A Language for the End-user Development of Mobile Context-Aware Applications

Valentim Realinho[1]*, Teresa Romão[2], and A. Eduardo Dias[2]
[1]*VALORIZA, Instituto Politécnico de Portalegre, Portugal*
vrealinho@ipportalegre.pt
[2]*NOVA LINCS, DI-Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa, Portugal*
tir@fct.unl.pt, aed.fct@gmail.com

Received: January 3, 2019; Accepted: January 27, 2020; Published: March 31, 2020

Abstract

This paper presents the IVOML language, the foundation of the IVO (Integrated Virtual Operator) platform which enables the creation, deployment and execution of context-aware mobile applications by end-users without programming skills. We use an event-driven workflow model to describe the behaviour of the applications. Thus, whenever the defined context occurs, the corresponding workflow immediately starts reproducing the desired behaviour without the need of user intervention. The IVO platform comprises two composite tools that provide a visual programming environment for editing IVOML files. IVOML can be used at runtime in mobile devices, by providing interpreters that run the applications written with the composite tools. This allows any created application to be distributed and executed on mobile devices based on various platforms without the need for maintaining separate versions. To test the whole platform, an Android and an iPhone runtime were created which provide the necessary support for the execution of the applications developed using IVO.

Keywords: ubiquitous computing, context-awareness, mobile devices, XML-based language

1 Introduction

The advent of the Internet of Things and mobile applications has made the possible contexts of use more and more varied, and creates new challenges for user interface developers [1]. Context-aware computing refers to applications that can dynamically adapt to changes in the user activities and environments [2]. The context is a particular situation for each user, and it can include, among others, location, activity being performed, time and nearby people or equipment. Context-aware computing involves sensing those situations to provide adequate information and services to the user.

Context-aware applications have already demonstrated the advantages of perceiving the surrounding environment [3, 4, 5, 6]. Some research has focused on developing frameworks, toolkits and infrastructures to support programmers in building context-aware applications. Other projects use end-user programming techniques, which empower end-users to prototype context-aware applications [2, 7, 8]]. LoMAK [9] enable domain specialists to generate locative media mobile apps from KML files. SMAT [10] is a mobile app that supports the authoring of locative media experiences with a focus on the creation of POIs (Points of Interest) and associated geo-fences which trigger the pushed delivery of media items such as photos or audio clips. Projects like App Inventor [11] and Puzzle [12] are more suitable

Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA), 11(1):54-80, Mar. 2020
DOI:*10.22667/JOWUA.2020.03.31.054*
*Corresponding author: VALORIZA, Instituto Politécnico de Portalegre, Campus Politécnico 10, Portalegre, Portugal, Tel: +351-245300200

for developing generic mobile applications lacking the support for specific issues concerning context-awareness.

Mobile applications must be aware of the physical environment while the Internet of Things makes the environment even more heterogeneous, making the write of such context-aware applications a fundamental problem of modern software engineering. Currently, there is no specific programming language for writing such applications, and most projects rely on ad-hoc approaches that use generic programming languages with the support of libraries that facilitates the sensing of the environment and interaction with the user.

Thus, the development of such applications is a laborious process requiring considerable expertise in both programming and interaction design. In particular, the implementation of their control flow often becomes a complex task.

To address the problem of the laborious work and specific expertise involved in building mobile context-aware applications, we have developed the IVO platform. We use an approach based on context-aware workflows to generate the end-user applications developed with IVO. In the heart of IVO, there is an XML-based language called IVOML, which enables the development of mobile context-aware applications. The choice for this approach is justified by the fact that XML [13] is a widely used standard that guarantees the interoperability and facilitates the distributed communication, without the need to use more formal models for representing context like logic or ontologies based models [14]. Furthermore, XML provides a meta-language suitable for representing and manipulating source code that could be used by external programs. As a proof of concept, we designed two visual programming tools that allow end-users, with no programming skills, to build and deploy mobile context-aware applications. By using these tools, users only use visual building blocks without the need to write any line of code using the IVOML language. IVOML can be used at runtime in mobile devices, by providing interpreters that run the applications written with the building tools. This allows any created application to be distributed and executed on mobile devices based on various platforms without the need for maintaining separate versions. We developed a mobile framework, currently available for Android and iOS platforms, that provides the necessary runtime able to execute the applications.

To summarize, the contribution of this paper is a proposal for an XML-based language for the development of mobile context-aware applications. The visual programing tools for the constructions of such applications and the framework for mobile devices mentioned above are detailed in our previous works [15, 16, 17].

The rest of this paper is structured as follows. In Section 2, we summarize the state of the art related to academic and industrial projects that were used as a basis and inspiration to create IVOML. Section 3 presents the IVO platform that allowed us to proof the concept behind the IVOML language. In Section 4, we present in detail the IVOML language. Section 5 presents the evaluation performed on the entire IVO platform, including both the visual programming tools and the device side. Section 6 concludes the paper and outlines some future directions.

2 Related Work

Context-aware workflow became an important research topic in order to introduce context awareness and mobility capabilities to workflows. The pioneer work of Jing et al. [18], Chakraborty and Lei [19], Cho et al. [20] and Jang et al. [21], benefited from previous research in the area of workflow management. Workflow management systems are not new and there are several products, both commercial as IBM WebSphere Application Server [22] and Microsoft BizTalk Server [23] or open source such as Apache ODE (Orchestration Director Engine) [24], the Orchestra [25] and the Riftsaw [26] (the latter based on Apache ODE).

Most research on context-aware workflow topic focuses on the issue of modelling and execution engines. WS-BPEL [27] (or BPEL for short) became the standard for workflow execution language. BPEL is a programming language based on XML for describing business process behaviours based on Web Services. It extends the Web Services interaction model and enables it to support business transactions. BPEL is popular within the open source community and several BPEL engines such as Apache ODE [24], Orchestra [25] and Riftsaw [26] have been implemented. IBM WebSphere Application Server [22] and Microsoft BizTalk Server [23] are examples of commercial BPEL engines that are able to execute the BPEL workflow models. Natively, BPEL does not support mobile users or context-aware workflows, which opens space for the research we have been conducting in this area.

Context4BPEL [28] and CAWE [29] focus on the use of frameworks and execution engines for specific domains being unable to give a more general response. Context4BPEL is a variant of BPEL for context-aware workflow modeling and has been tested in production processes in the manufacturing industry. CAWE is a framework for the development of context-aware applications based on Web Service technologies, which adapt to the user features and the execution context and was tested in a hospital management scenario.

Projects like Sliver [30], xBPEL [19], WHAM [18] and the one proposed by Pajunen and Chande [31], are execution engines that support mobile users. xBPEL and WHAM use only the location as the unique type of context data, and therefore are not generic enough to enable modeling of context-aware workflows.

CAWD [32] provides a design tool used to develop workflow models for context-aware applications. It is built on top of Microsoft Windows Workflow Foundation 3.5 and acts as a standalone application for designing and verifying context-aware workflows. Tang el al. [33] propose a context model and a context-aware workflow management algorithm for ubiquitous campus navigation modeled through Petri nets.

The event-driven workflow model that we use in IVO was in part inspired by the pioneering work of Dayal el al. [34] where the ECA (Event-Condition-Action) notation was proposed. Jang et al. [21] proposed a similar approach called GAT (Guard-Action-Trigger) for the design of a workflow engine that supports the definition and execution of long-running business processes. Ghiani et al. [35] present another similar approach for an authoring tool which supports the development of context-dependent user interfaces based on trigger/action rules. We call our model ECW (Event-Condition-Workflow).

These projects seek to add mobility and context-awareness to workflows, with the mobile device acting as an extension of the workflow system, following a service orchestration logic. The major difference between these projects and IVO is in the way a workflow is considered. For us a workflow is not as a business process, but part of a context-aware application that is created and used by end-users. We use the same context-aware workflow concepts but in a logic of local execution of the available activities. Activities are used as the building blocks of the workflows and are triggered when the defined start context is checked.

The proposed language presented in this paper aspires to allow the construction of a wide range of mobile context-aware applications for several purposes. It includes (i) activities or tasks commonly performed on smartphones like changing the profile, sending an email or SMS, among others; (ii) flow control activities like alternative or conditional paths, loops and parallelism, and (iii) advanced activities with user interaction mechanisms like forms, voice or integration with social networks. The full set of activities can be extended to enable the creation of a wider set of applications.

3 The IVO Platform

IVOML is the foundation of the IVO platform, which enables the creation, deployment, and execution of context-aware mobile applications by end-users without programming skills. In this section, we present the IVO platform, which includes two modelling tools for editing IVOML, files, and a mobile framework runtime that run the applications built with these tools.

The IVO platform and the performed tests allowed us to evaluate the IVOML language capabilities to create useful applications for various purposes including entertainment and leisure applications as well as business applications.

The two modelling tools provide a visual programming environment for the creation of mobile context-aware applications, enabling end-users to define the workflows and to set the contexts that are used to define the conditions that trigger the workflows, by IVOML automatic editing. The platform architecture, illustrated in Figure 1, is succinctly presented below, while a more detailed description can be found in our previous publications [15, 17].

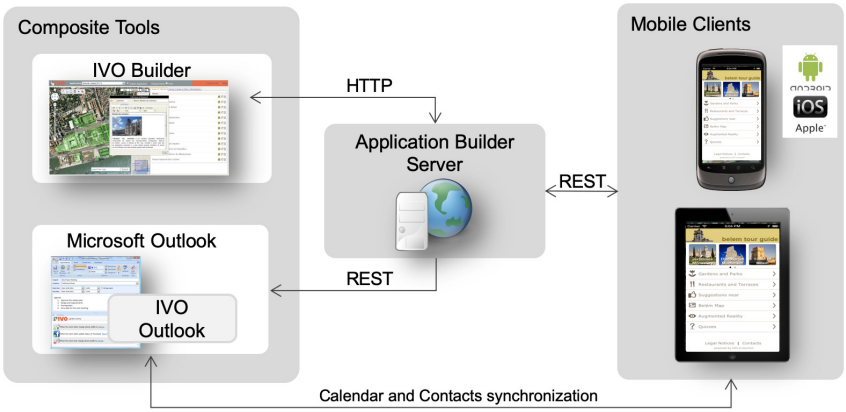


Figure 1: IVO architecture

3.1 Application Builder Server

The Application Builder Server provides the web pages used by IVO Builder and stores the created applications. It also provides a REST (Representational State Transfer) API that makes the services used by the mobile clients available.

