Future Generation Computer Systems 101 (2019) 605–620

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Adaptive interface ecosystems in smart cities control systems

Antonio J. Sánchez [a,*], Sara Rodríguez [b], Fernando de la Prieta [b], Alfonso González [b]

[a] *Informatics and Automation Department, University of Salamanca, Spain*
[b] *University of Salamanca, IoT Digital Innovation Hub, Spain*

H I G H L I G H T S

- Improve citizen experiences in the Smart Cities platforms.
- Application of Adaptive Interface Ecosystems (AIE)
- Graphic tool to evaluate interfaces.
- Solving graphical problems in the IoT user interfaces.

A R T I C L E   I N F O

Article history:
Received 30 January 2019
Received in revised form 17 April 2019
Accepted 20 June 2019
Available online 26 June 2019

Keywords:
Smart cities platforms
User experience
Adaptive interface ecosystems

A B S T R A C T

Recently it is possible to find a lot of studies that propose the improvement of user interfaces in Smart Cities data analysis systems. In general, these proposals tend to take too much care of the representation of data, but they tend to set aside the user experience (UX). New systems try to improve the experiences citizens in Smart Cities platforms, because many of these government platforms are not user-friendly and their quality, in terms of UX, are not up to the task. To solve this problem, both in the interface design and UX, the solution proposed in this article is the application of Adaptive Interface Ecosystems (AIE), a new concept that can be applied to existing interfaces. The advantage of using AIE on developed and in operation interfaces is that it is not necessary to redesign the interface, but rather they will be adjusted (adapting over time) to each user's own way of using and monitoring it. The AIEs give the possibility of perfecting the initial modeling of an interface, collecting and analyzing the segmented data of citizens' interactions. The proposed model has been evaluated by expert users in a real scenario and compared with other existing software with similar features.

1. Introduction

This research work deals with the improvement of user interfaces in Smart Cities data analysis systems. Currently, these control systems of Smart Cities in particular, and data analysis systems in general, tend to take too much care of the representation of data, but they tend to set aside the user experience (UX). In the study de [1] on various state systems, it can be proven that citizens have very diverse experiences in this type of platforms, that many of these government websites are not user-friendly and their quality, in terms of UX, are not up to the task. Because of this, we usually find very arduous and careless graphic interfaces, partly caused by a large number of information sources that usually cover. To solve this problem, both interface design and UX, the solution proposed in this article is the application of Adaptive Interface Ecosystems (AIE) –new kind of adaptive interfaces–, a new concept that can be applied to existing interfaces. The advantage of using these Ecosystems on developed and in operation interfaces is that it is not necessary to redesign the interface, but rather they will be adjusted (adapting over time) to each user's own way of using and monitoring it. The architecture of the AIEs, which work only at "view" level, provides a communication mechanism with the rest of the components of the interface that makes it independent of the data and information represented, with which the use and adaptation of the AIE can go far beyond integration with the interfaces shown here, as can be seen throughout the article, extending to almost any type of interface (regardless of its nature). The AIE also give the possibility of perfecting the initial modeling of an interface, collecting and analyzing the segmented data of citizens' interactions (As it is reflected in Section 3, this data can be used to generate different versions of the interface for production environments as a function of user groups segregated by their behavior).

The difference of the use of AIE to other usability techniques applied to components is that the AIE is a system that can control and consider for its calculations the entire interface and not only the isolated components.

* Corresponding author.
E-mail addresses: anto@usal.es (A.J. Sánchez), srg@usal.es (S. Rodríguez), fer@usal.es (F. de la Prieta), alfonsogb@usal.es (A. González).

https://doi.org/10.1016/j.future.2019.06.029
0167-739X/© 2019 Published by Elsevier B.V.

Usability [2], which is normally a fundamental part in commercial applications, and in which a large part of the financial muscle of business development is invested, is not usually the strong point of Cities Smart's management applications, that are usually covered under the umbrella of universities or open source projects that do not have the necessary funding. Apply the AIE is best answer for this projects, since the AIE offers a decoupled, flexible, easily integrable and low-cost solution.

The analyzed aspects of the data vary greatly from one city to another and however, although it is easy to plan their collection (easy in the sense that generally tends to collect all available data to then elucidate whether they are useful or not), the interest of analysis and visualization changes depending on the needs of each city. Imagine a simple case in which the most critical point in a city is the traffic, from the perspective of municipal management, because often there are numerous traffic jams in the city center and that has led to protests by citizens, while the collection of garbage and public lighting do not present serious problems neither energetic nor of supply. It is clear that the analysis system user will consult more frequently the data associated to the transport and he can hope that these are those that appear first in the main application page. If with the months the priorities changed and garbage collection was his preferred search and he started to consult it recurrently, he would expect that the data associated with the garbage collection would be the first to be displayed. The user behavior and its interests are therefore what determines in these cases the design of the interface (As pointed out [3] y [4]).

In the previous case, the use of AIE would affect the design of the interface, but there may also be situations that affect small components such as the position and placement of filters (see Fig. 1).

From the point of view of the end user of an IoT application for Smart Cities, the case study is similar, each different user has different preferences when consulting the applications provided by the municipal services. We can see, for example, the online application of the city of La Coruña [5], in Spain: in this portal, it is very likely that an urban cyclist what interests him most is information about bicycles in the city. If he uses daily the application, he understands the first things that application must show is the icon that takes him this information, or, most specifically, the map of access points to municipal bicycles which is what he uses the most frequently (see Fig. 2).

Throughout this article, we will analyze the lines of research that have led to the creation of AIE (engines of inferences, pattern recognition, machine learning and gamification) and the techniques and technologies used to apply them in Smart Cities control systems (web interfaces, ontologies and adaptive interfaces). Specifically, point 2 presents an analysis of the evolution of current usability techniques and interface design. The point 3 describes the analytic model: the structure and functioning of an AIE. Points 4 and 5 present the proposed AIE implementation in a real project. In particular, point 4 focuses on the design and implementation of AIE in web environments, where we will see that the design of the AIE is divided into 2 large blocks, one with the analysis model (which corresponds to the mathematical model described in point 3) and another with the connection layer. Point 5 presents the case study by applying the model to a Smart City data management platform called SPECTRA. Point 6 includes conclusions and results from the use of AIE and open challenges for future research.

2. Usability in the 21st century

Usability has been a subject widely studied since the first graphical interfaces were created and there is a lot of bibliography about it. Below is a review of the most relevant events and lines of research.

The fact that user interfaces have been considered as static entities over time, part of the very nature of what has been considered an interface in the world of communications during the last decades, a good example of this consideration is the definition of GUI collected in 'The Linux Information Project' [6]: A graphical user interface (GUI) is a human–computer interface (i.e., a way for humans to interact with computers) that uses windows, icons and menus and which can be manipulated by a mouse (and often to a limited extent by a keyboard as well). Today, the importance of good GUI design is even typified in ISO standards (see the ISO 9241 standard, where the principles of ergonomics are standardized). In 1987 the proposal of the UIDE (User Interface Design Environment) regularized the interface design according to the latest technological advances of the time, modeling the data objects of the applications, actions and pre and post-conditions associated with the actions of the users, but without taking into account the existence of types of users and their behaviors [7], always thinking about the existence of a "unique type of user", applying concepts such as "user-friendly" or "self-explanatory". In most cases today these proposals are still valid, since computer applications have very defined end-user types.

The Special Interest Group on Computer–Human Interaction (SIGCHI) of the Association of Computer Machinery (ACM) in [8] defines the concept of Human–Computer Interaction (HCI) as the design, evolution and implementation of interactive systems for the use human being and the study of the phenomenon that surrounds him. On the other hand, continuing in that line and deepening in what surrounds the communication interface, in [9] concept of interface is defined according to the role it plays in communication: either as access points (1), or as interpreter between the user and their goals (2) or as the person in charge of giving feedback to the user according to their expectations and their mental models (3). With regard to the present article, it is this last meaning (3) that is of most interest for the proposal.

According to [10,11] and [12] the most serious problems that must be solved to meet the user's expectations in the interfaces are:

- The use of menu options with an erroneous order and a confusing separation, inappropriate terminology or iconography that does not conform to the expected meaning.
- Inappropriate and inconsistent use of colors, sounds, icons, and other multimedia content that does not fit the user's cognition.
- Text illegibility due to appearance problems or wrong content.
- Lack of the adaptation of the interface to the tastes of the user and absence of a help system.
- Wasted computational resources.

As you can see, some of these expectations have to deal directly with the needs of the user to adapt the interface to their own concerns (that it is different of their own needs). This type of user could be called a "restless user". The expectations also have to see that with the initial design of the interface and its ability to adapt to the regular activity of a user (which we could call a "conformist user") that has its own activity patterns and to which the interface does not provide a direct solution.

