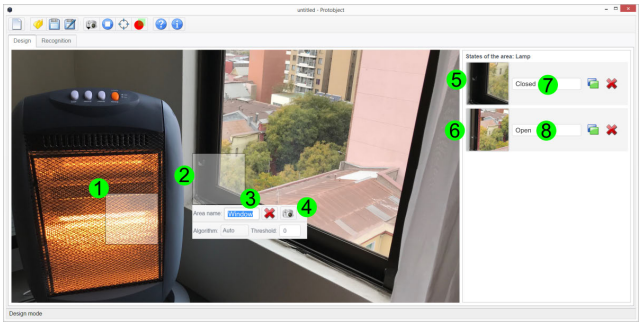

FIGURE 2. A screenshot in design mode. Heater (1) and window (2) are regions of interest. The window is selected, and the associated states ("Closed" and "Open") are displayed on the right panel.

a: DESIGN MODE

After connecting and positioning the smartphone camera, users can (a) visually create regions by semi-transparent rectangles around the target objects (e.g., heater and window in Figure 2, label 1 and 2); (b) assign a name to each region in the associated panel (e.g., ''Window'', label 3); (c) and record the states to be detected by taking snapshots of objects in the desired states by the camera-button (label 4). Pictures of the captured states are reported on the right (e.g., window closed and open, labels 5 and 6), and a name is given to each of them by typing a string in the field next to pictures (e.g. ''Closed'' and ''Open'', label 7 and 8).

When a new object is created and states are defined, they can be saved in a JSON file, which records name, position and size of regions along with pictures of associated states. Saved objects can be loaded, modified and stored again to modify the prototype under development.

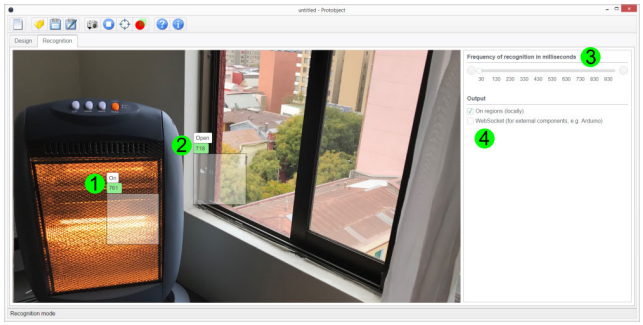

FIGURE 3. Snapshot of heater and window displaying their current state ("On" and "Open" respectively) in recognition mode.

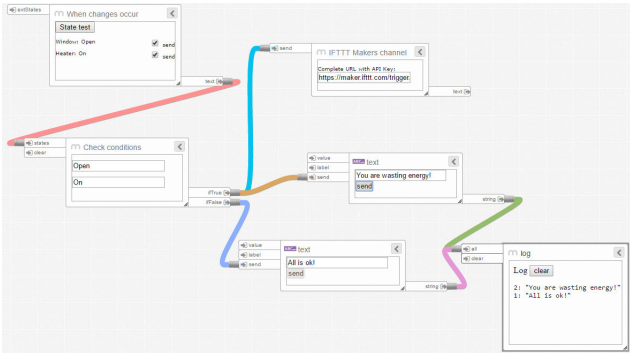

FIGURE 4. An example of visual design of interactive applications.

b: DETECTION MODE

After enabling the detection mode, Protobject starts monitoring the selected scene, and for each region the real-time state is displayed by showing the name associated with it (e.g., label 1 and 2 in Figure 3), and a similarity index (the number displayed with green background) that measures the similarity between the current image and the recorded images. The displayed value is computed using a linear combination of Pearson correlation among RGB components [27] and cosine similarity among HoG descriptors [28] of the images.

States are refreshed with a frequency that can be visually selected by a sliding cursor (label 3 in Figure 3). Protobject can send event messages with the detected states to trigger actions in external components by WebSocket channels that can be enabled on the control panel (label 4 in Figure 3).

C. PROGRAMMING THE INTERACTIVE APPLICATIONS

Protobject prototypes are designed to react to state changes detected by cameras. Any visual or code programming language can be used to define the behavior of prototypes. Protobject relies on the Meemoo visual data-flow programming language [29], [30] that lets non-tech-savvy users define their interactive behaviour by visually instantiating modules and connecting their input/output ports. A library of modules providing actions that can be connected to the Protobject app has been developed (Figure 4). So, for example, it is possible to state that ''if the heater turns on and the window is open, then an alert is displayed to the user.''

More sophisticated tasks can be written in JavaScript directly in the browser. In fact, a JavaScript library for the browser has been developed to simplify the use of Protobject: the library automatically handles events and communication via WebSocket, so users can focus solely on prototype programming. Finally, JavaScript is widely used by designers (especially front-end web designers).

D. INTEGRATION WITH THIRD-PARTY PLATFORMS

Considering that an interactive product generally consists of sensors (capable of sensing the environment) and actuators (capable of generating a change in the environment), we wish to highlight the relevance of integrating external platforms, especially for actuator control. In fact, Protobject works as a multipurpose (visual) sensor and has no actuation capabilities per se. To meet such as capabilities, then, Protobject has been designed as a tool that can be integrated to external platforms such as Arduino and IFTTT. Arduino can be easily controlled via REST API,[3] and then via AJAX (i.e. JavaScript) calls made directly from the browser, and so it integrates rather naturally with Protobject since it can also be programmed with JavaScript in the browser. However, using Arduino requires some programming skills, and is therefore suitable for more advanced users. As an alternative there is also IFTTT, which is suitable for novice users since it provides a simple web interface through which users can manage automations of hundreds of devices and services[4] through Webhooks integration.[5] IFTTT allows users to control countless devices, but they must be available (which is not always the case). For these reasons, in our workshop we only used IFTTT to send simple notifications, assuming everyone had a smartphone capable of receiving them. As matter of fact, both Meemoo and JavaScript library for Protobject supports IFTTT which makes its use almost immediate.