3.2 Composite Tools

IVO provides a completely visual programming environment that allows end-users to define a number of context conditions and workflows of activities, which are later triggered when the user is in the presence of those contexts. Two main building tools are available, allowing end-users with no programming skills to create mobile context-aware applications: IVO Builder, a web application; and IVO Outlook, for temporal and proximity contexts, composed of two *ons*, for Outlook Calendar and Outlook Contacts. Both of these tools are described next.

3.2.1 IVO Builder

Figure 2 illustrates the main screen of IVO Builder (main IVOML editor), loaded with a Tourist Guide application for Barcelona, Spain. IVO Builder is a web application that gives access only to registered

users. Each user develops his/her own applications in his/her area of the server. He/she can, however, make the developed applications available to other users.

The left side of the screen (zone 2 in Figure 2) is reserved for the geographical representation, which is overlaid with the location contexts (or geo-fences) represented in green with icon marks. The toolbar in this area gives access to basic features for creating new location contexts, namely: (i) creation of a geo-fence defined by a point; (ii) creation of geo-fence represented by a circle; (iii) manual creation of a geo-fence by defining the polygon that defines it; (iv) import geo-fences from geo-referenced articles on Wikipedia; (v) import geo-fences from existing areas in Wikimapia and (vi) from KML [36] files.

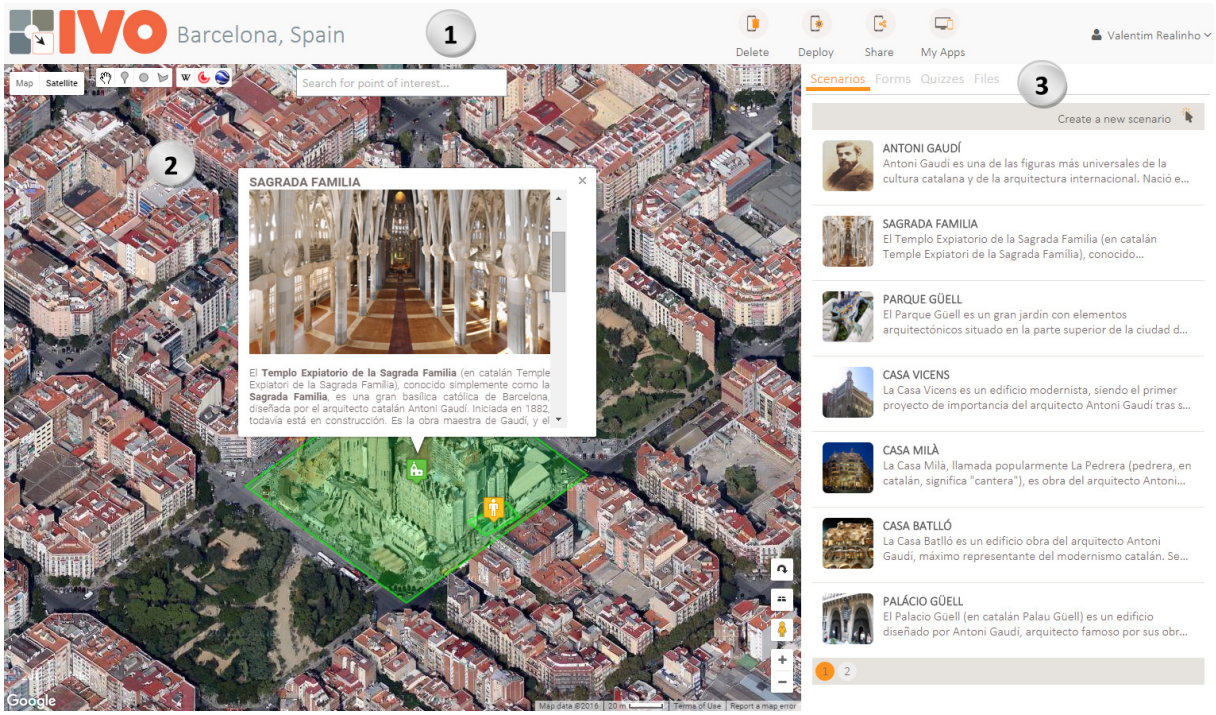


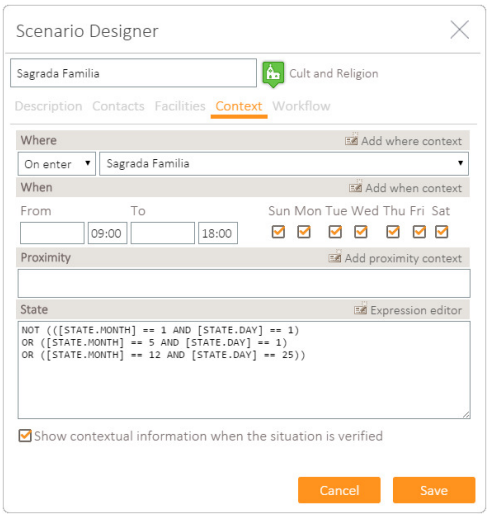
Figure 2: IVO Builder main screen loaded with a Tourist Guide application for the city of Barcelona, Spain. 1 – Top menu with application manager options; 2 – The map helps the user to create location contexts. The location contexts are marked in green with an associated icon, which represents the category of the place. 3 – This area contains IVO application components like scenarios, forms, quizzes and files including images, audio, video and scripts.

In addition to the toolbar, there are the usual controls to zoom and change the map view, satellite or hybrid, and a search box that allows users to quickly find a place through Google search APIs. The map also includes a 45-degree view (like shown in Figure 2) and a street view, which are convenient for a better identification of places. The right side of the main screen (zone 3 in Figure 2) is reserved for the representation of information on the application itself, such as the scenarios, forms, quizzes and files (including images, audio, video and scripts used by the application).

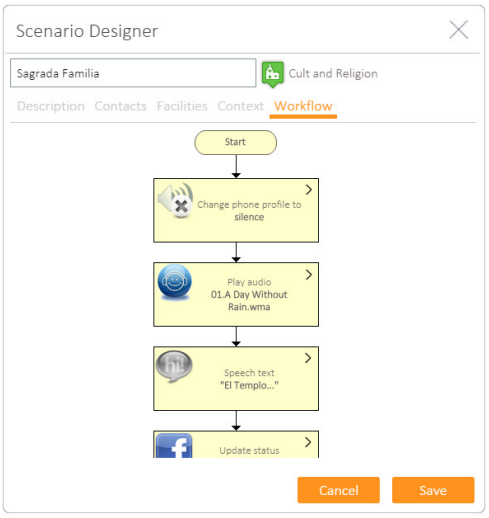
Once created, the geo-fences can be associated to “Where” dimensions of context (see Section 4.2) when defining a scenario (Figure 3a). In the example of Figure 3a, the context corresponds to the user entry in the area defined for the Sagrada Familia, on any day between 9:00h and 18:00h except on January 1st, May 1st and December 25th.

Figure 3b illustrates the workflow designer interface. Each rectangle corresponds to an activity and the arrows illustrate transitions which may have a condition guard associated. A transition from one activity to the next only occurs if the guard condition is verified, otherwise the workflow activity blocks

until the condition is met. The workflow starts when its starting context occurs (the conditions that define the scenario's context – Figure 3b - are true). In this case, we have a sequential workflow that puts the device in silent mode; plays an audio file and superimposes the message referred in the text-to-speech activity; and, posts a message on the user's Facebook wall. The complete set of currently available workflow activities (see Table 3) provides a wide range of capabilities, while keeping the system accessible to both novice and expert end-users. This set of activities can be extended to support the creation of a broader set of applications.



(a) Context associated with the scenario



(b) Workflow designer

Figure 3: Scenario designer

3.2.2 IVO Outlook

Besides IVO Builder, the platform provides users with IVO Outlook, a tool to develop applications based on temporal and proximity contexts, thus allowing for a richer context definition. IVO Outlook allows users to easily define temporal conditions, through the incorporation of workflows in the Outlook appointments, which are executed *"when the event starts"* or *"when the event ends"*. The IVO's Outlook *add-on* adds a form region at the bottom of the appointment window, containing the interface that enables the users to define workflows that may include the same activities that are available in IVO Builder. The IVO Outlook can also associate location contexts with the appointments, to allow the creation of rules like *"at this instant in time and when I'm at this location"*. These location contexts can be defined using a map provided by IVO Outlook or can be imported from the applications created with IVO Builder through a REST API. Proximity contexts can be created by associating a Bluetooth device to an Outlook contact. The synchronization tools provided by the smartphone manufacturer guarantee the synchronization of IVO Outlook with the mobile device.

3.3 Mobile Client Framework

Figure 4 illustrates the framework of the IVO Client which is composed by three major components: Sensing Monitor, Event-Condition-Workflow Engine and the Workflow Engine. The User Interface supplies the necessary services for the user.

Sensing Monitor. The sensing layer allows monitoring the contextual environment and can be extended through the development of new sensor handlers to enhance the system's capability to perceive the environment. The creation of these handlers is a task that involves programming an abstract class extension. Currently we have sensor handlers to provide location contexts through GPS, GSM and WIFI networks; Bluetooth for proximity contexts of people and equipment; 2D barcodes reading (QR-codes); temporal contexts through the smartphone Calendar; and Near Field Communication (NFC) [37]. In addition to the smartphone's internal sensors, it is possible to develop sensor handlers that communicate with an external sensor infrastructure.