The way to adapt these personal characteristics of the user to an interface, according to [13], is to base its design on the collection of information and knowledge of the user and its transformation into a format that can be stored and later analyzed by an inference engine. Inference engines usually use

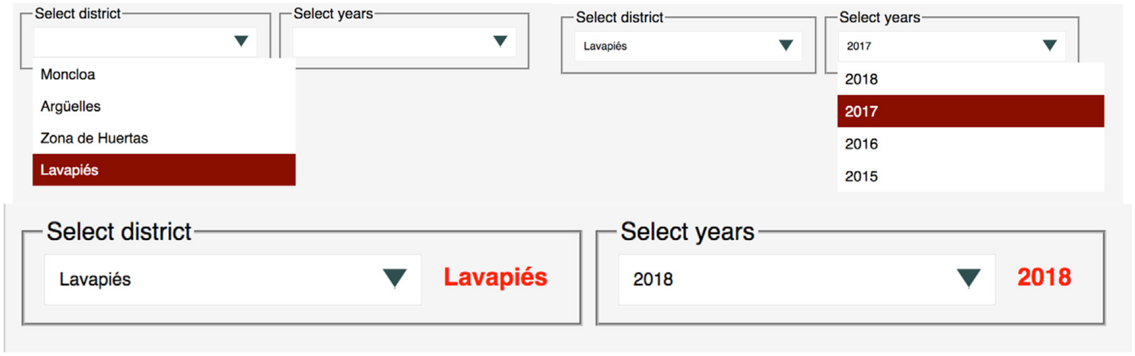


Fig. 1. In this case, the filtering for a user's usual search is for the year 2018 and the neighborhood of Madrid called "Lavapiés", the interface should give a more immediate solution, offering the selection of the filter in a more direct way, like in the last image.

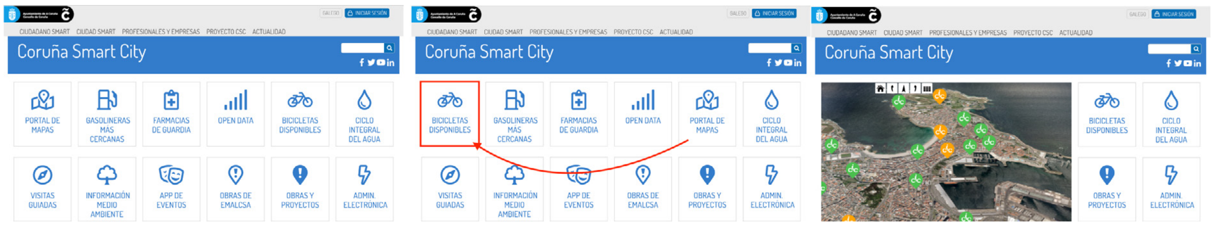


Fig. 2. Example of variation of the interface according to the interaction of an end user.

all kinds of techniques to make their predictions: logical predicates, Bayesian networks, scripts, fuzzy logic, semantic networks, application frameworks, etc. Interfaces based on an inference engine that adapt their appearance to each user is what [14] calls Adaptive Interfaces.

In order to capture user requirements by the inference engine, the user has to interact with the system. In the articles [9,15,16] and [17] are collected different techniques that follow this approach. The technique defined in [18] adds parameters such as location, user identity, time of use and activity as contextual elements for the analysis.

The fundamentals of an intelligent adaptive interface based on recursive fuzzy logic were already proposed last century in [19], where changes are proposed in an interface based on the mental state and the ability to perform user tasks. In this case the equations of the fuzzy logic were used to calculate the level of assistance that should be provided to the user in his tasks. As an example, a virtual reality hockey game was used. The statistical analysis of the experimental results revealed that the proposed adaptive interface was very effective. In this case, the granting of goals during the adaptation of the interface was paramount.

In [20] it is proposed the use of intelligent agents integrated into the system, which was responsible for collecting information and evaluating seven mental faculties of the user and two physical faculties: recognition, planning, action, desire, emotion, memory, language (mental), sensors and actuators (physical). Thanks to that, a subsystem of emotional dialogue is created between the user and the system and, finally, a form of effective dialogue line between them.

In [21], to recognize the emotionality with which the user confronts the resolution of tasks, they compare the effectivity of three types of neural networks – Multilayer Perceptron (MLP), Principal Component Analysis (PCA) and Generalized Feedforward type of Neural Network (GFFNN) – to recognize patterns in the face of the users to capture their emotions (as a result it was concluded that the approach with DCT-MLP1 is the most effective solution).

The proposal of [22] is a variant to the two previous cases, starting from the analysis of the characteristics of the users, to categorize them and group them. The categorization is based on two aspects: the mental models (which collect the information necessary to identify and build the mental structure of the user together with their positive attitude to use the system) and the achievement of objectives (i.e., the personal specifications of primary and secondary objectives that users may have when using the system, which had previously been analyzed in [23]).

Modeling from the objectives of [22] leads directly to the Seaborn and Fels studies, in [24], which mention that gamification, commonly used in games, can be used too, and that is directly related to the achievement of objectives in contexts that are not games. It should be noted that there is a difference between gamification environments (GA) and the feedback Intelligent Tutorial Systems (ITS) that uses motivational messages and confirms the actions of the user.

We can find, even today, methodologies such as TRIDENT (Tools for Interactive Development ENvironmenT) [25] for the generation of user interfaces based on a definition of requirements based on an enriched E/R model and Activity Chaining Graphs (ACG) that provide solutions to the problems of generating interfaces from scratch in a semiautomatic way and that have led to voice-based user interfaces based on grammar for structured reports (see [26]).

Other methodologies such as OVID (Object, View and Interaction Design) [27] started from the principles of the division of the interface the Object–View–Interaction elements to design the interface taking as input the requirements and the analysis of the tasks of the user. OVID generates a structured output that can be easily integrated into the methodologies for design.

TERESA (Transformation Environment for InteRactivE Systems representAtions) is also a semi-automatic solution designed to generate user interfaces for different platforms from task models created using the ConcurTaskTrees (CTT) notation defined in [28] and [29]. CTT is a notation based on the LOTOS language for the analysis and generation of user interfaces from task models.

The generation of an interface based on an UML (Unified Modeling Language) or UML modeling with expanded notation such as UWE [30] and WISDOM (Whitewater Interactive System Development with Object Models), described in [31] and [32] or procedural modeling tools (PMT) (as the process described in [33]) also are options to generate an interface automatically from scratch, but in no case they take account of the adaptability once the interface is up and running.

From the appearance of the agent concept (see [34]), different projects have emerged, especially in ICT (Information and communications technology) environments, in which an interface agent is included as a point of connection between the user and the interface itself (for more information see, for example, the work of [35] or BROA [36]).

In ICT environments we also can find many jobs in which adaptive interfaces have been generated to adjust the repositories of learning objects to the user's capabilities. See for example the works of [37–40] or [41] about the subject.

The most modern trends, as pointed out by [42], are focused on the use of machine learning when configuring the system or around the collection of environmental data by means of sensors to adapt the interfaces, as in the AOP example of [43].

In [44] it is pointed out that the negative feedback obtained (for example, buttons too small or a very low volume in an auditory interface) and unwanted behaviors (for example a menu item that does not take us where we want), obtained from the user experience, must also be taken into account when planning the interface, then all this data must be quantified and analyzed.

As you can see, all the research lines point to the collection and analysis of the parametrizable elements of the user behavior patterns in front of the applications (times, recurrences, etc.) to build intuitive and effective interfaces [42].

On the one hand the gamification techniques are very useful to calibrate the proposals generated as a result of the analysis of these data, and on the other, all the techniques used in the last decade for the definition of interfaces (which comes to summarize [33]) indicate that it is necessary to create an ontology that allows classifying the elements of an interface if you want to work with them to adapt it to the user's behavior. Putting these elements together in a system that provides the interfaces a mechanism to adapt themselves to the behavior of the user (always based on the functional limitations of the interface itself) can be the automated and unattended technical solution for projects where it is not possible to have a team of UX.

3. Structure and functioning of an AIE (analytical model)

Basically, an interface is a contact point between two areas of different mechanics that require communication, the are based on data collectors and providers, translators, after all. But, for this reason, if the mutation and expansion of languages is something usual, why are the interfaces rigid by nature? are they not the ones that modify their processes of capturing and projecting data over time? In nature, the interfaces are modified. A human being is capable of generating body language, gestures or the tone of their conversation depending on who is in front, whether child or adult.

Let us think that a user interface is like a natural habitat, where only the strongest, the most adapted, survive. We also think that, although the designer of any application believes that it is the best for the user, either based on his experience or based on the empirical and statistical evidence that supports his theory, the user is an autonomous being who proceeds, acts and thinks for their own. Adding another point to the theory: the user is part of a user interface, but it is not a part like as a living subject within an ecosystem, but rather it is an atmospheric element, a natural force to which the elements of the system must adapt to strengthen the system itself and become strong themselves over time.