Finally, we consider it worth noting that even the browser itself can be used as an actuator in some way. In fact, a browser can be used to display text, emit sounds, mimic the behavior of an intelligent object (such as an RGB lamp, see Figure 10 below), and much more. However, in this case, the knowledge of JavaScript is essential to be able to mimic all the desired behaviors.

IV. USER EVALUATION

To evaluate Protobject, we designed a user study that is exploratory in nature and aims to answer the following research questions:

- To what extent is Protobject and its approach to prototyping user friendly? Are there different perceptions, in terms of ease of use, between tech-savvy and non-tech-savvy users?

[3]https://github.com/marcoschwartz/aREST
[4]https://ifttt.com/services
[5]https://ifttt.com/maker_webhooks/details

- What is the perceived usefulness of Protobject? How and to what extent can Protobject support the prototyping process and cooperation between developers and designers?

To answer these questions, we designed a workshop in which users with different skills (technologists and designers) build prototypes together. The workshop was carried out in two sessions and were organized in Milan during the Brera Design Days (Figure 5), an international event focused on talks, workshops, and exhibitions on design.

FIGURE 5. Brera Design Days workshop.

A. WORKSHOP ORGANIZATION

1) HOSTING ENVIRONMENT

The workshop sessions were held in a room set up with four tables and the objects necessary for the design of different prototypes (Figure 6).

2) PARTICIPANT RECRUITMENT

Participants were recruited on-line by the organizers of Brera Design Days through social networks (i.e., Facebook and Twitter) and the website of the event. The workshop was also advertised on Makerspaces, coworking spaces, and Facebook groups focusing on design and technology.

3) WORK TEAMS

The first session of the workshop was attended by 14 people, who were divided into 4 teams: two of them consisted of 4 people, and the remaining two consisted of 3 people. The second session was attended by 8 people, who were divided into 3 teams: two of them consisted of 3 people while the remaining one consisted of 2 people. The teams were composed mainly of designers; each one including a person with some soft programming skills, except for the two-person team whose participants declared no programming skills.

FIGURE 6. The workshop room.

4) FACILITATORS

We provided each session with facilitators to assist and help participants during the workshop. The role of the facilitators was to explain the Protobject approach when requested, but without participating in the development of the prototypes. In the first session of the workshop, a programmer and two co-authors of this paper took the role of facilitators. The programmer did not participate in the second session of the workshop. The principal designer of Protobject drove the workshop, monitored the participants' behaviour and collected feedbacks.

5) PROPOSED PROTOTYPES

During the workshop, participants were asked to build several prototypes: (1) a smart app to control home phones and Skype, (2) a smart umbrella stand, (3) an energy saving system, and (4) a tangible interface for controlling a light.

Prototypes have been implemented exploiting both Meemoo and JavaScript as programming environments to evaluate their ease of use. Supplementary materials presents Meemoo schemes and JavaScript code for the four prototypes while the details of how the prototypes work are discussed below.

a: SMART APP TO CONTROL HOME PHONE AND SKYPE

Phone and Skype calls may occur at the same time, so we implemented a *do-not-disturb* prototype that avoids simultaneous calls.

In this scenario, we turned an ordinary desk phone into a ''smart'' phone that controls Skype status by switching it between ''do not disturb,'' when in use, and ''available,'' when idle. To detect the state of the phone, Protobject can use a region around the physical phone to check if the handset is in place (associated with the ''idle'' state to activate the

"available" status on Skype), or not (associated with the "in-use" state to activate the "do not disturb" status on Skype).

b: SMART UMBRELLA STAND

It may happen that a person leaves the house without an umbrella when it is raining. Therefore, the user has to go back to catch the umbrella. To prevent this annoying situation, a system that sends a warning message to that person immediately after leaving the house has been developed. Specifically, we turned an ordinary umbrella stand into an intelligent one that can send a message (e.g., using IFTTT and Telegram) when it rains and the user walks out the house door without an umbrella.

In this scenario, three physical regions monitored by Protobject (see Figure 7) were defined: a blue region to sense if the door is "open" or "closed"; a green region to sense the presence of someone in front of the door; and a red region to check umbrella presence in its stand (in Figure 7, the umbrella stand consisted of a kitchen roll).

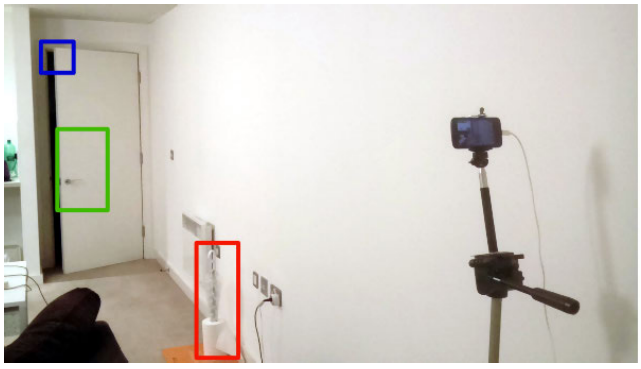

FIGURE 7. Smart umbrella stand scenario.

The smart umbrella stand works as follows. The prototype starts monitoring the door waiting to detect the closing event. When the door closes, the application sends the notification reminding the umbrella when the following three conditions occur simultaneously: (1) the user is not detected inside the home; (2) the umbrella is present in the umbrella stand; and (3) it is raining (this is detected through an external API, for example using Weather.com)

c: ENERGY SAVING SYSTEM

Nowadays, energy wastage should be avoided more than ever, and technology can help in this regard, for example by preventing the heating system from working with an open window. To this end, we envisioned a prototype that sends an alert message to the homeowner when this situation occurs.