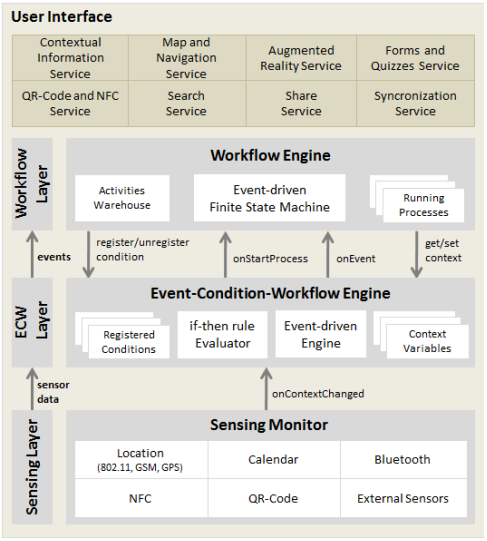


Figure 4: IVO Client Framework

Event-Condition-Workflow Engine. This layer provides a simple but powerful mechanism for register/unregister conditions, that the Workflow Engine uses in order to be asynchronously notified by the event-driven engine whenever a certain condition holds. Whenever an `onContextChanged` event is raised by the Sensing Layer, the event-driven engine evaluates all registered conditions. The event `onStartProcess` is raised when a workflow start condition is detected, while onEvent is raised whenever the condition associated with an asynchronous activity is verified. The conditions are evaluated by the if-then rule evaluator which uses Java Expression Language (JEXL) [38], therefore allowing a great expressiveness. Boolean expressions used in conditions can make use of all context variables and common operators, such as $>$, $>=$, $<$, $<=$, $=$, $!=$, AND and OR.

Workflow Engine. The Workflow Engine internal logic is built on an event-driven finite state machine (FSM) with guards represented through an activity transition table. The finite state machines are very expressive and can be programmed by the purely event-driven approach of specifying the activity/-transition graph, as exemplified in Figure 3b. Each state (represented as a rectangle) corresponds to an activity available in IVOML. The arrows illustrate transitions and can have a guard that refers to conditions defined by the context variables. A transition only occurs if the transition guard condition holds. The FSM starts with the workflow start condition. This condition can include location contexts, temporal contexts, proximity contexts and any expression that uses the context variables.

User Interface. This layer provides the interaction with the user through the designed screens that provide information in many ways such as text, image, audio, video, augmented reality and haptic feedback. Text-to-speech and voice recognition are also available, being this layer divided in accordance with the types of service available to the user: (i) contextual information; (ii) map and navigation; (iii) aug-

mented reality; (iv) forms and quizzes; (v) QR-Code and NFC readers; (vi) search; (vii) share; and (vii) synchronization services. Figure 5 illustrates the user interfaces of several iOS applications developed using IVO: (a) main screen of a Tourist Guide application; (b) contextual information associated with one of the spots; (c) results of one quiz; and (d) an augmented reality interaction using the smartphone camera.

(a) (b) (c) (d)

Figure 5: Example of applications user interfaces: (a) main screen of a Tourist Guide of Belem, Lisbon, Portugal, (b) contextual information associated with one spot, (c) a quiz about the Ulm Minster, Germany and (d) nightly augmented reality in Ulm, Germany

4 IVO Markup Language

This section presents the IVO Markup Language (IVOML), an XML-based language for the development of mobile context-aware applications. IVOML is the foundation of the IVO platform which enables the creation, deployment and execution of context-aware mobile applications by end-users without programming skills.

The basic design principles that guided the definition of IVOML were an event-driven workflow model and a scenario-based design.

Event-driven Workflow Model. We use an event-driven workflow paradigm, in which workflows are started by events that result from the evaluation of an *if-then rule engine*. Workflows determine the program flow, expressed as a set of activities, which can include alternative or conditional paths, loops and parallelism. This approach enable the dynamic nature of context-aware applications: the workflows must be initiated only when the user is in the presence of a pre-defined context detected by the application; changes in the environment can occur after the workflow has been started and can potentially alter the workflow and what it is intended to be accomplished [17]. In addition, we support the concept of guard conditions that is used to condition the occurrence of a transition from one workflow activity to the next. All activities can have an associated condition, allowing the transition to the next activity if the condition holds. If the condition is not verified, the activity enters in a blocked mode until it is checked, or a timeout occurs.

Scenario Based Design. We use a scenario based design because scenarios are appropriate to de-

scribe the behaviour of a system when reacting to external environment stimuli [33, 34, 35], and are therefore an interesting mechanism for describing context-aware applications. Potts [39] defines a scenario as a particular case of how the system should be used, or in a broader sense, a scenario is simply a proposed specific use of the system which can be translated in the form of short stories [40]. In IVOML, a scenario is defined by the following elements:

(i) **Contextual information** associated with the scenario which can be static or dynamic and able to adapt to context. This information should be able to be displayed on mobile devices with screens of various dimensions.

(ii) **Context** which precisely defines a situation in accordance with the location, time, proximity and state dimensions. The state dimensions refer to state variables of the mobile device, such as battery level, or brightness. The situation corresponds to a condition that is evaluated whenever there is a change in the environment in which the user is.

(iii) **Workflow** which should be executed when a situation applies.

(iv) **Forms and quizzes** that are executed as workflow activities. Forms are more intended for a business use and allow the introduction of data that can be stored in a server running a web-service. Quizzes can provide an interesting and amusing way to enrich users' experience for example in leisure and entertainment applications [16]. They are related to the real user activities and are essentially aimed at stimulating more active participation from users.

Based on the design principles presented above, we defined IVOML, whose main schemas are presented in the next sections. Not all the XML schemas could be included here due to lack of space. Those shown, illustrate however the general idea.

4.1 Application

Figure 6a represents the root element of IVOML which represents the main element of an application. The constraints, which are hidden in Figure 6a, defines the validation rules between the various objects, namely the relationships illustrated in Figure 6b.

These relationships represent the dependency of the objects and facilitate the implementation of applications avoiding redefining contexts and workflows. Contexts are the main element and may exist in one or more usage scenarios. The scenarios allow to define the application's behaviour (or whatever the corresponding workflow) when a particular context occurs. Thus, one context can be part of several scenarios and the same workflow can also be used in several scenarios.

The attributes of the root element allow the global definition of the applications and are described in Table 1. These attributes allow the global characterization of the application. Next, we describe each of the elements that constitute an IVO application.

4.1.1 Application description

The application `description` can be set through the optional element description, which can contain HTML5 code for a richer description of the application. The developed IVO Client shows the description of the application as a splash screen and whenever the user clicks in the "About" button available.

4.1.2 Home

The application home screen can be defined in the optional element `home` which can contain HTML5 code for a richer home screen of the application. The home screen can include, in the HTML code,

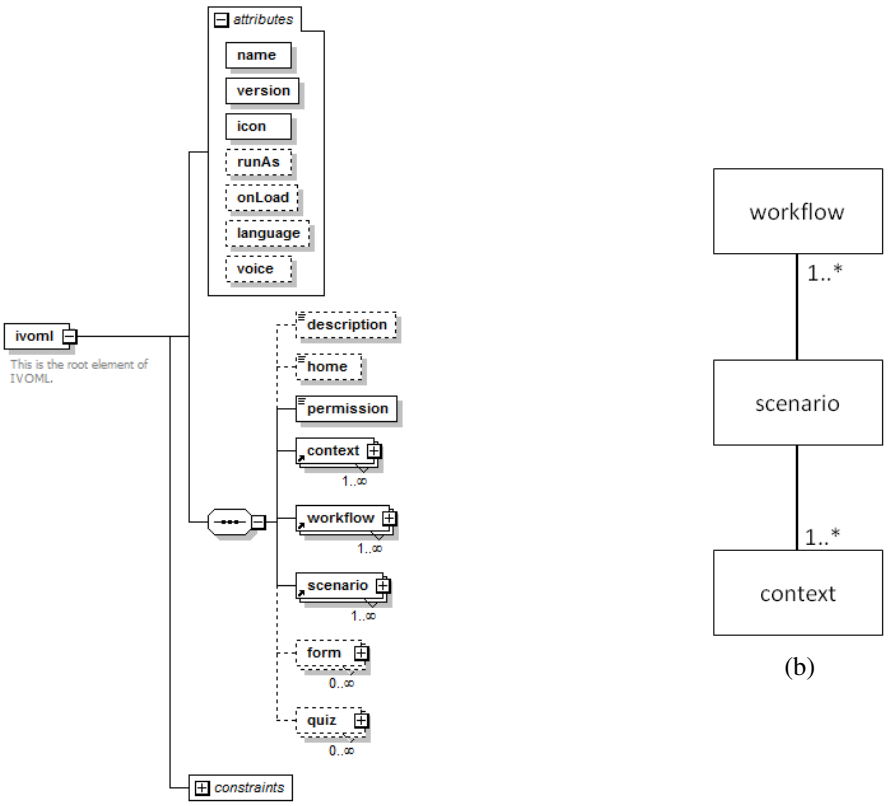


Figure 6: (a) General schema of IVOML with the root element of an application, and (b) main relationships between objects

Table 1: Main attributes of an application

Attribute	Description
`name`	Application name
`version`	Application version, which can be used to determine changes in applications and the need for updates.
`icon`	Application icon.
`runAs`	Enables to define the operating mode of the application and can take the values: (i) `service` that allows the application to run in background as a service, or (ii) `application` that allows the application to run as a normal application with user interface.
`onLoad`	Optional attribute that indicates the workflow that is started when the application is loaded.
`language`	Optional attribute that indicates the language used in the various contents of the application. Corresponds to the international country code and if nothing is set, the value set on in the mobile device is used.
`voice`	Optional attribute that indicates the language used in spoken contents of the application (text-to-speech) and voice recognition activities of workflows. Corresponds to the international country code and if nothing is set, the value set on in the mobile device is used.

special tags to direct access to all the features available in the mobile device, like a particular context (or category of context), quiz, form, map or augmented reality browser, as illustrated in the application home screen of Figure 5a.