If each element of an interface is considered as an individual that is adapting to a system, it is clear that in its mechanism must have implemented the necessary functionality to recognize the system's events and, with a clear objective, change its way of proceeding to make the system more capable of facing the next event with greater diligence.

This analogy with nature already determines some of the properties that the elements of an EIA must have:

- Interactions. Each event in which the user provides some type of information to the system, either directly or indirectly, is an interaction. In such a way that a domain model similar to that of [45] can be used to store the information.
- Memory. Each component must evolve over time, this implies that it must be aware of its status and note if the mutations produced have positive or negative consequences.
- Communication. Each component must be able to communicate with the other components, since the measures it takes on its own response capacity are meaningless if it cannot establish a scale. For example, a button could not determine if one-click over it is more or less, without knowing if the user makes also clicks over on the rest of the components.
- Limits. Somehow each component must know in what ranges it should be able to be modified (we do not want a text box to flood the whole screen or that a link on privacy policy disappears from a website because no one accesses its content).

Besides, there is a great difference between the components of an interface and the subjects of an ecosystem: while the subjects have as their primary objective their own survival and as second the survival of his species [46], the components, however, must be responsible that the interaction that ends in obtaining data, occurs in the fastest and most comfortable way for the user.

3.1. The mutations

A mutation in nature is "a change in the genetic sequence, and is the main cause of diversity among organisms" [47]. How to transfer that concept to the digital world? The concept of digital mutation is not new [48], we find it, for example, in viruses. The difference between the mutations of viruses and those of the elements of an interface, is that while viruses can change the structure of their code (so that when they replicate themselves is more difficult to be localized them by antivirus) the elements of an interface should modify its external appearance and, when it would be required, also its functionality.

Biological mutations of a species cause side effects: if a species adapt better to the environment then the others species and manages to thrive, it is often usual for it to do so at their expense. In nature this process is self-regulates over time and is predictable thanks to theories of system dynamics such as those described by the Lotka–Volterra equations. Unfortunately, in digital environment self-regulation does not exist, and the limits of mutations should be moderated before the system starts (see the 'Limits' section).

What parameters (visual and functional) of an interface element are susceptible to mutations? Considering the simplest possible categorization between elements of an interface (input elements, output elements, input–output elements and composite elements) the mutation factors can reduce to pregnancy (quality of the visual forms that capture the attention of the observer for its simplicity, balance or stability of its structure) mutations and behavioral mutations. The mutation of behavior must not be confused with the functionality associated with the system interface, but with the form of acting of the interface itself (It is the same difference as form and substance).

In a well-constructed interface, according to [49], users are advancing step by step, from the labeling of the most abstract information to the section whose content is more specific, they are establishing action paths.

To explain a bit better about functionality mutations, we will approach the following example: a user frequently accesses the functionality associated with a second menu level of an application, in fact, it does so more frequently than some of the options arranged in the first level of that menu (that is, it determines an action path). The normal mutation in this case would be the inclusion of the second level menu item within the first level, to the detriment of some of the least used first level menu items. The pre-selection of elements or changes in input data suggestions can also be considered as functionality mutations.

As previously mentioned, mutations of an element of an interface differ in their objective against mutations of nature. The biological mutations have as focus to improve the adaptation of the individual to the environment, taking him to a more favorable position, the digital mutations in this case would not favor the individual but the final user of the interface. This leads us to consider that the mutations will not be 'favorable' in all cases, and that we can find elements that lose excellence in terms of Pregnancy or behavior to favor others.

In order to identify who exactly an element must improve its conditions or when to withdraw from the rest of the elements, it must be aware of the state of those elements, therefore the mechanism of communication between the elements is very important (see the 'Communication' section); but in the same way they must be aware of the environmental conditions, that is, the available resources.

It may seem evident to consider that if an element gains Pregnancy within the interface through a mutation, the rest of the elements must decrease it by an amount in a proportional degree to compensate, so that the total sum of the pregnancies (P) of elements before and after the mutation is equal, so that the system remains stable.

$$\sum P_{t-1} = \sum P_t \tag{1}$$

It would be something similar to the conservation law of energy, but it would not be true in all cases. Although the growth of the Pregnancy of the elements of a system is not infinite, there may be states in which the resources of a system have not been fully expended and the total sum of the pregnancies of the elements is different before and after the mutation. A simple example of this type of status can be the following: consider a software application with menus, in which a second level menu item is again used from a first level and therefore must be 'promoted' to the first level. The expected behavior in this case is that one of the components of the first level becomes the second to counteract the mutation, but this would only be necessary if the total number of first-level elements had reached their peak, otherwise only they would see increased first-level menu items.

Using the menus as an example can lead us to even think that the total pregnancy of a system (considered as the sum of the pregnancy of its elements) can become infinite: we can have an infinite number of menu levels, but that clearly it is not desirable for an interface, because an interface will has infinite functionalities. Therefore, before defining the implementation of an AIE, its inevitable define the limits first.

As a consequence of the evidence of unused resources of a system we can redefine the previous equation in the following way (where Pu is the pregnancy associated with the unused resources):

$$Pu_{t-1} + \sum P_{t-1} = Pu_t + \sum P_t, \text{ where } Pu_{t-1} > Pu_t \tag{2}$$

3.2. Environment

It seems obvious, but it is important to point out that by its nature not all the elements of an interface have similar functions and therefore the changes in the pregnancies of the elements should affect only their direct competitors; that is to say, it would not make sense that, for example, in a web application, the exchange of menu items between different levels affects the size of the text of the headers or the color of the elements of the footer. Therefore, we must consider that the application elements are associated in environments. We can define an environment as the set of elements of an AIE that are affected by the mutation of one of its elements, and thus redefine the pregnancy function:

$$P_{ambient} = \sum P_i + Pu_{ambient} \tag{3}$$

The environments can also be considered as elements of an AIE in such a way that they can mutate and compete with each other if necessary. It does not make sense that the mutation of a menu item affects a text entry box of a form, but that the size of the space assigned to the secondary menu is affected if the size of the main menu increases.

As an element of a system environments can also be grouped together giving rise to new environments. Therefore the pregnancy of a system is determined as the sum of the pregnancies of each of the first level environments (those that are not contained in any other environment) plus the sum of the pregnancy of the available resources:

$$P_{system} = \sum P_{ambient-i} + Pu_{system} \tag{4}$$

3.3. Resources

It only makes sense to define a resource within an environment, since it will be affected by the changes in the elements of the environment. A resource can be either a aesthetic properties (size, brightness, contrast, tone, saturation, position, etc.), for a visual interface, or sound properties (tone, volume, etc.), for a sound interface. A resource within an AIE is in summary a physical property that provides performance to an element of the interface that can be detected by the senses and measurable (the intensity of a umami, for example, can also be considered a resource if we consider it like part of a taste interface element).

3.4. Memory and storage

Like any conscious component within a system, an AIE element must have a memory that allows it to know its own state and environment over time to decide what actions to take each time the user interacts with the interface. The memory must therefore store the data of the resources consumed by elements and environment.

It is not necessary for each element to store the data of its partners in the environment, but it will make easier the calculation; but, as will be seen later, it must store the resources used and available in your environment.

Each component must also store the limits that affect its mutation capabilities.

3.5. The pregnancy mutation

Initially, the interface maker will consider that there are elements of lesser or greater pregnance, since it is impossible to design a system where all the elements are perceived by the user with the same consideration, and only their position makes the user decant towards a certain categorization.

When has an element muted? When should an element grow or decrease? The pregnancy mutation is much easier to calculate than the functionality mutation because it has availabled measurable quantities. In the biological world the minimum mutation in reproduction processes is of a gene, in the digital case it is not necessary that the element reproduces to mutate, but it is necessary to determine what is the minimum quantity of pregnancy, which it can be altered [50] in a mutation.

3.6. The minimum unit of pregnancy

Since the pregnance is not a specific physical quantity but rather a sum of qualities of an object, the identification of a minimum unit is not trivial. A minimum unit of excellence could be a pixel, if it determines the size within a computer screen, or a decibel, if it is to compare sounds; In short, the range of properties that can affect the performance of an object can be very large.

A key factor in determining this minimum unit, that is that the lowest increase or decrease in pregnancy between two elements, can only be determined within a scope where both elements can be considered equal. That area within an AIE will be the environment. Which determines that all the elements within an environment use the same minimum unit of pregnancy, and also the same physical properties.

There is not a form to determine the minimum unit within an environment, the designer/developer must calculate it manually. He must identify the properties that modify the pregnancy of the elements of the environment.