In this case, Protobject can exploit two smartphone cameras (Figure 8) to crosscheck two areas: the window (green area), and the heater control panel (blue area). The prototype monitors window and heating status waiting to detect the event of opening and turning on respectively. When one of these two events occurs, the prototype cross-checks the window and the heater. In case the window is open and, at the same time, the heater is on, a notification is sent to the user (e.g., using IFTTT and Telegram) inviting him/her to close the window.

FIGURE 8. Energy saving scenario.

d: TANGIBLE INTERFACES FOR CONTROLLING A LAMP

In this context, the challenge is to develop techniques to capture interactions with tangible components without any electronic component. We experimented three kinds of components – buttons and two kinds of potentiometers (i.e., knob and linear) – that have been built by exploiting common and widespread objects: paperclips, pens, bottle caps, drawing pins and rubber bands put in a corkboard (see Figure 9).

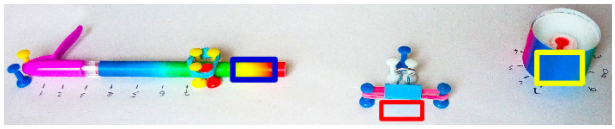

FIGURE 9. Examples of tangible interfaces: a button (in the middle), and two potentiometers (linear and knob).

Buttons are designed using a rubber band and paperclips fixed with drawing pins (the middle object in Figure 9: red region). Protobject recognizes the state (pushed or not) by monitoring the presence or absence of the clip in the detection region.

The linear potentiometer consists of the body of a pen that moves through four drawing pins inserted into the cork panel and held together by an elastic band (the blue region on the left in Figure 9). The knob potentiometer is made with a bottle cap fixed to the cork panel with a drawing pin. The latter works also as a rotation pivot (the yellow region on the right in Figure 9). Coloured strips let Protobject recognize the state of potentiometers: a colour strip that goes from red to cyan was put in the body of the pen, and a strip that goes from red to red (to support continuous rotation) wraps the bottle cap.

The interface objects in Figure 9 can be used to control a virtual lamp as the one displayed in Figure 10. The button

turns on or off the lamp, the linear potentiometer regulates its intensity whereas the knob potentiometer is used to change its colour. Protobject senses the interactions through the blue, red and yellow regions in Figure 9, and triggers the appropriate actions. This example shows how even a browser can be used as an actuator, reproducing the operation of an RGB lamp.

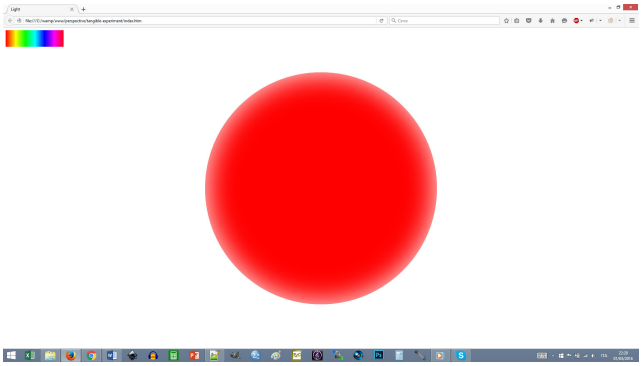

FIGURE 10. The virtual lamp displayed on the browser.

6) MATERIALS

Participants were asked to bring their laptops to participate in the workshop. The smartphones and tripods to hold them were provided by the organization. Participants, however, could use their own smartphones if they wanted to. Each workshop teams has been provided with a table and pre-arranged materials and components to let them build the prototypes: a phone handset was placed on each table for developing the first prototype (Figure 11); a panel simulating a door along with an umbrella and the related stand for developing the second prototype (Figure 12); a fan heater with a small-scale model of a window for developing the third prototype (Figure 13); and corkboards, caps and coloured paper strips for designing the tangible interface for controlling a light (Figure 14).

FIGURE 11. Home phone connected with Skype.

7) PROCEDURE

At the beginning of the workshop, a short introductory talk explained the Protobject aims, basic operating instructions, and the workshop structure that consisted of the construction of 4 prototypes. In details, the design of the first prototype (phone connected with Skype) was entirely guided:

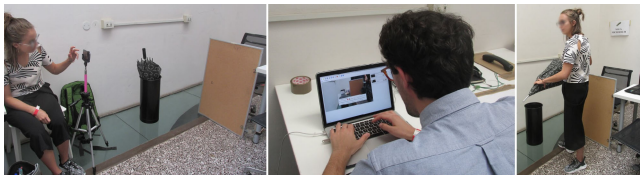

FIGURE 12. Smart umbrella scenario: the umbrella stand, and the fake door.

FIGURE 13. Energy saving tool: the heater and the fake window of cardboard.

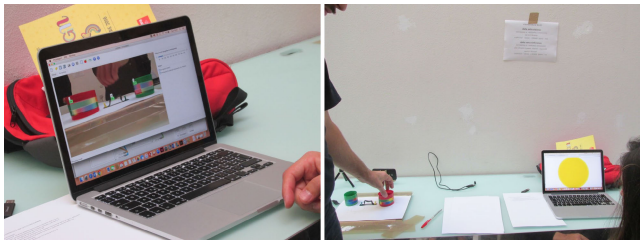

FIGURE 14. Tangible interface for controlling a light: two caps used as potentiometer and a strip of paper used as button.

it was explained how to capture the states of the handset (picked/hanged-up), and how to design the interactive behaviour using the Meemoo visual programming, and then JavaScript coding in the browser. After developing the first prototype, each group was asked to develop the other three on their own. Each group completed at least three prototypes (including the guided one). At least one prototype, among the unguided ones, has been programmed using both Meemo and Javascript and the browser. The duration of the workshop was about 3 hours.