4.1.3 Application permissions

The element `permission` defines the requirements of the application, which corresponds to a list of values shown in Table 2.

Table 2: Application permissions

Permission	Description
`internet`	The application needs an internet connection.
`gps`	The application uses GPS locations.
`network`	The application uses network locations.
`bluetooth`	The application uses bluetooth services.
`ar`	The application uses the augmented reality browser.
`calendar`	The application uses the calendar for the definition of time contexts.

4.1.4 List of elements

Each application can include five lists, representing the contexts (`context` element), the workflows (`workflow` element), the scenarios (`scenario` element), the forms (`form` element) and the quizzes (`quiz` element) used by the application. The next sections describe the individual schemas of each list.

4.2 Context

The definition of contexts is made by the `context` element whose schema is shown in Figure 7. Each context is defined by a name, an optional `category` and an `icon`. The optional `contextual-information` element represents the data associated with the context (e.g. the textual description and photo of a location) and can contain HTML5 code for a richer information. The `facilities` and `contact` elements are two other elements that structure the information associated with the context. A context may contain several locations, temporal, proximity and state dimensions (`where`, `when`, `proximity` and `state` elements), and should include at least one of these elements.

Each context dimension may comprise multiple instances of the same type. Generally, the evaluation of the context associated with a scenario is given by the following equation:

$$Context = \prod(\sum Where_n, \sum When_n, \sum Proximity_n, State) \quad (1)$$

Figure 8 shows an example of a context which uses one location, one temporal and one state dimension. The location corresponds to the placemark (or geo-fence) defined as Jeronimos Monastery, the temporal dimension refers to the Jeronimos Monastery summer time schedule, and the state dimension refers to exceptions of this schedule. So, this context corresponds to a situation in which the user enters the Jeronimos Monastery during the summer time schedule (from 1st May to the 30 September, all weekdays except on Mondays, January 1st, May 1st and the Christmas day).

Figure 9 shows another example of a context that only uses the state dimension. In this case, the context corresponds to a situation where the mobile device battery level is less than 10%.

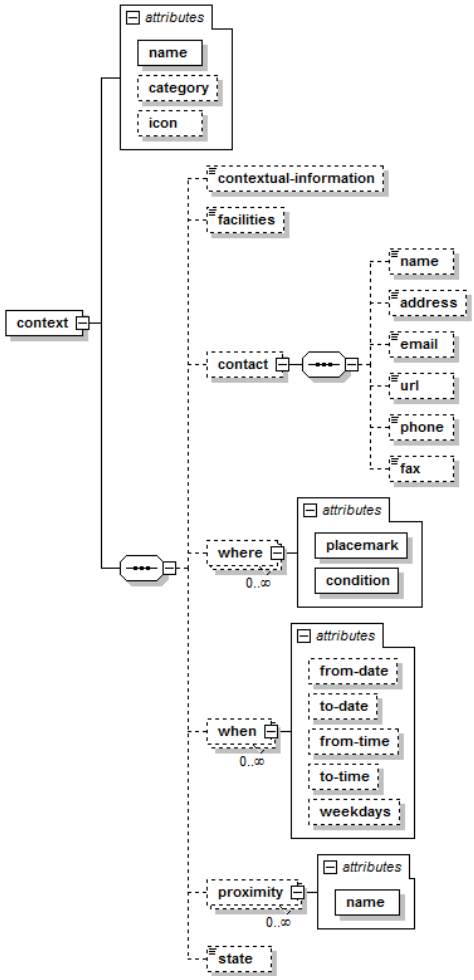


Figure 7: Context schema

```
<context name="Jeronimos Monastery - summer time"
                 category="Monuments">
   <where placemark="Jeronimos Monastery"
                 condition="onEnter"/>
   <when from-date="05-01" to-date="09-30"
        from-time="10:00:00" to-time="18:00:00"
        weekdays="Sun Tue Wed Thu Fri Sat"/>
   <state>
      <![CDATA[
      NOT (([STATE.MONTH] == 1 AND [STATE.DAY] == 1)
        OR ([STATE.MONTH] == 5 AND [STATE.DAY] == 1)
        OR ([STATE.MONTH] == 12 AND [STATE.DAY] == 25))
      ]]>
    </state>
</context>
```

Figure 8: Context example

```
<context name="Battery Low">
    <state>
        <![CDATA[
            [STATE.BATTERYLEVEL]<0.1
        ]]>
    </state>
</context>
```

Figure 9: Example of a state context

4.3 Workflow

An application can contain multiple workflows each corresponding to the desired behaviour of the application when a certain context is verified. Figure 10a represents the schema of a workflow, which is characterized by the name, the `show-contextual-information` attribute which indicates whether the contextual information associated with the context should be shown to the user, and one sequence of activities that define the workflow (described in Table 3). Figure 10a omits many of the activities available as the format of such always follows a similar pattern. In fact, most of the activities consist of empty elements with only a set of attributes that defines the activity, as illustrated in Figure 10b and Figure 10c.

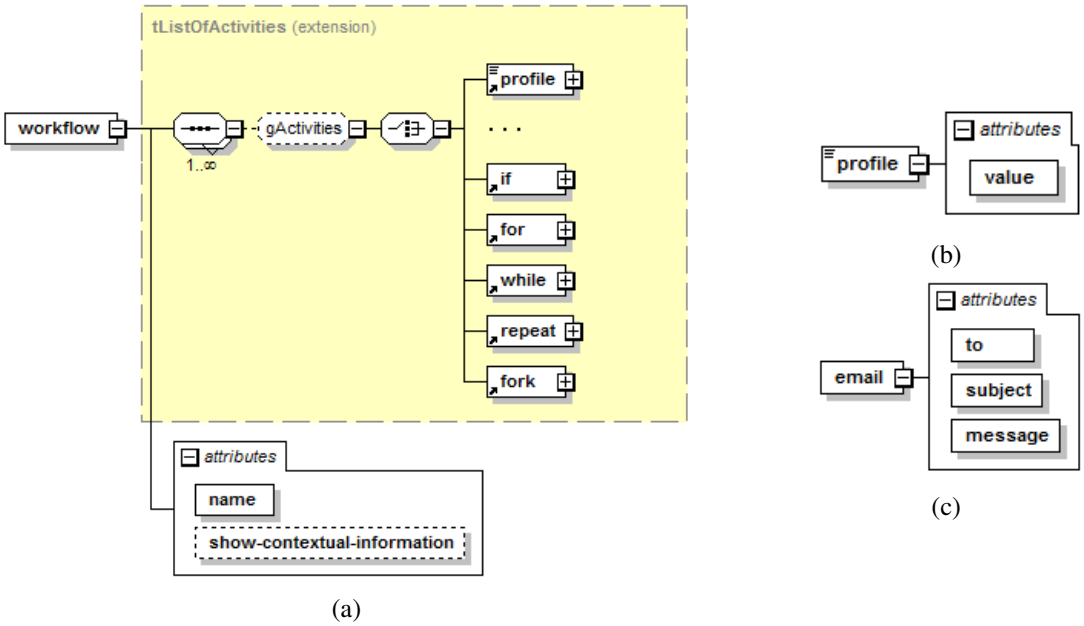


Figure 10: (a) Workflow schema and schema of (b) profile and (c) email activities

The activities currently available were chosen take into account the basic primitives of BPEL [33] and the basic building blocks defined by the Workflow Management Coalition (WfMC) [41]. They can be classified in three categories as follows.

Common activities. Activities or tasks commonly performed on smartphones like changing the profile, sending an email or SMS, among others.

Advanced activities. Provide more advanced user interaction mechanisms like advanced forms or voice; provide integration with social networks and scripting support for the development of more advanced features.

Flow control activities. Allow modelling sequential, parallel, conditional and iterative processes. This activity group provides equivalent mechanisms to the programming languages instructions of the

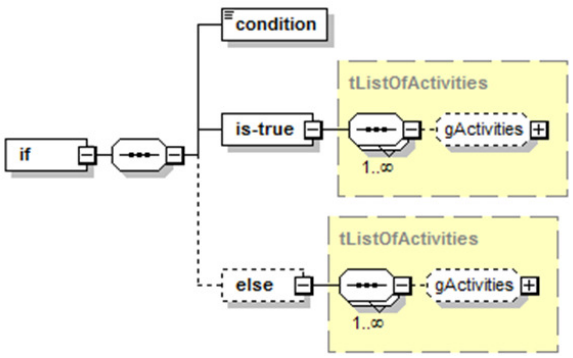
Table 3: Workflow supported activities

Activity	Description
Common activities	
`profile`	Changes the phone profile including turning on and off sound, Bluetooth, wifi and gps.
`dial`	Dials a phone number.
`sms`	Sends an SMS to a phone number.
`email`	Sends an email.
`alert`	Displays a warning window to the user.
`message`	Shows a message to the user.
`input`	Allows data input.
`audio`	Allows control of an audio file (play, pause, resume and stop).
`video`	Allows control of a video file (play, pause, resume and stop).
`photo`	Launches the camera to take a picture or choose one from the gallery.
`url`	Opens a Web page.
`webservice`	Executes a web-service.
Advanced activities	
`run`	Runs an application.
`script`	Runs a script.
`navigateto`	Launches the navigation browser to guide the user to a particular location.
`facebook`	Updates Facebook status.
`twitter`	Updates Twitter status.
`assign`	Assigns a value to a variable.
`executeform`	Executes a form.
`executequiz`	Executes a quiz.
`tts`	Text to speech.
`voicerecognition`	Voice recognition.
`waitfor`	Waits for a given condition.
`sleep`	Waits for a defined time (in seconds).
Flow control activities	
`if`	Conditional path.
`for`	For loop.
`while`	While loop.
`repeat`	Repeat loop.
`fork`	Executes blocks of activities in parallel.

types if-then-else, for, while, repeat-until and fork. These activities have a recursive scheme since they may contain blocks of elements within the activities. The `tListOfActivities` type (the corresponding activities are hidden in the following schemes to save space) corresponds to the same element type illustrated in the workflow schema of Figure 10a. As stated before, this set of supported activities can be extended to support the creation of a wider set of applications.