There may be cases in which pregnancy is a composite unit, such as dupla {height in pixels, intensity of red color (0, 255)}, and this can complicate the calculations. Although in these cases it is also possible to establish a minimum unit, such as {1 px, a unit of color}, it is not desirable, although there are studies that allow this type of calculation [50].

3.7. The pregnancy variation

If one element of an interface is more used than another, the ideal is that it pregnancy favor with the rest [51]. Under this base and to be in agreement with Eq. (2), the increase of the pregnancy of an element (P_e) must be equal to the sum of the decreases of the pregnancy of the rest of the elements plus the decrease of the pregnancy not used (associated with resources not consumed):

$$P_e(t+1) = \Delta P_e + P_{e,}(t); \quad \Delta P_e = -\left(\sum \Delta P_i + \Delta Pu_{ambient}\right) \tag{5}$$

So,

$$\Delta P_{ambient} \begin{cases} = 0, & if\ Pu_{ambient} = 0 \\ > 0, & if\ Pu_{ambient} > 0 \end{cases} \tag{6}$$

3.8. The mutation speed

The mutation is not necessarily something that occurs every time there is an interaction with the user. In the biological cycles it takes generations for a species to mutate (although this is fundamentally conditioned because mutations occur during the reproductive process) and although this is not necessarily the same in a digital environment, the truth is that a more contrasted analysis of the conditions of the interface use will possibly lead to a much greater understanding of the user's interaction processes. Therefore, generating an instantaneous mutation every time there is an interaction with the interface is not the most indicated.

Ideally, elements and environments use their own memory to store the user's interactions for a period of time and then making a decision based on perform utilization statistics.

The speed of adaptation of the system will be something that can change over time. The most common is that the interface varies very quickly at the beginning and then passes to long periods of stability, in the same way that happens with biological systems. The learning method of a simple perceptron [52], where the weights vary very quickly at the beginning of their learning can be a very similar model to the one expected.

The concept of initial mutation speed, which we will call v_m, will be very similar to the momentum of a multilayer network and will be a constant in the mutation function.

3.9. The interactions

An interaction, within the AIE, is defined as the event that causes a variation of the pregnancy of an element. The nature of this does not depend on the interface itself, but is directly linked to each element, i.e. each element of the system could have a different interaction associated. For example, in an interface of a software application an input box could have as an event when user make click over it or when the user press a key while the cursor is on it, or both. However, with a map (Google Maps for example) the user interacts in a very different ways can zoom, drag a marker, scroll, etc. And if we consider an interface of sound type, the user can interact with the elements by means of silences or tonal variations in front of the proposals of the system inputs.

On the other hand, the interaction does not always mean a positive variation of the pregnancy, but may have associated a positive or negative reaction on the part of the user. For example, that a user repeatedly clicks on a menu option does not mean that this option is the one that he wants to click, it can also mean that the interface is misconfigured and generates a repeated failure by the user. Therefore, the feedback returned by the user is an element to take into account to determine if a pregnancy increase or decrease occurs. This feedback can be collected directly (if the user indicates to the system that their interaction has been involuntary) or indirectly (if the system collects the user's non-verbal response, such as can be seen in [21,53–55] or so many other works of facial expressions recognition).

The collection of data indirectly should not only be based on the user's own visual characteristics, but also on the interaction itself. For example, think of the user's disaffection that may cause recesses in responses, variations in tone or variation in heart rate.

3.9.1. The mutation function (τ)

The decision as to which elements to mutate or what will be its transformation into minimum units of pregnancy should be taken based on the data stored in the memory about the use of the element by the user. Each element must define what an interaction represents for itself (a click, the completion of a field, the acceptance of a sound order... it will depend on the nature of the element and the type of interface).

The variation of the pregnancy of an element will be determined by the following equation:

$$\Delta P_i = v_m \cdot \tau\left(\frac{i_i}{i_{ambient}}\right), \quad where \begin{cases} i = number\ of\ user\ interactions \\ \tau\ (x)\ is\ the\ environment\ mutation\ function \end{cases} \tag{7}$$

Meanwhile, the diminution of the pregnancy of the rest of the elements will be calculated in the following way:

$$\Delta P_j = -\Delta P_i \cdot \left(\frac{P_j}{P_{ambient} - \Delta P_i}\right) \tag{8}$$

The variation of the pregnancy of the resources not used would be calculated in the same way:

$$\Delta Pu_{ambient} = -\Delta P_i \cdot \left(\frac{P_{ambient}}{P_{ambient} - \Delta P_i}\right) \tag{9}$$

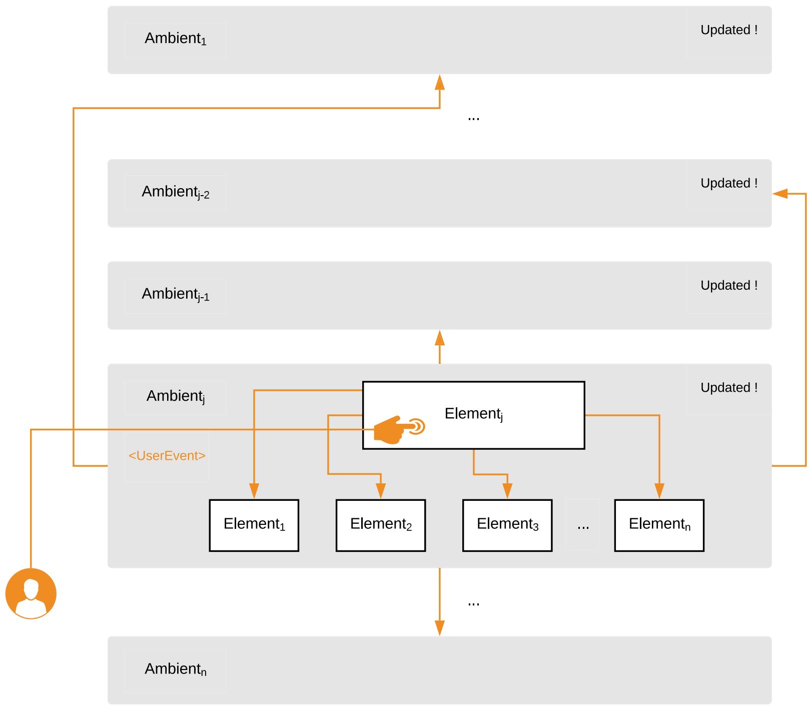


Fig. 3. Interactions notification (1).

3.10. The mutation of functionality

Functionality mutations are easily detectable by humans, but it is difficult to find a global solution that solves all cases. Most of the time functional mutations can be refocused as mutations of pregnancy, reevaluating the guidelines that determine the mutation functions of each environment.

A clear example of functionality mutation is that which would group the result of several calculation processes. This case could be solved if, at the time of planning the functionality associated with the interface, it was decided to include an element that would perform the complete calculation, and this element, little by little, through mutations of excellence, was acquiring a more relevant position in relation to its use.

3.11. Communication and memory

In an AIE, the communication between the elements that form it is providential when it comes to setting the moment and the intensity of the mutations. Each element must have a detailed record of its own interaction with the user and a generalized record of the user's interaction with the rest of the elements of the environment.

In each interaction the element must modify the data about the interactions and the environment stored by it and communicate to the rest of the elements that an interaction has taken place. If the environment to which the element belongs is in turn part of another environment, this last high order environment must perform the same operation. In this way each user interaction will involve a succession of cascading notifications from the element with which it interacts to the elements that make up the 'first level' environment of the system (see Fig. 3).

Once all the elements are updated, we proceed to evaluate if mutations have to occur. This implies that the update process must notify the initiator element that it has finished (see Fig. 4).

The changes caused by a mutation must not affect any element outside the environment from which they happen, whether the resources of the system are being fully exploited or not.

To have control over unused resources within each level, there must be a component that controls them. This component can be either the environment element itself or another isolated component, depending on the implementation.

As it is possible to have interactions in a concurrent way in the system, the ideal is that there is a communication mechanism in which the messages about the interactions and the messages about the mutation processes that take place can be queued. So that some do not interrupt others or the calculations are based on outdated data (see Fig. 5).

3.12. Configuration and limits: AIE and UX design

As mentioned previously, the formalization of limits is a priority task in the design of an AIE. Before the start of operation of an AIE the pregnancy of each element and the available resources, the mutation rate and the mutation function must be defined.

The fact that the elements of the interface mutate freely can come into conflict with aesthetic considerations (aesthetics in their widest range, not only in the visual range, that is the most commonly treated). Therefore, the establishment of the available resources of an AIE does not have to be the same of the system, this must be determined by the interface designer. or the developer The limits of growth and decrease size of the elements of the system must be established also by the designer o the developer, in such a way that the aesthetic aspect is not affected or invades the space of other functionalities of the system to which it does not apply the AIE. This type of decisions, which are of the following type: maximum size of the system input boxes, minimum size of the texts or images or maximum and minimum volumes of the system sounds, are the dimensions that the interface maker must establish when before introducing an AIE system.