After the workshop, participants were asked to complete an anonymous questionnaire in which we asked for personal information, and to answer four closed-ended questions and two open-ended questions. The workshops were part of an event open to the public, and participants were informed that by participating in the activity, they would agree that the data collected would be used for research purposes. In addition, before completing the questionnaire, participants were reminded that the data would be used for research purposes.

In the personal information section, we collected data on age (numeric field), academic background (free text area), job (free text area), familiarity with technology (6-point scale), programming skills (multiple selection), smartphone renewal frequency (multiple selection), old smartphone ownership (yes/no).

In the closed-ended question section, we collected data using a 6-point ordinal scale on the ease of leveraging states for prototyping, ease of use of the Protobject UI, ease of programming with Meemoo, and ease of programming with JavaScript + browser.

In the open-ended questions section, we asked participants to answer the following questions:

- How would you describe today's experience? What did you learn? Was it useful or useless? Do you think Protobject can change the way you design interactive systems? Please try to be as detailed as possible.
- What would you improve about Protobject? The interface? The way it is used? If you have ideas write them down below trying to be as detailed as possible.

B. RESULTS

This section discusses the outcome of the questionnaire, which was filled in by 18 out of 22 participants, and the feedbacks collected during the workshop.

1) ANALYSIS OF QUESTIONNAIRE RESPONSES

Participants were aged between 21 and 57 years old (M: 33, SD: 10.2). Moreover, their profiles were quite varied. With respect to the academic background, 9 out 18 participants studied design (from architecture, to product and industrial design), 4 studied computer science or one of its sub-areas, the rest had different backgrounds (e.g., sociology and technology, or design and engineering).

Regarding their jobs, 7 out of 18 work as designers, four work in the technological field, two were still students, the rest work in other fields.

Regarding the proneness to buy a new version of smartphones, 6 out of 18 declared to replace smartphones only when broken, 1 out 18 declared to renew smartphone once a year, 4 out of 18 renew smartphone every two years and the rest of the participant every three years. Moreover, 13 out of 18 participants owned old smartphones.

All participants rated at least 3 on 6 (the highest score, indicating "the ability to understand all technological gadgets") their relationship with technology.

Regarding programming skills, 9 participants stated that they can develop simple scripts/programs, 2 medium complexity software, 1 high complexity software, 2 are able to understand logic and processes without being able to program, and 3 have no idea about programming, while 1 participant did not answer the question.

a: QUANTITATIVE ANALYSIS

A quantitative analysis was done on the closed-ended questions. Particularly, binomial tests were carried out to detect positive or negative tendencies on the responses of the questionnaire. A 6-point ordinal scale was chosen because of its balanced condition between negative tendencies (1, 2 and 3) and positive tendencies (4, 5 and 6). Significant positive tendencies were detected for 3 out of 4 items taken into account: use of states for prototyping (proportion 0.00 vs. 1.00, $p < 0.001$), ease of use of Protobject UI (0.22 vs. 0.78, $p = 0.031$), and ease of programming with Meemoo (0.22 vs. 0.78, $p = 0.031$). Regarding the ease of programming with JavaScript + browser, no significance tendency was found (0.39 vs. 0.61, $p = 0.481$). These results are summarized in Figure 15 through diverging stacked bars.

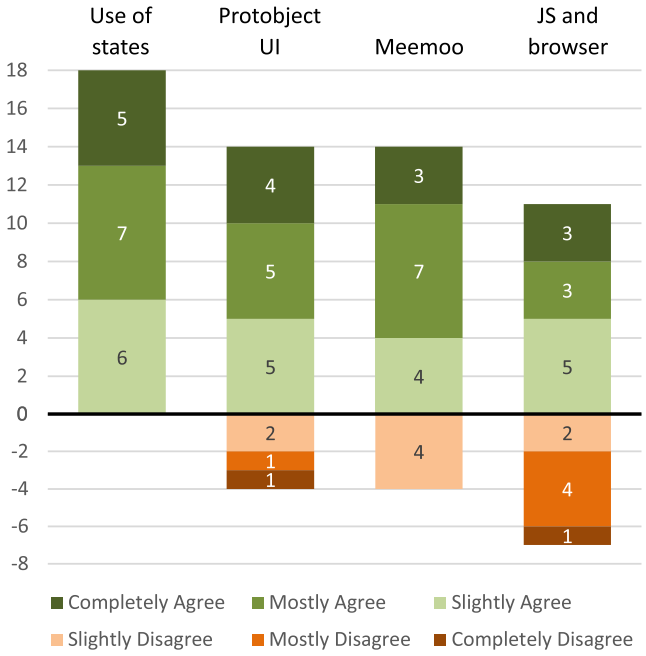

FIGURE 15. Diverging stacked bar showing the distribution of responses considering different evaluation items with positive and negative tendencies.

Further quantitative analysis was done on the closed-ended questions to see if users with different profiles (i.e., different programming abilities, relationships with technology, academic backgrounds, and jobs) gave different ratings performing the Kruskal-Wallis test. The only item where significant differences were found is the ease of programming with JavaScript + browser.

In particular, the use of JavaScript + browser was rated easier by participants with (a) a better relationship with technology ($p = 0.016$; mean rank: 5.50 for 3 value, 4.25 for 4 value, 14.33 for 5 value and 9.50 for 6 value), (b) better technical backgrounds ($p = 0.014$; mean rank: 6.83 for design background, 7.63 for mixed background and 15.26 for technical background), and (c) more technical jobs ($p = 0.013$; mean rank: 5.36 for designers, 8.00 for others, 11.25 for students and 15.25 for technical jobs). No differences were found according to different programming abilities ($p = 0.072$).

b: QUALITATIVE ANALYSIS

This section presents the comments that emerged from the open-ended responses grouped into different themes.