4.3.1 Conditional paths – the if activity

Figure 11a illustrates the `if` activity schema, and Figure 11b, a workflow that uses this activity to define conditional flows.



(a)

```
1  <workflow name="Order">
2    <executeform name="Order"/>
3    <if>
4      <condition>
5        <![CDATA[[Order.Discount] > 10]]>
6      </condition>
7      <is-true>
8        <email subject="Approve order"
9            message="Please approve the order."
10           to="manager@mycompany.com"/>
11     </is-true>
12     <else>
13       <webservice
14           url="http://www.mycompany.com/order.php"
15           user="xxx" password="xxx"/>
16     </else>
17   </if>
18 </workflow>
```

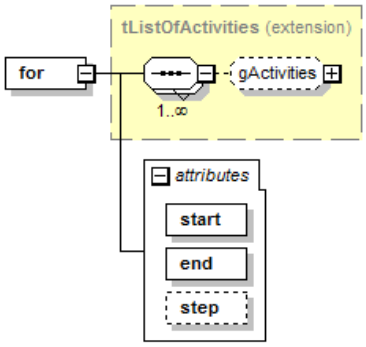
(b)

Figure 11: (a) if schema and (b) a workflow example with a condition

In the example of Figure 11b, the `Discount` variable of `Order` form (lines 4 to 6) is used in the condition (of the `if` activity) to determine the flow that the workflow must follow. If this value is greater than 10, an email is sent (lines 7 to 11), otherwise the specified web-service is executed (lines 12 to 16). In this example, each branch of the if statement contains only one activity, but it may actually contain one indefinite number of any of the available activities.

4.3.2 Loops - for, while and repeat activities

Like programming languages, `for`, `while` and `repeat` activities are used to repeat a block of activities a certain number of times (`for` loop), while (`while` loop) or until (`repeat` loop) a given condition is checked. Figure 12a illustrates the `for` loop schema, and Figure 12b, an example that speaks three times the text `"Hello world!"` (line 2) with an interval of ten seconds between them (line 3).



(a)

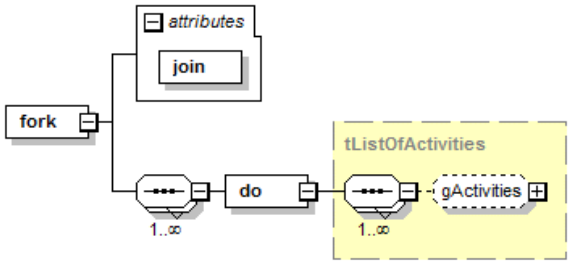
```
1  <for start="1" end="3">
2    <tts text="Hello world!"/>
3    <sleep time="10"/>
4  </for>
```

(b)

Figure 12: (a) For loop schema and (b) an example of a for loop

4.3.3 Parallelism – fork activity

The `fork` activity allows performing blocks of activities in parallel. The synchronization of these blocks may be performed by a `join-and` wherein the workflow remains parallel until all blocks are completed, or by means of a `join-or` in which the workflow continues right after one of the blocks has been completed. Figure 13a illustrates the `fork` schema, and Figure 13b, an example where the execution of the web-service and the sending of the SMS, run in parallel, and continues only after both blocks have ended because it was determined that the synchronization of the blocks must be done through a `join-and` (line 1).



(a)

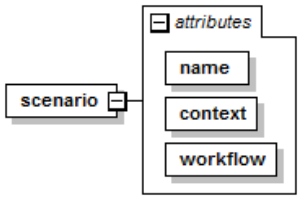
```
1  <fork join="and">
2    <do>
3      <webservice
4            url="http://www.mycompany.com/order.php"
5            user="xxx" password="xxx"/>
6    </do>
7    <do>
8      <sms phone="xxx" message="Order registered."/>
9    </do>
10 </fork>
```

(b)

Figure 13: (a) Fork schema and (b) example of a parallelism

4.4 Scenarios

A scenario represents a particular situation and the behaviour of the system when this situation occurs. Figure 14a shows the `scenario` schema, and Figure 14b, two scenarios corresponding to the situations (contexts) when the user enters the Jeronimos Monastery in summer time (lines 1-3) and in winter time (lines 5-7).



(a)

```
1  <scenario name="Jeronimos Monastery - summer time"
2          context="Jeronimos Monastery - summer time"
3          workflow="Entering Jeronimos Monastery"/>
4
5  <scenario name="Jeronimos Monastery - winter time"
6          context="Jeronimos Monastery - winter time"
7          workflow="Entering Jeronimos Monastery"/>
```

(b)

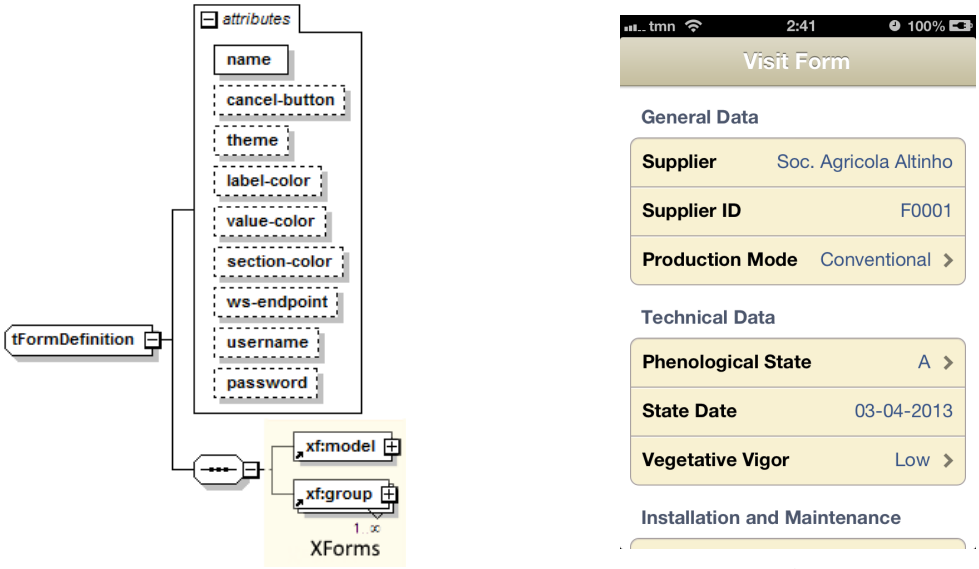
Figure 14: (a) Scenario schema and (b) two scenarios with the same behaviour (same workflow)

In the example of Figure 14b, the application behaviour is the same in both situations corresponding to the execution of the workflow with name `"Entering Jeronimos Monastery"` (lines 3 and 7).

4.5 Forms

IVOML can contain forms compatible with XForms [42]. XForms uses an MVC (Model-View-Controller) approach, which translates into a clear separation between the description of the data used by the form (Model), the user interface (View) and the business rules that define the validation of the data provided (Controller). Figure 15a shows the generic schema of a form which include elements as defined in the

W3C Recommendations for XForms [42]. The interface is described in abstract, leaving the interpreter to define how each element will be presented to the user depending on the platform used. Figure 15b shows an example of an iOS interface of a form for an agricultural application.



(a) (b)

Figure 15: (a) Form schema and (b) an example of a form running on iOS (b)

The attributes of a form allow its global definition and are described in Table 4.

Table 4: Form attributes

Attribute	Description
`name`	Form name.
`cancel-button`	Optional attribute that indicates whether the user can quit without completing the form.
`theme`	Optional attribute that indicates the theme to use in the form which can be `dark` or `light`.
`label-color`	Optional attribute that indicates the color used in labels.
`value-color`	Optional attribute that indicates the color used in values entered by the user.
`section-color`	Optional attribute that indicates the background color used in sections of the form.
`ws-endpoint`	Optional attribute that indicates the web service endpoint used to save the data.
`username`	Username used to access the web service.
`password`	Password used to access the web service.

4.6 Quizzes

IVOML can also contain quizzes defined through the `quiz` element whose schema is illustrated in Figure 16. A quiz may have several questions each with several possible answers of which only one is correct. The attributes of a quiz (shown in Table 5) allow to configure its behaviour.