The functions of mutation must be defined also thinking that they are a way to limit the appearance of generations in such short periods of time that the user finds it difficult to assimilate the changes. Therefore, mutation functions do not necessarily have to be continuous mathematical functions.

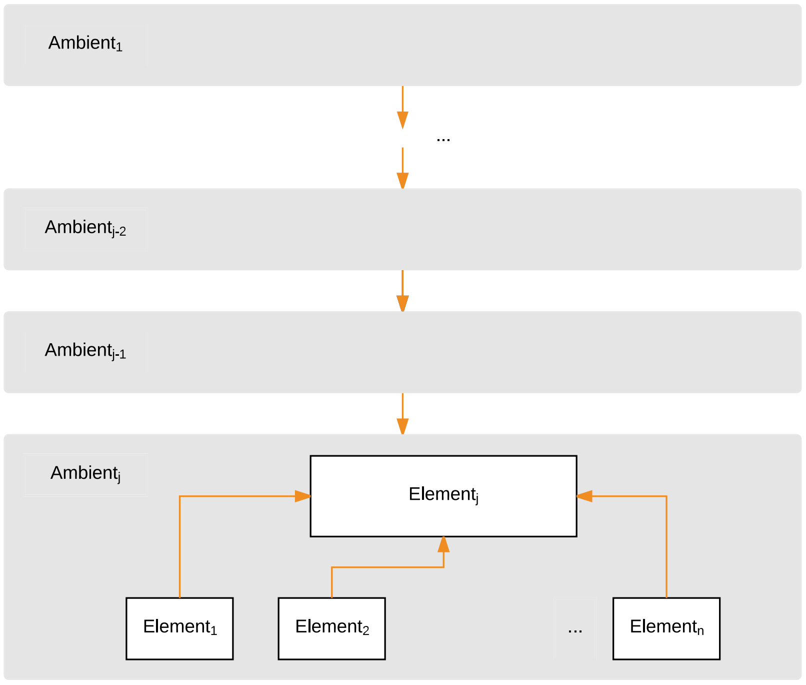


Fig. 4. Interactions notification (2).

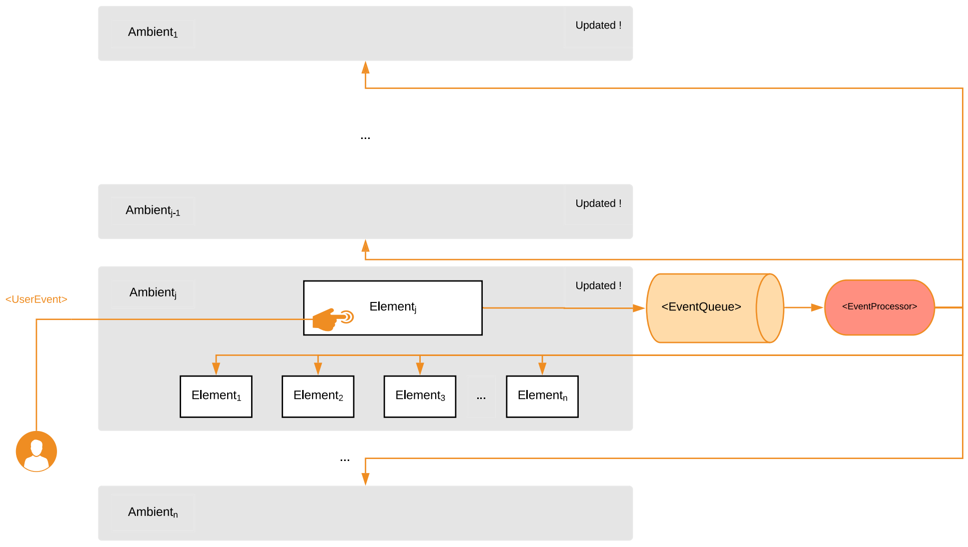


Fig. 5. Queue of events.

3.13. System updates

One doubt that arises when viewing an AIE is whether it should restart its calculations if a new element appears in the system. For example, suppose that, in a mobile application, a new element appears associated with a new functionality, what would happen then with this element? Would it be included in the system with the minimum value for its pregnancy? If it is a new functionality, could be considered that the user would be very interested to know it and therefore should be included in the system with a maximum value for its pregnancy?

It is clear that the initial importance of this new element is not defined by itself or by the system, but rather it is the interface designer who must decide it.

We can find two different cases in which it refers to the inclusion of new elements in the system. In the first case, the element is totally new and has no relation with the rest of the elements of the system (for example a menu item associated with a new functionality that has no primary relationship or associated with the rest of the system's functionalities — imagine an app for mobile e-mail management devices that would allow, as a novelty, to manage the traditional postal delivery and that would have, as a consequence, a new element in the main menu that gave access to that functionality), in this case the designer of the UI must be able to specify the exact place where he must place the menu item.

In the second case, the new element is directly related to one of the elements already present in the system and therefore it can be assumed that the quality of that pregnancy will be the same as that of the element already present.

In both cases, the UI designer must decide what initial pregnancy to assign to the new element, whether an associated one

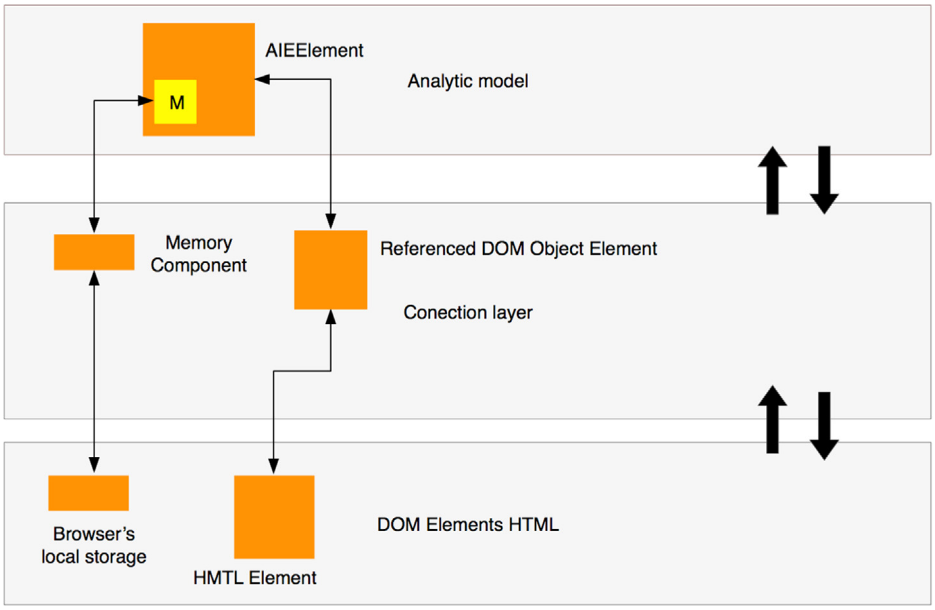


Fig. 6. System layer diagram.

or a new one. The direct assignment of a pregnancy can solve this problem. But it can also be approached from a more technical solution and that is the inclusion of the lifetime of an element in the formula of the calculation of the pregnancy, seeing it thus defined by two parameters:

$$\tau\left(\frac{i_i}{i_{ambient}}, \frac{K_i}{K_{ambiente}}\right) \tag{10}$$

The first parameter, as previously defined, would be the number of interactions of the element with respect to the environment, the second would be the element's life time versus the environment's life time (measured in cycles). This would lead us to introduce a new concept, the update cycle.

Update cycle: Number of times that an element or environment of the system undergoes an update. The memory of the system must take account of these cycles, which increase each time an event occurs within the system that entails a mutation of the UI.

Now the cycles are defined, we have to:

$$\Delta P_i \Delta P_i = v_m \cdot \tau\left(\frac{i_i}{i_{ambient}}, \frac{t_i}{t_{ambient}}\right), \quad where \begin{cases} i = number\ of\ user\ interactions \\ \tau\,(x,t) = environment\ mutation\ function \end{cases} \tag{11}$$

where t_i is the number of cycles of the element and $t_{ambient}$ is the number of cycles of the environment that contains it.

3.14. The maturation function. The end of the mutation

It is important to define when the ecosystem has finished adapting to the user, it is not practical for the ecosystem to be mutating throughout its life cycle, since at a certain moment can be produced anomalous patterns of behavior and that leads the system to evolve to an unintended state (it would be something similar to overfitting the networks of neurons).

With the concept of cycles already introduced, a control function can take charge of limiting the mutation period, as well as increasing or decreasing its effect. We call this function the maturation function (ξ).