Usefulness/interest of the experience: Many participants perceived the workshop experience useful and/or

interesting (P1, P4, P6, P7, P8, P9, P10, P11, P12, P15, P16 P17 and P18).

Clarity of the initial presentation: P4 and P10 stated that the initial presentation did not clearly explained Protobject's objectives and value. P10 and P8 initially understood that Protobject could be used for final systems instead of just prototyping. Moreover, P10 would have preferred clearer examples of Protobject usage as part of the design process of an interactive system, and discussion on relations with final products built with Arduino and 3D Printers.

User interface of Protobject: According to P4 and P8, the user experience could be improved. P8 added that a user guide could have been enough to learn Protobject. According to P7, the user interface should be improved to let people with different backgrounds better understand it. P10 stated that the icons of Protobject were not so intuitive. P11 stated that the UI of Protobject was not so intuitive especially for people who are not familiar with programming. Anyhow, P9 and P12 stated that Protobject is easy to use and intuitive for unskilled users.

Meemoo editor: P2 stated that Meemoo has no affordance in many important points and additional help would be needed. P12 also expressed somehow the same ideas claiming that Meemoo should be supported by tooltips for helping users.

Towards real usage: P1 expressed the intention to use it for lighting projects whereas P11 for interior design projects. P12 also declared the intention to use Protobject for interaction design projects.

Thinking about applications: P2, P4, and P6 stated that they were considering/thinking what kind of applications, or in which situations, Protobject could be applied to – suggesting that this was not entirely clear.

Simplified understanding of interactive systems programming: After the workshop experience, P4 said he was less worried about prototyping interactive systems; P5 had a clearer idea regarding programming; and P17 learned how to develop simple interactive behaviours. P16 expressed a similar evaluation claiming that Protobject could help designers since it simplifies their understanding of interactive systems since technological skills are not required.

Meta-design promoting collaboration between designers and programmers: P7 stated that Protobject could be used for meta-design[6] processes by making prototypes understandable. P16 expressed a somewhat complementary idea, pointing out that Protobject can create a bridge between designers and programmers letting them share common practices for co-design.

Interfacing with JavaScript: P8 said that good integration with JavaScript saves programming time and allows programmers to focus on design.

App for iPhone: P15 pointed out that the app for iPhone is missing limiting the use of Protobject.

[6]Meta-design is a preliminary process aimed to define the initial stages of a design project.

Training issues: P14 pointed out that changing the position of cameras can become an issue that may require the re-execution of the training phase.

2) OBSERVATION

During the workshop, we observed different behaviours of the participants. First, we noticed that the designers often needed the help of programmers and/or facilitators to identify the states that define the interactive behaviour of a system.

Second, we realized that participants tended to capture the entire image of the objects (e.g. the whole phone), rather than just the significant part for state changes (e.g., the part of the phone where the handset is placed).

Finally, the most tech-savvy participants and some designers showed to be really engaged when designing with Protobject. In the first session of the workshop, we observed that the three programmers continued to explore the functionality of Protobject beyond the three hours of the workshop, and that two designers stayed with them to learn more about the potential of the tool. In the second session of the workshop, the team of two designers stayed longer to ask for explanations about the different features of Protobject.

3) FEEDBACKS RECEIVED DURING THE WORKSHOP

Most of the feedbacks received during the workshop were reported by the participants in the questionnaire. Therefore, there is not much to add here. The only additional feedback we report was given by a product designer. She suggested that Protobject could be further developed in two directions to address maker-oriented needs, and designer-oriented needs. In fact, makers can be attracted by the rapidity of prototyping interactive behaviours, and designers can leverage Protobject to design interactive systems faster by reducing errors. At this regard, she told us what she had to deal with when designing a router. In that case, the customer granted the designers little time (one week) to design a router and its interactive behaviour. According to the designer, the time was not enough to develop a true interactive product and the interaction could only be sketched. When interactivity was added to the product (at a later stage), it was not usable. The designer said that if they could use Protobject, they would test the interactivity with the customer at an early stage, thus avoiding a failure.

V. DISCUSSION

The approach to Protobject prototyping was generally well received by the participants at the workshop. Most of them considered the tool useful, interesting, and claimed to have learned about programming interactive objects. Tech-savvy participants appreciated its simplicity when developing interactive systems, and most participants have increased their awareness of how interactive systems work. However, the results show that there are aspects that can be improved. Below, we answer the research questions presented in Section IV in detail.

A. TO WHAT EXTENT IS PROTOBJECT AND ITS APPROACH TO PROTOTYPING EASY TO USE? ARE THERE DIFFERENT PERCEPTIONS, IN TERMS OF EASE OF USE, BETWEEN TECH-SAVVY AND NON-TECH-SAVVY USERS?

Ratings of the closed-ended questions suggest that there are significantly positive trends regarding the ease of use of the Protobject UI. The qualitative analysis, however, shows that some users considered that the user interface could have been better to accommodate users with different backgrounds, and in terms of icon clarity. This, on further reflection, is not surprising considering that many of the participants were experienced designers and it is therefore plausible that they evaluate aesthetic and design aspects rather critically.