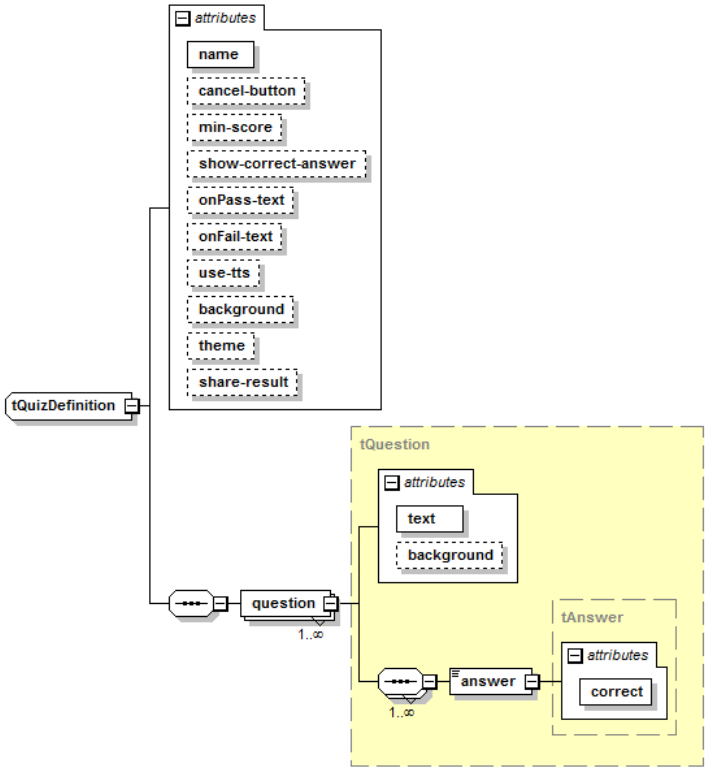


Figure 16: Quiz schema

Table 5: Quiz attributes

Attribute	Description
`name`	Quiz name.
`cancel-button`	Optional attribute that indicates whether the user can quit without completing the quiz.
`min-score`	Optional attribute that indicates the minimum percentage of questions the user must correctly answer so that the workflow continues to the next activity.
`show-correct-answer`	Optional attribute that sets whether feedback should be provided after each response.
`onPass-text`	Optional attribute that indicates the message to show to the user after he successfully answer the quiz (hit the min-score value).
`onFail-text`	Optional attribute that indicates the message to show to the user after he unsuccessfully answer the quiz (hit the min-score value).
`use-tts`	Optional attribute that indicates whether the quiz should be spoken.
`label-color`	Optional attribute that indicates the color used in labels.
`background`	Optional attribute that indicates the image to use as the background of the quiz.
`theme`	Optional attribute that indicates the theme to use in the form which can be dark or light.
`share-result`	Optional attribute that indicates whether the result reached in the quiz can be shared on social networks.

4.7 IVOML Embedded in KML

IVOML is a simple markup language, yet complete enough to fully describe a mobile context-aware application, and it can be embedded into standard representation of geographic information formats such as KML [36] and also GPX [43]. The IVO Builder (described in the Section 3.2.1) allows end-users to easily create geofences as location contexts (Placemark tags of KML) and then use these contexts in IVOML.

Figure 17 illustrates a KML example with IVOML code embedded. In this example, the workflow `"Leaving Office"` is triggered when the user leaves the area defined in the placemark `"Office"` (placemark definition ommitted due to the lack of space) and an SMS is automatically sent to the specified phone number with the specified message.

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns=http://www.opengis.net/kml/2.2
xmlns:ivo=http://www.integratedvirtualoperator.com/ivo
ml/1.0>
  <Document>
    <ExtendedData>
      <ivo:ivoml name="Leaving Office"
                version="1.0">
        <ivo:permission>Internet GPS
        </ivo:permission>
        <ivo:context name="Leaving Office ">
          <ivo:where condition="onExit"
                placemark="Office" />
        </ivo:context>
        <ivo:workflow name="Leaving Office">
          <ivo:sms message="Leaving Office."
                to="..."/>
        </ivo:workflow>
        <ivo:scenario name="Leaving Office"
                context="Leaving Office"
                workflow="Leaving Office" />
      </ivo:ivoml>
    </ExtendedData>