A system will remain stable while the mutation is finished and no new elements have appeared that could cause the user to behave differently. Therefore, the parameter that this function must receive is not the number of cycles in the environment, but the number of cycles since the last time an element was inserted in it (or failing that from the beginning of the system). This function, which return to zero every time a new element is introduced, we call the environment maturation degree (m). Then:

$$\Delta P_i \Delta P_i = v_m \cdot \tau\left(\frac{i_i}{i_{ambient}}, \frac{t_i}{t_{ambiente}}\right) \xi(m_{ambient}) \tag{12}$$

4. Implementation of an AIE in a web application

The implementation of an AIE, whatever the environment in which it will be applied, must be divided into 2 layers: on the one hand, the analytical model, explained in Section 3, and on the other hand the procedural model that allows the elements to be connected to the interface (we can call it the connection layer) (see Fig. 6).

The analytical model of the AIE will consist of a components series that allow calculations to be made and compile and store the information provided by the connection layer. These elements will be the Environments, formed in turn by Elements of the environment.

The implementation of the analytical model may change depending on the mechanisms provided by the used language, but its functionality must be the same for any environment. The connection layer however will vary greatly depending on the environment for which it is decided to adapt an AIE. In the case of the presented study, the development of the AIE in JavaScript allows to generate classes for the connection layer through an inheritance mechanism, where the parents are the elements of the analytical model. Its task is very simple because it will limit itself to making calls to the DOM of the browser.

The Environment will be assigned maturation functions and maturation speed functions (both to be defined according to the needs of the user). The Elements of the environments will have associated Pregnancy Calculators, Memories and Event Processors.

The Pregnancy Calculator will be responsible for implementing the mathematical model that adjusts the values of pregnancy (as

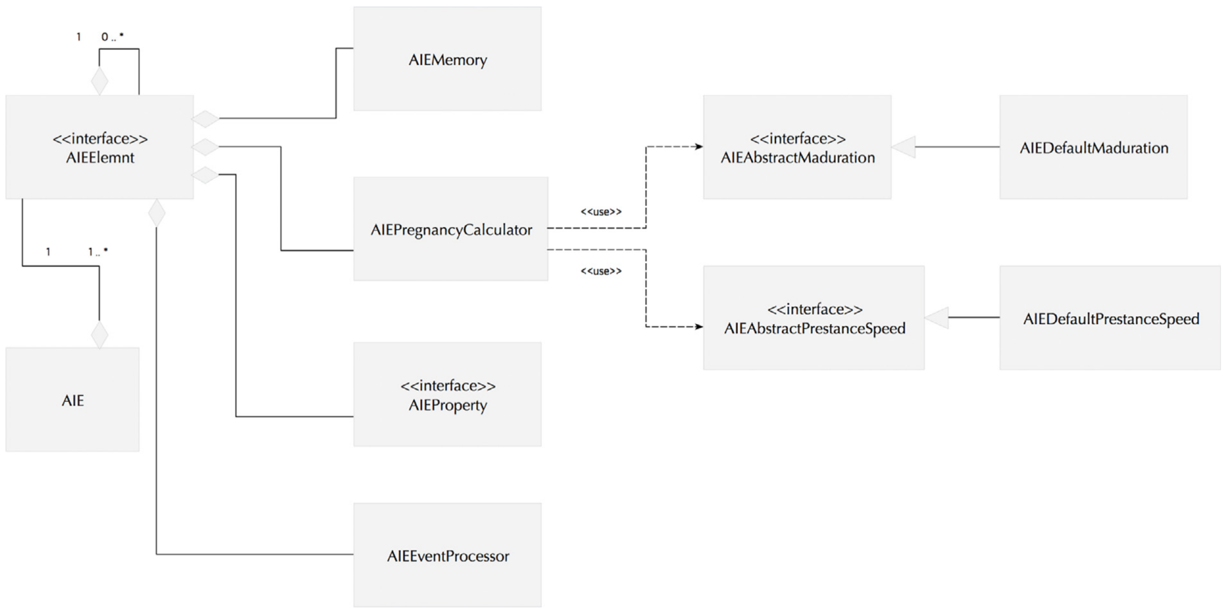


Fig. 7. UML diagram of the classes of the analytical model.

seen in Section 3). To do this, it will use the maturation and maturation speed functions defined in the Environment.

The Environments may have associated one or more elements of environment and the elements of an environment may have associated zero or more elements of environment (which would lead to the formation of sub-environments).

On the other hand, each Environment element (AIEElement) will have associated zero or more Properties, the joint value of these properties is what will determine the physical appearance of the element. The value of these properties will vary together with the pregnancy of the element itself. The Properties are an abstract representation of the physical properties of the elements, and it will be necessary that an element that communicates their changes of state to the elements of the interface (the connection layer would be responsible of it).

A diagram of UML classes has been made, from which a possible implementation is derived, shown below (Fig. 7), where these elements are modeled and the relationships between them. In the diagram we have composed a class structure where AIEElement represents an interface element, AIE the higher level environment, AIEMemory and the component associated with the management of the memory, AIEPregnancyCalculator would perform the calculation functions of pregnancy and AIEEventProcesor would be responsible to manage the user's actions on the components.

When adapting an AIE to an HTML browser's interface, we must bear in mind that the elements of the interface in one way or another must have associated HTML elements. These elements can be described directly in the language itself or in similar languages that need to be pre-compiled, such as JSX, or generated dynamically in JavaScript. Regardless of how they are generated, it will be necessary to determine which elements of the interface are part of the AIE (because they do not necessarily have to be all of them, for example, a page footer or a notification may have to remain in a web should not be affected them for different reasons, such as a legal restriction).

One of the advantages of working with HTML is that the elements are easily labeled (it is a property of the own language). Therefore, it is easy to add a series of attributes to the tags that do not fall within the HTML standard and therefore do not generate collisions or dependencies with other page elements or scripts.

When carrying out the implementation of the AEI has chosen to add to the original attributes of HTML the following this new attributes (all contain the prefix aie-, to be easily identifiable when reading the code) (see Table 1):

All elements must be defined (they do not matter that they are hidden) in the DOM (Document Object Model) of the own document when invoking the initial directives of the AIE library. If new elements were added to the interface in a dynamic way, the system would have rebooted (this possibility exists and is valid, persistence of the values of the elements that are on the screen, but requires extra processing).

The connection layer serves to link the model with the elements of the interface that have been marked. In this way it allows the model to adapt to different formulas for the generation of web elements (for example, you can create different connection layers for the most used programming frameworks such as React (see [56]), which works with JSX, or Angular (see [57]), which uses ng-templates, without the need to reimplement the model) (see Fig. 8).

The connection layer is formed, as already advanced, by a Property class and it is the responsible for modifying the DOM and that will be supported by a Measure class (which will solve the calculations with the different types of browser measures like px, em or "%"), a Memory class that will resolve where to store the AIE data inside the browser (by default in the localStorage), an Element class that will work as a link between the Element of the model and the DOM element, and finally a Environment class that allows to apply an abstract factory pattern to the design to make it more usable.

Taking into account the described implementation process and the defined elements of the model, point 5 describes how to has made the implementation for SPECTRA platform, a Smart City management project. The project fulfills the premises to be the perfect project for the use of an AIE: it is a platform for IoT in development, whose budget does not propose the inclusion in the development team of a designer or expert in UX, for which it will be in AIE which is in charge of automating the design of the interface according to the preferences of the user.

5. Case study. The SPECTRA platform

The SPECTRA platform (Smart PErsonal Co2-free TRAnsport), is a project co-financed by the CDTI and the European Regional

Table 1
HTML attributes.

Attribute	Value	Description
aie	(vacio)	Identify what an aie element
aie-name	ID Example: Aie-name = "filter"	Unique identifier of the element
aie-prestance-fields	PROPERTY|max:MAX_VALUE|min:MIN_VALUE, Example: aie-prestance-fields = "font-size|max:1.1|min 0.5"	NName of the properties and maximum and minimum values that can be reached (in as much as 1). PROPERTY is a string and MAX_VALUE and MIN_VALUE are decimal numbers. Valid identifiers of PROPERTY have the same name as the CSS properties they represent, plus some values that have no reference within CSS, such as "level" (which determines the depth level of an element in a menu or element tree)
aie-trigger	EVENT_NAME Example: aie-trigger = "click"	Name of the event that indicates one that the user's interaction must be collected for the calculation of performance. They are the same event names that are used in JavaScript.