Significantly positive trends were also identified in the ease of use of states for prototyping, however a participant pointed out that changing the camera position requires a new training step, which is not convenient. Moreover, we observed that the way in which participants define the regions of the image to capture states is often inadequate. In fact participants tend to define a region on the entire object rather than just on the part where the state change occurs. Capturing the entire image of an object (e.g., a door) can lead to potential misbehaviours due to the high probability of interfering actions (e.g., people moving in the room, or objects occasionally placed in front of the door). Instead, capturing the smaller region useful for observing the state of the object mitigates errors (e.g., the blue region of the door in Figure 7 is difficult to obstruct). Accordingly, more effort should be put into explaining why and how to select relevant regions in order to capture states with as little interference as possible. Additionally, observation suggests that designers often needed programmers helps to identify correctly the states needed to make a prototype. This is probably because of their different way of thinking: programmers find it easier to think in terms of states, while designers find it easier to think in terms of visual attributes. However, we believe that this is the reason why Protobject facilitates the cooperation between designers and developers making state identification process clearer. In fact, Protobject uses visual sensing – easily understood by designers – from which states – easily understood by programmers – are derived.

Significantly positive trends were also identified in the ease of use of the Meemoo visual programming language although 2 users suggested that Meemoo should include additional help such as tooltips. Regarding JavaScript + browser there were no significantly positive trends although one participant (programmer) pointed out that integration with JavaScript is good, and this lets users focus on system design.

In addition, while Meemoo was evaluated easy to use by most of the participants regardless of their academic backgrounds, jobs or relationships with technology, the same is not true for JavaScript. In fact, JavaScript was rated easy to use by tech-savvy users, but not by the rest of the users. This were somehow expected since programming by code requires skills that may not be common among non-tech-savvy users.

Regarding the other closed-ended questions, however, the ratings show no significant differences with respect to different participants' backgrounds. Thus, it can be stated that users perceive both the interface and the use of the prototyping states as easy to use regardless of their (technical or not) background.

Finally, in terms of practical usability, the results of the questionnaire indicate that the majority of participants own old, unused smartphones, and thus could develop prototypes with Protobject without having to use the smartphone they normally use for everyday tasks. One user, however, pointed out that the iPhone application is missing – even if it could be easily developed.

B. WHAT IS THE PERCEIVED USEFULNESS OF PROTOBJECT? HOW AND TO WHAT EXTENT CAN PROTOBJECT SUPPORT THE PROTOTYPING PROCESS AND COOPERATION BETWEEN DEVELOPERS AND DESIGNERS?

Overall, 13 participants perceived the workshop experience as useful and/or interesting. In addition, the participants gave several insights that give us a better understanding of the usefulness of Protobject, and how it can be used to support the prototyping process. 4 participants stated that Protobject is useful for simplifying the understanding of interactive systems programming, due to the simplicity with which the programming itself is approached. Therefore, from the point of view of prototyping process, it can be said that Protobject makes it easy for non-tech-savvy users to understand interactivity. This makes us realize that Protobject could be a useful tool to introduce less experienced users to programming. This aspect was further confirmed by the fact that Protobject was considered useful for facilitating cooperation between users with different backgrounds. In fact, one user considers that Protobject can effectively support the meta-design process in order to make prototypes understandable. Another user stated explicitly that Protobject is able to create a bridge between designers and programmers and have them share common practices to design together. This aspect was also noted several times during the workshop. In fact, it has been observed that the visual perception of the state of objects – supported by Protobject – makes it easier for less experienced users to understand the interactive system and its programming, and this facilitates cooperation. Concerning usefulness in real-world situations, 3 participants would find Protobject useful for lighting, interior design and interaction design projects respectively. However, there were also users who, on the contrary, wondered what projects Protobject could be used for, suggesting that it is not easy to imagine how it could be used in real situations.

Another aspect related to supporting the prototyping process concerns the perceived clarity of the Protobject approach itself. In fact, three users did not understand the purpose and value of Protobject from the initial presentation. In fact, it was necessary to engage in the hands-on experience of the workshop to understand these aspects. This would not seem

to be a very relevant weakness. However, this aspect should not be underestimated since the lack of clarity of objectives and values may limit and/or confuse users with respect to the actual support that Protobject can give to the prototyping process. Therefore, we believe that properly communicating the objectives and value of Protobject in order to support the prototyping process is crucial, and needs to be improved.

VI. PROTOBJECT APPROACH LIMITATIONS

The workshop also allowed us to reflect on the limitations of Protobject in general terms. A major limit that emerged dealt with the management of states that are not visible. Even if there are solutions in some cases (e.g., by making states visible with coloured stripes), in other cases there is no way to detect the internal states of devices. A possible solution is the adoption of different approaches, as the one proposed by Mackay et al. [1] to make the invisible state visible by using paper signs: in this way we lose the automatic behaviour of the prototype, but we gain in simplicity and completeness. Another solution for this purpose could be the use of Arduino, although this requires reverse engineering skills to be able to sense the internal states of the devices, so it does not seem to us to be an easily viable option.

Another limitation of Protobject is that the design of reactive prototypes – which trigger an action when a specific state is reached – is limited due to the lack of active components (smart appliances, actuators, LEDs, relays). Although many smart objects – even IFTTT-compatible ones – are available on the market, their availability is not as immediate as the immediacy with which users can reinvent the use of common objects with Protobject for sensing purposes. From another perspective, it can also be argued that a common object cannot become reactive with great simplicity and flexibility. In fact, to make an object reactive, it is necessary to modify it by installing electronic and mechanical components (e.g., actuators, motors, servos, etc., controlled by Arduino). Moreover, such as modifications almost always requires designing the necessary mechanics to install such actuators, for example using 3D printers. In such a case, the required technology skills are much higher, hence the number of potential users is reduced, but the resulting prototypes would be much richer and expressive. Also in this case, a simpler alternative could be the adoption of paper solution to emulate the actual prototype, letting designers and/or stake-holders perform the reaction [1]. We can also imagine a two steps approach, using Protobject extended with a paper prototyping in a preliminary stage, and moving to a solution with Arduino/3D printers for a more effective and fully working prototype later.