    <Placemark id="Office">
      placemark-definition
    </Placemark>

  </Document>
</kml>
```

Figure 17: KML with IVOML embedded

5 Evaluation

We have conducted formal and informal evaluations, which aimed to identify constraints and test the entire platform to the needs and practices of users. The major goals for those evaluations were: (i) evaluate the ability of the IVOML language to create useful context-aware applications; (ii) evaluate the usability and the ability of the composite tools available in the platform to create those applications; (iii) evaluate the client developed framework and its ability to run the applications created with the composite tools; and (iv) evaluate the user experience when interacting with the platform. The following applications were developed and tested:

1. Leisure and entertainment applications: Ulm tourist guide (Germany), Belem tourist guide (Lisbon, Portugal), Barcelona tourist guide (Spain), Portalegre tourist guide (Portugal), Peddy-paper game and a Wine tourism Application.

2. Business applications: Vineyards management application and a Sales force management application.

The informal evaluations were performed mainly by technicians of a wine company who use the platform on a regular basis. The vineyards management application allows the automation of the process used by the technicians during their visits to the grape suppliers. The majority of the location contexts used in this application are related to the vineyards and were imported to IVO through a KML file generated from the Geographic Information System (GIS) of the company. Other location contexts were created by the agricultural technicians using the polygon tool provided by IVO Builder. The contextual information associated with the vineyards, which includes the acreage of each grape variety and the data gathered by the agricultural technicians during their previous visits to the vineyard, is retrieved from the wine company servers. This application uses a set of forms to collect the data filled in by the technicians when they are in the field (see Figure 15b for an example). The wine tourism department of the company uses another mobile application (also created using the IVO platform) designed to support the activities of the tourists when they visit the cellars and wineries, the restaurant or the historical and natural heritage sites. The formal evaluations were performed by groups of volunteers with different backgrounds, through usability tests at both the end-user level (mobile device) and the developer-user level (builder tools). The test scenarios were designed in order to allow for the testing of most of the features available within the platform. These evaluations were presented in detail in our previous works [15, 16, 17], and only the main conclusions are summarized below.

5.1 Composite Tools

To evaluate the IVO Builder, users were challenged to create an application to guide tourists through points of interest in the city of Lisbon, while for the evaluation of IVO Outlook, users were asked to create contacts and associate them with a Bluetooth device (smartphone) in order to allow activities of type ”nearby people”, and to create appointments to which the user then adds workflows [15].

After performing the usability tests, users were asked to answer a questionnaire that captured personal data, experimental feedback, as well as users' suggestion and comments. Experimental feedback was evaluated through different sections of the questionnaire which included questions regarding (i) the easy of learning, (ii) the ease of use, (iii) how easy it was to perform the proposed tasks, and (iv) one question based on the Microsoft “Product Reaction Cards”, which aimed at capturing the user's feelings when using the system, since they facilitate the measuring of intangible aspects of the user's experience [44].

For the first three sections (ease of learning, ease of use and how easy it was to), users indicated their level of agreement with each statement by circling a value on a 5-point Likert-type scale, with a response of 1 (one) meaning “strongly disagree/very difficult” and a response of 5 (five) meaning “strongly agree/very easy”. Table 6 summarizes the average of responses from all users and the standard deviation for each of these sections while Figure 18 shows a box plot with whiskers with maximum 1.5 IQR (first and third quartile, median, mean) for overall results from the user evaluation.

Table 6: Overall results of the first three sections of the experimental feedback

	IVO Builder	IVO Outlook
Ease of learning	4.0 ± 0.81187	$4.3 \pm 0{,}84017$
Ease of use	3.7 ± 0.98116	$4.4 \pm 0{,}76699$
How easy it was to	4.1 ± 0.90815	4.7 ± 0.21082

As shown in Table 6 and in Figure 18, in general the user's feedback was quite positive with only a few minor issues reported and solved in the next version of the composite tools.

Regarding the fourth part of the questionnaire, based on the Microsoft “Product Reaction Cards”, we concluded that 90% (IVO Builder) and 89% (IVO Outlook) of the participants held positive feelings

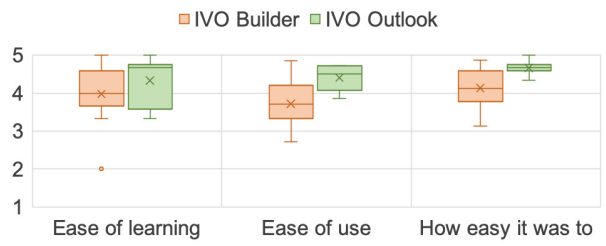


Figure 18: Box plot with whiskers with maximum 1.5 IQR (first and third quartile, median, mean) for overall results from the user evaluation of Composite Tools

when classifying their experience using the system. As shown in Figure 19 the most selected words were "useful", "professional", "simple" and "pleasant".

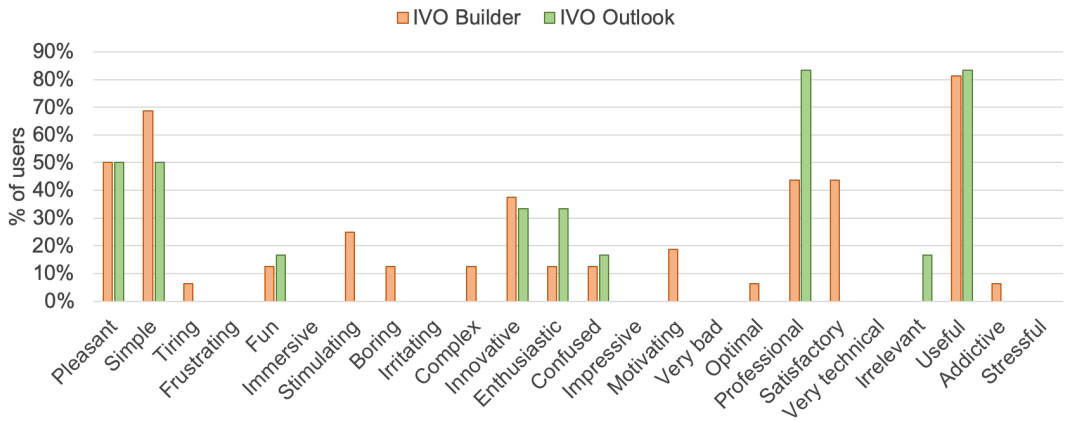


Figure 19: Emotional involvement of Composite Tools

The evaluations concluded that even users who never had contact with the composite tools could easily build context-aware applications using the platform. This evaluation also showed that the visual programming environment facilitates the rapid development of such applications.

5.2 IVO Client

To evaluate the client framework and its ability to run applications created using IVOML, we built two applications with IVO Builder and tested them with several users. We intended to study the applications' usability and to evaluate the system's ability to produce usable and useful applications. The first application was a "peddy-paper" game where users had to pass through four different locations at our Faculty Campus [17], and the second was a mobile tourist guide for the area of Belem in Lisbon [16].

After performing the usability tests, users were asked to answer a questionnaire with the experimental feedback evaluated through different questions regarding (i) usefulness; (ii) the easy of learning, (iii) the ease of use, (iv) how easy it was to perform the proposed tasks, and (v) one question based on the Microsoft "Product Reaction Cards". We use the same approach to collect user answers and to analyse the results, which are summarized in Table 7 while Figure 20 shows a box plot with whiskers with maximum 1.5 IQR (first and third quartile, median, mean) for overall results from the user evaluation. The results show that the interactions were generally deemed easy to perform.

Regarding the fourth part of the questionnaire, based on the Microsoft "Product Reaction Cards",

Table 7: Overall results of the IVO Client evaluations

	Peddy-paper Game	Tourist Guide
Usefulness	not asked	4.4 ± 0.69452
Ease of learning	4.1 ± 0.81931	4.3 ± 0.57735
Ease of use	4.1 ± 0.92243	4.1 ± 0.82075
How easy it was to	4.3 ± 0.83333	4.6 ± 0.62361

we concluded that participants held positive feelings (98% of the total words selected) when classifying their experience using the applications. As shown in Figure 21 the most selected word was "pleasant" followed by the words "simple", "fun" and "useful" on the peddy-paper game, and "pleasant", "stimulating" and "useful" on the tourist guide application. Over 40% of the participants considered the system "innovative".

Both evaluations (peddy-paper game and tourist guide) generated consistent results as shown in Table 7.

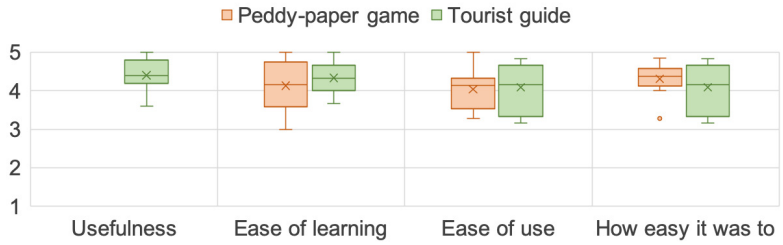


Figure 20: Box plot with whiskers with maximum 1.5 IQR (first and third quartile, median, mean) for overall results from the user evaluation of IVO Client

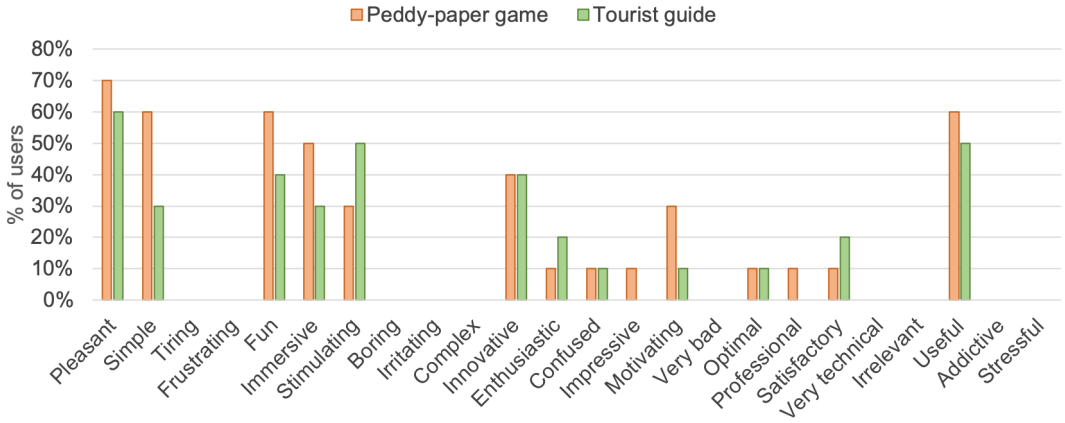


Figure 21: Emotional involvement of IVO CLient

6 Conclusion and Future Work

This paper introduces IVOML, an XML-based language that is the foundation of the IVO platform which enables the creation, deployment and execution of context-aware mobile applications by end-

users without programming skills. The composite tools provided by the IVO platform allow the creation and the deployment of the applications using IVOML. The platform also provides runtimes that allow the execution of such applications on Android and iOS mobile platforms.

The design principles that guided the definition of IVOML were an event-driven workflow model and a scenario-based design.

IVOML and the IVO platform were evaluated through formal and informal tests. Overall, the results were very positive, with users expressing very encouraging feelings, and affirming their willingness to use the system they found useful. The vineyards management application has been in use for about 6 years, which confirms IVO's ability to facilitate the creation of useful, feasible, and robust context-aware applications. This was the first application developed with IVO platform and it has been easily maintained and updated through IVO.

As future work, we intend to improve the building tools with modules for testing and debugging, and conduct experiments with Raspberry Pi [45], Arduino [46], Intel IoT Platform [47] and wearable devices.

Acknowledgments

This work was funded in part by Agência de Inovação (ADI) under contract 70/2007/33B/00216/00178 and by FCT/MCTES NOVA LINCS PEst UID/CEC/04516/2019.

References

[1] M. Manca, F. Paternò, and C. Santoro, "A model-based framework for mobile apps customization through context-dependent rules," *Universal Access in the Information Society*, vol. 18, no. 4, p. 909–925, November 2019.

[2] A. K. Dey, R. Hamid, C. Beckmann, I. Li, and D. Hsu, "a cappella: programming by demonstration of context-aware applications," in *Proc. of the 2004 SIGCHI Conference on Human Factors in Computing Systems (CHI'04), Vienna, Austria*. ACM, April 2004, pp. 33–40.

[3] M. Weiser, "The computer for the 21st century," *Scientific American*, vol. 265, no. 3, pp. 94–104, 1991.

[4] W. N. Schilit, "A system architecture for context-aware mobile computing," Ph.D. dissertation, Columbia University, New York, 1995.

[5] A. Harter, A. Hopper, P. Steggles, A. Ward, and P. Webster, "The anatomy of a context-aware application," *Wireless Networks*, vol. 8, no. 2, pp. 187–197, March 2002.

[6] A. K. Dey, G. D. Abowd, and D. Salber, "A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications," *Human-Computer Interaction*, vol. 16, no. 2, pp. 97–166, December 2001.

[7] Y. Li, J. Hong, and J. Landay, "Topiary: a tool for prototyping location-enhanced applications," in *Proc. of the 17th annual ACM symposium on User interface software and technology (UIST'04), Santa Fe, New Mexico, USA*. ACM, October 2004, pp. 217–226.

[8] A. K. Dey, T. Sohn, S. Streng, and J. Kodama, "iCAP: Interactive prototyping of context-aware applications," in *Proc. of the 4th International Conference on Pervasive Computing (PERVASIVE'06), Dublin, Ireland*, ser. Lecture Notes in Computer Science, vol. 3968. Springer, Berlin, Heidelberg, May 2006, pp. 254–271.

[9] T. Do, K. Cheverst, and I. Gregory, "LoMAK: a framework for generating locative media apps from KML files," in *Proc. of the 2014 ACM SIGCHI symposium on Engineering interactive computing systems (EICS'14), Rome, Italy*. ACM, June 2014, pp. 211–216.

[10] K. Cheverst, T. V. Do, and D. Fitton, "Supporting the mobile in-situ authoring of locative media in rural places: Design and expert evaluation of the smat app," *International Journal of Handheld Computing Research*, vol. 6, no. 1, pp. 1–19, January 2015.