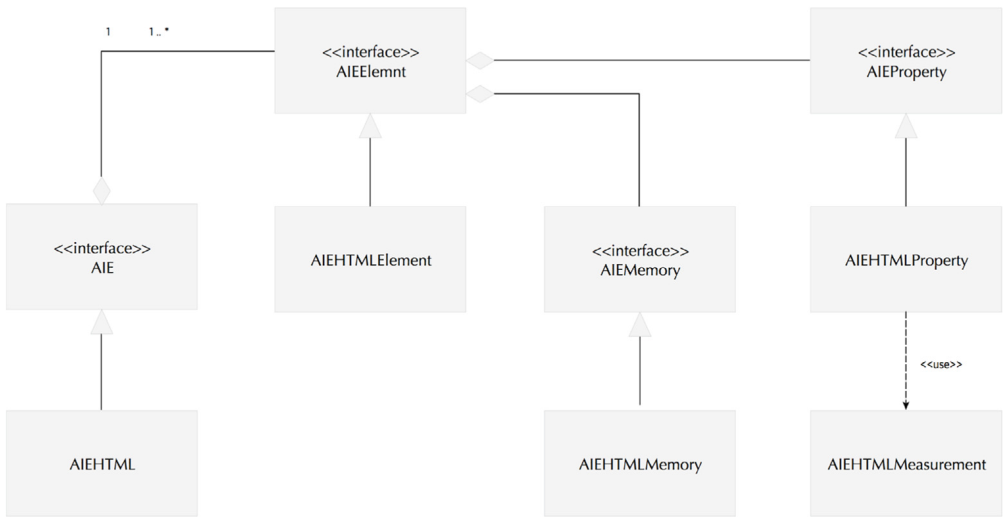


Fig. 8. UML diagram of the connection layer elements and their relationship with the classes of the analytical model.

Development Fund (ERDF), developed by a consortium made up of 8 companies and supported by 10 other organizations , including the University of Salamanca, whose main objective is to achieve an improvement in urban mobility by reducing congestion and the impact on the environment. One of the solutions provided by this project is a platform for intelligent traffic control that obtains information from public databases to, among other things, improve the traffic congestion that occurs at rush hour when regulating the traffic lights of a certain area. The traffic control platform has a web application on which the use of AIE has been applied.

Fig. 9 shows how one of the screens of the application where the ozone in the air of the city of Santander is showed.

For the configuration of the AIEs in the interface of the SPECTRA web application, it has only been necessary to link the JavaScript file with the library and label the sections and components that wish to be analyzed. Here are some examples of labeling:

<div id = "ozon_latitude" aie aie-name = "graph_1" aie-trigger = "mouseover, click">

<label class = "submenu_item" for = "chk_sb_8" aie aie-name = "submenu_8" aie-trigger = "click">

<div class = "submenu_content" aie aie-name = "submenu_items" aie-prestance-fields = "position">

This labeling will determine the sub-environments in which the screen of the application has been divided. In the following image you can see how they have been defined. The HTML estaments define what elements are inside each sub-environment (see Fig. 10).

To have even more clear control of how the system has been finally configured, a widget has been created specifically for the browser, which connects directly to the AIE library through the browser's APIs (see [58]), which allows see not only the initial configuration, but also the values of the pregnancies of the elements each time the user interacts with any of them (see Fig. 11).

The user of the test, for this case study, is interested in consulting the contents of the ozone section on a daily basis, with a special predilection for the graphs of the lower sections that show the ozone content in the air based on latitude and longitude. For this reason its usual procedure is to click on the main menu and select the "Medio ambiente" section, then in the secondary section "Ozono", and finally interact with the graphics that interest him.

For each click or mouse movement performed by the user, the AIE records the interaction and recalculates the pregnancy of the elements according to Eq. (12). The physical changes in the elements observed on the screen do not apply immediately, but they remain saved within the JavaScript classes of the model elements (AIEElement) and persist in the browser memory thanks to the Memory class associated with each element(AIEMemory).

The browser widget also provides hitmap with the user's interactions (Fig. 12). Each element, depending on its nature, has been assigned an event that the AIE must inspect: the menus and sub-menus control by the user's click, while the graphs control the time the mouse moves over and the number of clicks.

After the data collection, once the mutations are applied to the interface (for this test the mutation was released manually,

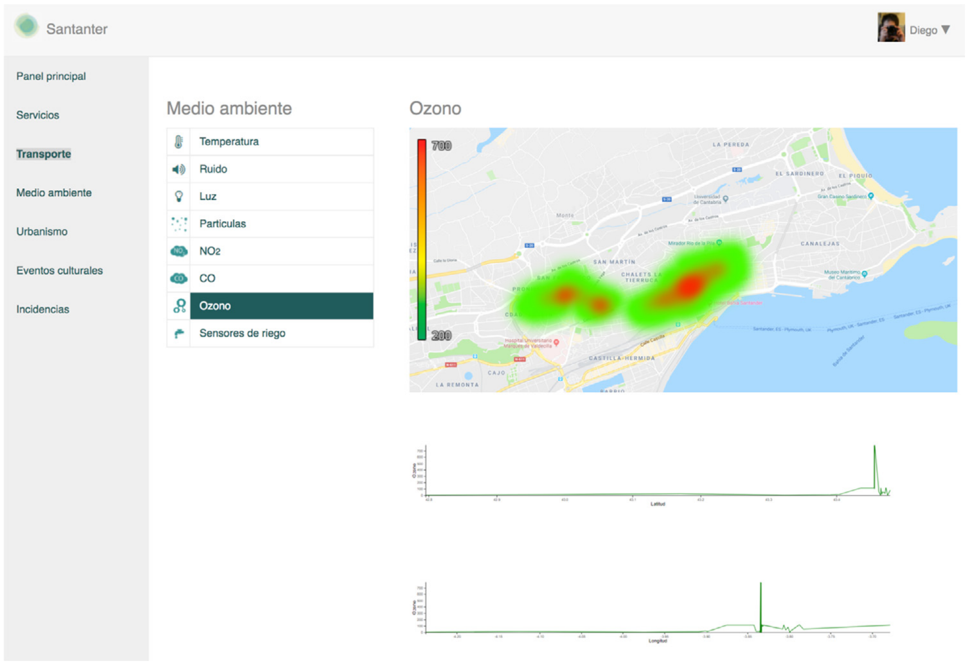


Fig. 9. Screen with information about ozone in the atmosphere of SPECTRA.