The discussion on limits is relevant because of the role Protobject can play at the educational level, to open the imagination of designers beyond a static conception of space, while giving technologists the possibility to 'see' the spatial state machine they should implement. Becoming a *reflective practitioner* [6] is not only a matter of spending time re-thinking what has been done, but also to learn to see things differently with new lenses.

VII. CONCLUSION

According to Preece et al. [4], the purpose of early stage design of interactive artefacts is to ''establishing requirements for the user experience'' and triggering ''idea generation and designing alternatives that meet those requirements.'' Such an activity can be supported by working prototypes that can help designers to better understand what is being designed and its impact on users and stakeholders.

Since the creation of interactive artefacts involves multi-disciplinary skills, ranging from interaction designers, to developers, to domain experts, the early development of prototypes is of great help to support brainstorming and set up a common understanding and establishing shared objectives.

Therefore, prototyping tools should be inexpensive and practical to provide a working environment to be tested as soon as possible. In such a way, it is possible to quickly refine the design of interactive applications until all the involved stakeholders are convinced of the value of the project and an engineering phase can start to deliver the final product.

The use of common objects (such as smartphones, doors, umbrellas, pencils, etc.) promoted by Protobject is inexpensive, and can be easily installed anywhere to simulate real environments. In addition, new ideas can be fostered by the collaboration among team members, who are encouraged to bring in their individual skills. Particularly, designers can concentrate on spatial organization, while technologists on functions, to deliver a joint multi-disciplinary artefact.

In Protobject, the collaboration starts with the definition of the portions of space involved in the prototype and the association of states with space configurations. This activity can be carried out jointly to establish a common understanding and provide the basis for the development of interactive prototypes. The observation at the workshop gave evidence to this intuition: the materiality of the experience designers lived with Protobject allowed them to promote a common understanding of the designed space. Their participation was facilitated by the perception of low technological demand and allowed them to understand the transition from spatial situations to the definition of representative states. They understood that the crucial aspects to design interactive systems are to transform the space into a state machine, and design state transformations that are spatially – and visually – characterized.

While two workshop sessions may not be enough to fully understand user perceptions of Protobject and its usability, they were helpful in understanding the pros and cons, and allowed us to understand how users use it, and what can be improved. Somehow, we expected Protobject to facilitate cooperation between users with different skills (e.g., designers and technologists) and to be useful in the early stages of design. Anyhow, the most relevant unexpected feedback was its value for learning purposes. The workshop qualitative results suggest that Protobject facilitates the understanding of interactive system programming. In fact, many participants said they had learned how to program interactive systems, even though it was not the focus of the workshop.

We believe the learning aspect using Protobject deserves further investigation, and we started using it in university courses: it was one of the experimental tools used in introductory programming courses in 2018 and 2019. A main reason, confirmed by the experiment, was to let students understand the potential impact of programming in real-world environments. The Protobject-supported approach to real-world programming was perceived intuitive and fun, even more then game-based programming [31]. Finally, the development of a new web-based version of Protobject is underway (https://protobject.com), with a specific focus on educational use. The aim is to expand its prototyping capabilities using not only cameras, but also the various sensors and actuators found in smartphones.

REFERENCES

[1] M. Beaudouin-Lafon and W. Mackay, "Prototyping tools and techniques," in *The Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies and Emerging Applications*. USA: Lawrence Erlbaum Associates, 2002, pp. 1006–1031. [Online]. Available: https://dl.acm.org/doi/10.5555/772072.772136

[2] A. Bellino, "Protobject: A sensing tool for the rapid prototyping of UbiComp systems," in *Proc. ACM Int. Joint Conf. Pervasive Ubiquitous Computing, Adjunct*, Sep. 2016, pp. 257–260.

[3] F. Cabitza and C. Simone, "Building socially embedded technologies: Implications about design," in *Designing Socially Embedded Technologies Real-World*. Cham, Switzerland: Springer, 2015, pp. 217–270.

[4] Y. Rogers, H. Sharp, and J. Preece, *Interaction Design: Beyond Human—Computer Interaction*, 4th ed. Hoboken, NJ, USA: Wiley, 2015.

[5] P. Dourish, *Where the Action is: The Foundations of Embodied Interaction*. Cambridge, MA, USA: MIT Press, 2004.

[6] D. A. Schön, *The Reflective Practitioner: How Professionals Think in Action*, vol. 5126. New York, NY, USA: Basic Books, 1983.

[7] T. Binder, G. De Michelis, P. Ehn, G. Jacucci, P. Linde, and I. Wagner, *Design Things*. Cambridge, MA, USA: MIT Press, 2011.

[8] K. Willis, E. Brockmeyer, S. Hudson, and I. Poupyrev, "Printed optics: 3D printing of embedded optical elements for interactive devices," in *Proc. 25th Annu. ACM Symp. User Interface Softw. Technol.*, Oct. 2012, pp. 589–598.

[9] Y. Kawahara, S. Hodges, B. S. Cook, C. Zhang, and G. D. Abowd, "Instant inkjet circuits: Lab-based inkjet printing to support rapid prototyping of UbiComp devices," in *Proc. ACM Int. Joint Conf. Pervasive Ubiquitous Comput.*, Sep. 2013, pp. 363–372.

[10] V. Savage, A. Head, B. Hartmann, D. B. Goldman, G. Mysore, and W. Li, "Lamello: Passive acoustic sensing for tangible input components," in *Proc. 33rd Annu. ACM Conf. Hum. Factors Comput. Syst.*, Apr. 2015, pp. 1277–1280.