[11] MIT, "APP inventor," http://appinventor.mit.edu [Online; Accessed on July 9, 2019], 2012.

[12] J. Danado and F. Paternò, "Puzzle: A mobile application development environment using a jigsaw metaphor," *Journal of Visual Languages & Computing*, vol. 25, no. 4, pp. 297–315, August 2014.

[13] W3C, "Extensible Markup Language (XML) 1.0 (Fifth Edition)," https://www.w3.org/TR/REC-xml [Online; Accessed on May 28, 2015], 2008.

[14] T. Strang and C. Linnhoff-Popien, "A context modeling survey," in *Proc. of the 1st International Workshop on Advanced Context Modelling, Reasoning And Management (UbiComp'04), Nottingham, UK*, September 2004, pp. 33–40.

[15] V. Realinho, A. E. Dias, and T. Romão, "Testing the usability of a platform for rapid development of mobile context-aware applications," in *Proc. of 13th IFIP TC 13 Internatinoal Conference on Human-Computer Interaction (INTERACT'11), Lisbon, Portugal*, ser. Lecture Notes in Computer Science, vol. 6948. Springer, Berlin, Heidelberg, September 2011, pp. 521–536.

[16] V. Realinho, T. Romão, F. Birra, and A. E. Dias, "Building mobile context-aware applications for leisure and entertainment," in *Proc. of the 8th International Conference on Advances in Computer Entertainment Technology (ACE'11), Lisbon, Portugal*. ACM, November 2011, pp. 1—-8.

[17] V. Realinho, T. Romão, and A. E. Dias, "An event-driven workflow framework to develop context-aware mobile applications," in *Proc. of the 11th International Conference on Mobile and Ubiquitous Multimedia (MUM'12), Ulm, Germany*. ACM, December 2012, pp. 1—-10.

[18] J. Jing, K. Huff, B. Hurwitz, H. Sinha, B. Robinson, and M. Feblowitz, "WHAM: Supporting mobile workforce and applications in workflow environments," in *Proc. of the 10th International Workshop on Research Issues in Data Engineering (RIDE'00), San Diego, California, USA*. IEEE, February 2000, pp. 31–38.

[19] D. Chakraborty and H. Lei, "Pervasive enablement of business processes," in *Proc. of the 2nd IEEE International Conference on Pervasive Computing and Communications (PERCOM'04), Orlando, Florida, USA*. IEEE, March 2004, pp. 87–97.

[20] Y. Cho, J. Han, J. Choi, and C.-W. Yoo, "A uWDL handler for context-aware workflow services in ubiquitous computing environments," in *Proc. of the 2005 International Conference on Embedded and Ubiquitous Computing (EUC'05), Nagasaki, Japan*, ser. Lecture Notes in Computer Science, vol. 3823. Springer, Berlin, Heidelberg, December 2005, pp. 131–140.

[21] J. Jang, A. Fekete, P. Greenfield, and S. Nepal, "An event-driven workflow engine for service-based business systems," in *Proc. of the 10th International Enterprise Distributed Object Computing Conference (EDOC'06), Hong Kong, China*. IEEE, October 2006, pp. 233–242.

[22] IBM, "IBM WebSphere Application Server," https://www.ibm.com/cloud/websphere-application-server [Online; Accessed on December 20, 2018], 1998.

[23] Microsoft Corporation, "Microsoft BizTalk Server," https://docs.microsoft.com/en-us/biztalk [Online; Accessed on December 20, 2018], 2000.

[24] Apache Software Foundation, "Apache ODE (Orchestration Director Engine)," http://ode.apache.org [Online; Accessed on December 20, 2018].

[25] OW2 Consortium, "OW2 Orchestra," https://projects.ow2.org/view/orchestra [Online; Accessed on December 20, 2018], 2012.

[26] Riftsaw, "Riftsaw," http://www.jboss.org/riftsaw [Online; Accessed on December 20, 2018].

[27] OASIS, "OASIS Web Services Business Process Execution Language (WSBPEL) TC," https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsbpel [Online; Accessed on December 20, 2018].

[28] M. Wieland, O. Kopp, D. Nicklas, and F. Leymann, "Towards context-aware workflows," in *Proc. of the 19th Conference on Advance Information Systems Engineering (CAiSE'07), Trondheim, Norway*, ser. Lecture Notes in Computer Science, vol. 4495. Springer, Berlin, Heidelberg, June 2007, pp. 577–591.

[29] L. Ardissono, R. Furnari, A. Goy, G. Petrone, and M. Segnan, "Context-aware workflow management," in *Proc. of the 7th International Conference on Web Engineering (ICWE'07), Como, Italy*, ser. Lecture Notes in Computer Science, vol. 4607. Springer, Berlin, Heidelberg, July 2007, pp. 47–52.

[30] G. Hackmann, M. Haitjema, C. Gill, and G.-C. Roman, "Sliver: A BPEL workflow process execution engine for mobile devices," in *Proc. of the 4th International Conference on Service Oriented Computing (ICSOC'06), Chicago, Illinois, USA*, ser. Lecture Notes in Computer Science, vol. 4294. Springer, Berlin, Heidelberg, December 2006, pp. 503–508.

[31] L. Pajunen and S. Chande, "Developing workflow engine for mobile devices," in *Proc. of the 11th IEEE International Enterprise Distributed Object Computing Conference (EDOC'07), Annapolis, Maryland, USA*. IEEE, October 2007, pp. 279–279.

[32] A. Z. Abbasi, M. U. Ahsan, Z. A. Shaikh, and Z. Nasir, "CAWD: A tool for designing context-aware workflows," in *Proc. of the 2nd International Conference on Software Engineering and Data Mining (SEDM'10), Chengdu, China*. IEEE, June 2010, pp. 128–133.

[33] F. Tang, M. Guo, M. Dong, M. Li, and H. Guan, "Towards context-aware workflow management for ubiquitous computing," in *Proc. of the 2008 International Conference on Embedded Software and Systems (ICESS'08), Sichuan, China*. IEEE, July 2008, pp. 221–228.

[34] U. Dayal, M. Hsu, and R. Ladin, "Organizing long-running activities with triggers and transactions," in *Proc. of the 1990 ACM SIGMOD international conference on Management of data (SIGMOD'90), Atlantic City, New Jersey, USA*. ACM, May 1990, pp. 204–214.

[35] G. Ghiani, M. Manca, and F. Paternò, "Authoring context-dependent cross-device user interfaces based on trigger/action rules," in *Proc. of the 14th International Conference on Mobile and Ubiquitous Multimedia (MUM'15), Linz, Austria*. ACM, November 2015, pp. 313–322.

[36] Open Geospatial Consortium, Inc, "OGC KML," http://www.opengeospatial.org/standards/kml [Online; Accessed on December 20, 2018], 2008.

[37] K. Finkenzeller, *RFID Handbook: Fundamentals and Applications in Contactless Smart Cards and Identification (Third Edition)*. John Wiley & Sons, 2010.

[38] Apache Software Foundation, "Java Expression Language (JEXL)," http://commons.apache.org/jexl [Online; Accessed on December 19, 2018].

[39] C. Potts, "Using schematic scenarios to understand user needs," in *Proc. of the 1st Conference on Designing Interactive Systems: Processes, Practices, Methods, & Techniques (DIS'95), Ann Arbor, Michigan, USA*. ACM, August 1995, pp. 247–256.

[40] J. M. Carroll, "Five reasons for scenario-based design," *Interacting with Computers*, vol. 13, no. 1, pp. 43–60, September 2000.

[41] WfMC, "Workflow Management Coalition (WfMC)."

[42] The Forms Working Group, "XForms 1.1," http://www.w3.org/TR/2009/REC-xforms-20091020 [Online; Accessed on December 19, 2018].

[43] TopoGrafix, "GPX 1.1 Schema Documentation," http://www.topografix.com/GPX/1/1/ [Online; Accessed on December 19, 2018], 2009.

[44] J. Benedek and T. Miner, "Measuring desirability: New methods for evaluating desirability in a usability lab setting," in *Proc. of the 2002 Usability Professional' Association Annual Conference (UPA'02), Orlando, Florida, USA*. Microsoft Corporation, January 2002.

[45] Raspberry Pi, "Raspberry Pi," http://www.raspberrypi.org [Online; Accessed on December 27, 2018].

[46] Arduino, "Arduino," http://www.arduino.cc [Online; Accessed on December 27, 2018].

[47] Intel, "Intel IoT Platform," https://www.intel.com/content/www/us/en/internet-of-things/infographics/iot-platform-infographic.html [Online; Accessed on December 27, 2018].

Author Biography

Valentim Realinho is Adjunct Professor and Deputy Director of the School of Technology and Management of the Polytechnic Institute of Portalegre. He has a Computer Science PhD degree from University of Évora, an MSc in Organization and Information Systems also by University of Évora, a Postgraduate in Management from Catholic University of Lisbon, and a BSc in Informatics Engineering from University of Coimbra. He coordinates the Computing, Design and Marketing Center of C3i (Interdisciplinary Coordination for Research and Innovation) and is an integrated member of VALORIZA - Research Center for the Valorization of Endogenous Resources. As a researcher, he has been involved in several national and European projects related to ubiquitous computing, context-aware computing, Internet of Things and machine learning. Before joining academy, he worked in the private sector, having a vast experience in the development, consultancy and project management of information systems and technology, having been engaged in several projects in collaboration with public and private companies. He was Manager of Andersen Consulting Technology Division where he was responsible or co-responsible for several projects in large companies in the financial and telecommunications sector.

Teresa Romão is an Assistant Professor at NOVA University of Lisbon, where she teaches and develops research work in the area of Human-Computer Interaction. She studied computer science at FCT/UNL and received her PhD degree from the same University in 2001. She is also a member of the research center NOVA LINCS (Multimodal Systems group). She has been coordinating and participating in several national and European research projects related with computer entertainment, augmented reality, mobile storytelling, ubiquitous computing and persuasive technology. She is a member of the Digital Media PhD Scientific Committee in the scope of UT/Austin-Portugal Program. Teresa Romão supervised numerous PhD and MSc thesis already completed and she is currently supervising four PhD students and five MSc students. She has authored and co-authored many publications in books, peer reviewed journals and conferences (CHI, UIST, INTERACT, MobileHCI, ACE). Teresa Romão has been involved in the organization and served on the program committees of various national and top international conferences.

A. Eduardo Dias After finishing his degree in Computer Science, Eduardo obtained his PhD in Contextual Information and Ubiquitous Computing in 1999, and has been conducting leading research since then in the fields of Augmented Reality, Virtual Reality, Mobile Computing, Contextual Awareness, Entertainment and Second Screen Interaction, among others. With over 20 years of academic experience, he has supervised numerous PhD and MSc students, coordinated and participated on many national and international R&D funded projects, and has had his work published in books, journals and renowned international conferences. Since 1992, Eduardo has co-founded eight companies in diverse areas, and countries, ranging in specialisation from hi-tech to sustainable homes. In 2000 he co-founded YDreams, a cutting-edge globally operating IT company which has revolutionised the way people interact with computers. While there, he was a Vice-President until 2008 responsible for many award-winning flagship products and for closing crucial worldwide partnerships with top brands like Nokia, Vodafone, BBC, Navteq, Siemens, Chelsea FC, and Cristiano Ronaldo. During those eight years he supervised sales, business development, entertainment, and human resources, and was co-responsible for YDreams' Corporate Communication from day one. In 2006, Eduardo integrated the

team that negotiated an €8.5 million private equity investment, which catapulted the company's value to €50 million. From 2012 to 2015 Eduardo was the CEO of the bViva Group, which he co-founded with a pool of international shareholders. Currently he is co-founder and CEO of Viva Superstars, a company created by leading digital media experts to revolutionise the way people connect and interact with television shows and live events. Viva Superstars is supporting the leading media companies in Portugal on their digital transformation.