[11] G. Laput, E. Brockmeyer, M. Mahler, S. E. Hudson, and C. Harrison, "Acoustruments: Passive, acoustically-driven, interactive controls for handheld devices," in *Proc. ACM SIGGRAPH Emerg. Technol.*, Jul. 2015, pp. 2161–2170.

[12] V. Savage, C. Chang, and B. Hartmann, "Sauron: Embedded single-camera sensing of printed physical user interfaces," in *Proc. 26th Annu. ACM Symp. User Interface Softw. Technol.*, Oct. 2013, pp. 447–456.

[13] S. Greenberg and C. Fitchett, "Phidgets: Easy development of physical interfaces through physical widgets," in *Proc. 14th Annu. ACM Symp. User Interface Softw. Technol.*, Nov. 2001, pp. 209–218.

[14] N. Villar, J. Scott, S. Hodges, K. Hammil, and C. Miller, "NET gadgeteer: A platform for custom devices," in *Proc. Int. Conf. Pervasive Comput.* Cham, Switzerland: Springer, 2012, pp. 216–233.

[15] G. D. Abowd and E. D. Mynatt, "Designing for the human experience in smart environments," in *Smart Environments*. Wiley, 2004, ch. 7, pp. 151–174. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/047168659X.ch7, doi: 10.1002/047168659X.ch7.

[16] S. Hodges, N. Villar, J. Scott, and A. Schmidt, "A new era for ubicomp development," *IEEE Pervasive Comput.*, vol. 11, no. 1, pp. 5–9, Mar. 2012.

[17] L. Oehlberg, W. Willett, and W. E. Mackay, "Patterns of physical design remixing in online maker communities," in *Proc. 33rd Annu. ACM Conf. Hum. Factors Comput. Syst.*, Apr. 2015, pp. 639–648.

[18] D. Salber, A. K. Dey, and G. D. Abowd, "The context toolkit: Aiding the development of context-enabled applications," in *Proc. SIGCHI Conf. Hum. Factors Comput. Syst. CHI Limit (CHI)*, 1999, pp. 434–441.

[19] T. Zimmer and M. Beigl, "AwareOffice: Integrating modular context-aware applications," in *Proc. 26th IEEE Int. Conf. Distrib. Comput. Syst. Workshops (ICDCSW)*, Jul. 2006, p. 59.

[20] S. Thomke and E. Von Hippel, "Innovators," *Harvard Bus. Rev.*, vol. 80, no. 4, pp. 74–81, 2002.

[21] M. Ono, B. Shizuki, and J. Tanaka, "Touch & activate: Adding interactivity to existing objects using active acoustic sensing," in *Proc. 26th Annu. ACM Symp. User Interface Softw. Technol.*, 2013, pp. 31–40.

[22] D. Maynes-Aminzade, T. Winograd, and T. Igarashi, "Eyepatch: Prototyping camera-based interaction through examples," in *Proc. 20th Annu. ACM Symp. User Interface Softw. Technol.*, Oct. 2007, pp. 33–42.

[23] D. Salber and J. Coutaz, "Applying the wizard of Oz technique to the study of multimodal systems," in *Proc. Int. Conf. Human-Computer Interact.* Cham, Switzerland: Springer, 1993, pp. 219–230.

[24] T. Grill, O. Polacek, and M. Tscheligi, "ConWIZ: The contextual wizard of Oz," *J. Ambient Intell. Smart Environments*, vol. 7, no. 6, pp. 719–744, Nov. 2015.

[25] C. Ardito, P. Buono, M. F. Costabile, R. Lanzilotti, and A. Piccinno, "A tool for wizard of Oz studies of multimodal mobile systems," in *Proc. 2nd Conf. Human Syst. Interact.*, May 2009, pp. 341–344.

[26] R. Sefelin, M. Tscheligi, and V. Giller, "Paper prototyping-what is it good for: A comparison of paper-and computer-based low-fidelity prototyping," in *Proc. CHI Extended Abstr. Human Factors Comput. Syst.*, 2003, pp. 778–779.

[27] A. A. Goshtasby, "Similarity and dissimilarity measures," in *Image Registration*. Cham, Switzerland: Springer, 2012, pp. 7–66.

[28] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2005, vol. 1, no. 1, pp. 886–893.

[29] W. M. Johnston, J. R. P. Hanna, and R. J. Millar, "Advances in dataflow programming languages," *ACM Comput. Surv.*, vol. 36, no. 1, pp. 1–34, Mar. 2004.

[30] F. Oliphant, "Meemoo: Hackable web app framework," M.S. thesis, School Arts, Des. Archit., Aalto Univ., 2012, p. 60. [Online]. Available: http://urn.fi/URN:NBN:fi:aalto-201206102286 and https://aaltodoc.aalto.fi/handle/123456789/3944

[31] A. Bellino, V. Herskovic, M. Hund, and J. Munoz-Gama, "A real-world approach to motivate students on the first class of a computer science course," *ACM Trans. Comput. Educ.*, vol. 21, no. 3, pp. 1–23, Sep. 2021.

ALESSIO BELLINO is currently an Assistant Professor with the School of Informatics and Telecommunications, Universidad Diego Portales, Chile. His research interests include interaction design, human–computer interaction, and ubiquitous computing.

GIORGIO DE MICHELIS is currently a Senior Professor with the Department of Informatics, Systems and Communication, Università di Milano–Bicocca, Italy. His research interests include interaction design, information systems, CSCW, and knowledge management.

FLAVIO DE PAOLI is currently an Associate Professor with the Department of Informatics, Systems and Communication, Università di Milano–Bicocca, Italy. His research interests include human–computer interaction and distributed systems.