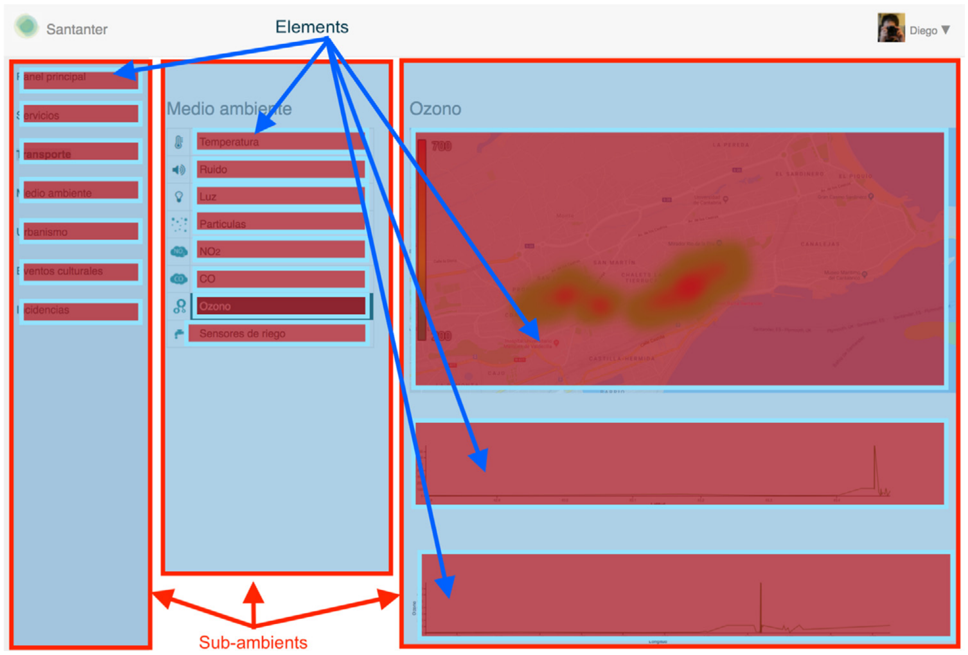


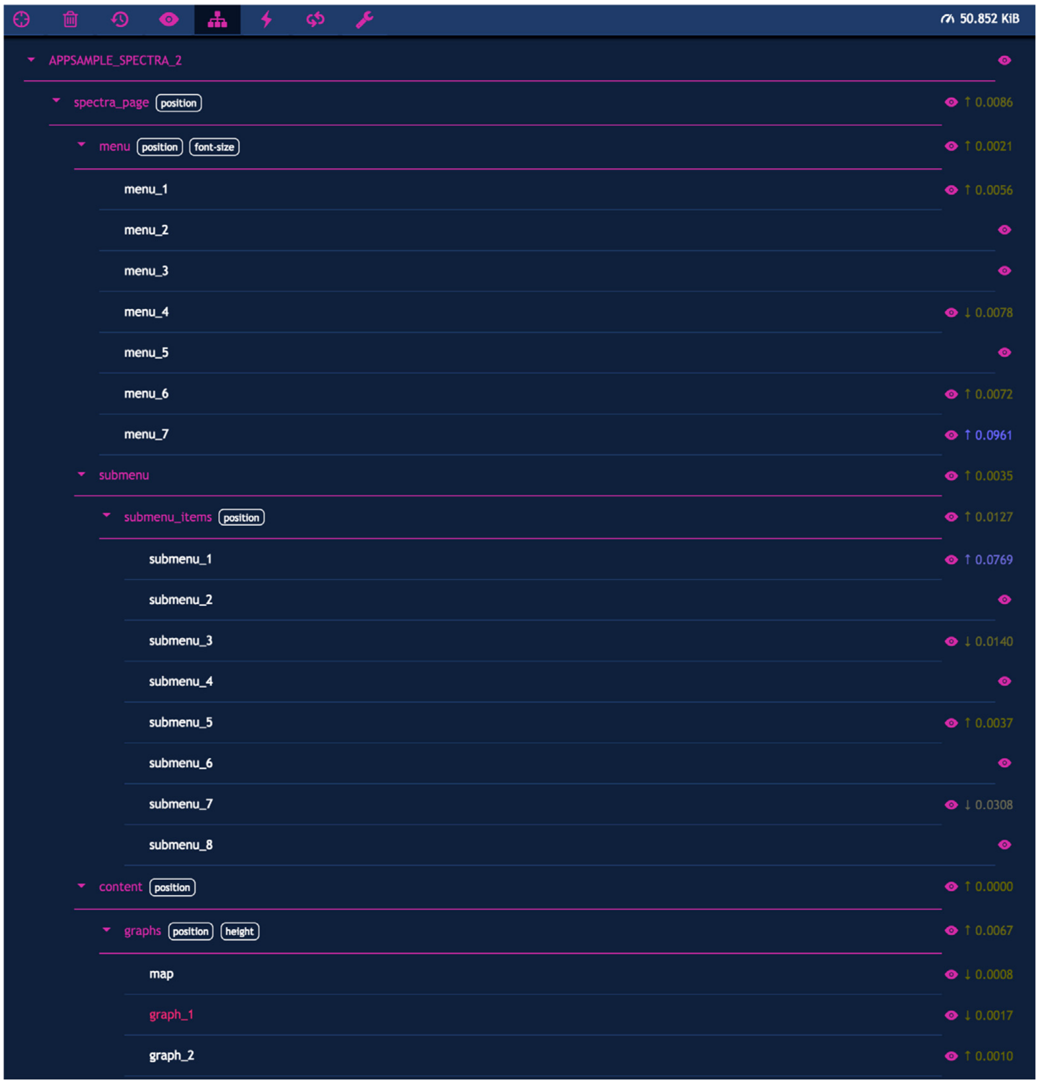
Fig. 10. Definition of sub-environments in the example.

through an action of the browser widget), it can be observed how the elements with which the user interacts most frequently have been placed on the left side, and that the size of the texts of the buttons that it uses the most has increased slightly (perhaps where it is best appreciated is in the main menu in the item "Medio ambiente") and these, in addition, have been moved to the top (see Fig. 13).

The request for the mutation, although in this case study has been made manually, can be scheduled to run when the analysed user patterns begin to be recurrent (since it makes no sense to apply changes when the user is, for example, testing the application), but in this article will not enter to define or calculate those times.

6. Conclusions and future lines of research

As we have seen in the example, the use of this approach significantly reduces the work of a UX/UI designer and also improves the maintainability of an interface, since it is able to adapt itself



```
50.852 KiB
APPSAMPLE_SPECTRA_2
  spectra_page [position]                          ↑ 0.0086
    menu [position] [font-size]                    ↑ 0.0021
      menu_1                                       ↑ 0.0056
      menu_2
      menu_3
      menu_4                                       ↓ 0.0078
      menu_5
      menu_6                                       ↑ 0.0072
      menu_7                                       ↑ 0.0961
    submenu                                        ↑ 0.0035
      submenu_items [position]                     ↑ 0.0127
        submenu_1                                  ↑ 0.0769
        submenu_2
        submenu_3                                  ↓ 0.0140
        submenu_4
        submenu_5                                  ↑ 0.0037
        submenu_6
        submenu_7                                  ↓ 0.0308
        submenu_8
    content [position]                             ↑ 0.0000
      graphs [position] [height]                   ↑ 0.0067
        map                                        ↓ 0.0008
        graph_1                                    ↓ 0.0017
        graph_2                                    ↑ 0.0010
```

Fig. 11. Structure of the AIE, properties that are modified within each environment and values of the pregnancies in real time seen from the browser widget.

to the users individually, because it is not necessary that all users would share the interface evolution data.

It is interesting to know how and to what degree the data obtained from a non-homogeneous set of users could help in making an EIA evolve more quickly.

The AIE are applicable to small environments or applications, but it is discouraged that they should remain constantly in operation once the system has entered a advanced production phase, since it carries a processing, storage and computation load of the extra data that with the time and the use of the system is increasing. Therefore, the ideal is to apply only the development and test phases and in the first steps of production.

One of the key points that remains to be determined is knowing when a system has collected enough data to interrupt the use of the AIE and readjust the interface. It is clear that when ΔP_i obtains values close to 0, applying the maturation function, it will be mature enough to be able to start walking alone. But constant changes in the interface in a test environment can greatly hinder the work of the test user and therefore it can be useful to program and notify the mutations, so as not to surprise the user....

If we start from the initial description of the most common errors in [10,11] and [12], the AIE gives an automated solution to the following cases:

- The use of menu options with an erroneous order and a confusing separation. (1)
- Inappropriate and inconsistent use of colors, sounds, icons, and other multimedia content that does not fit the user's cognition (2)
- Illegibility of text due to appearance problems. (3)
- Lack of adaptive property of the interface to the user's tastes. (4)
- Wasting of the computational resources associated with the creation of interfaces. (5)

Leaving aside the sections of:

- Inappropriate terminology or iconography. (1)
- Wrong content of the texts. (2)
- Absence of aid system. (4)

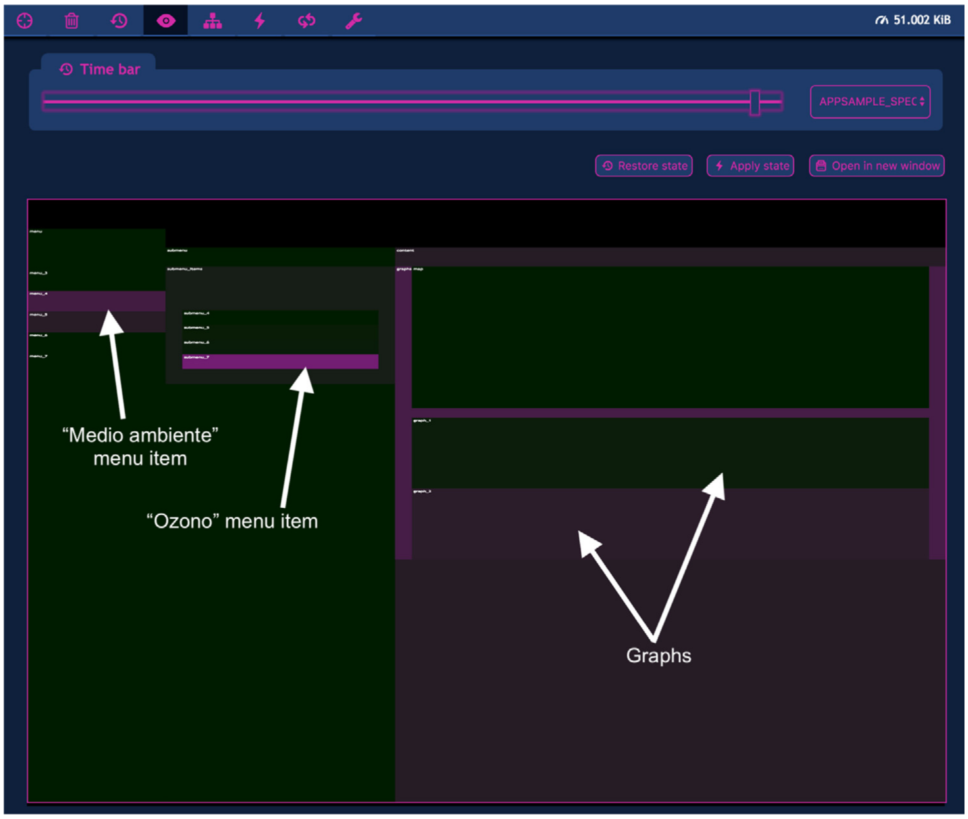


Fig. 12. Hitmap with user interaction data.

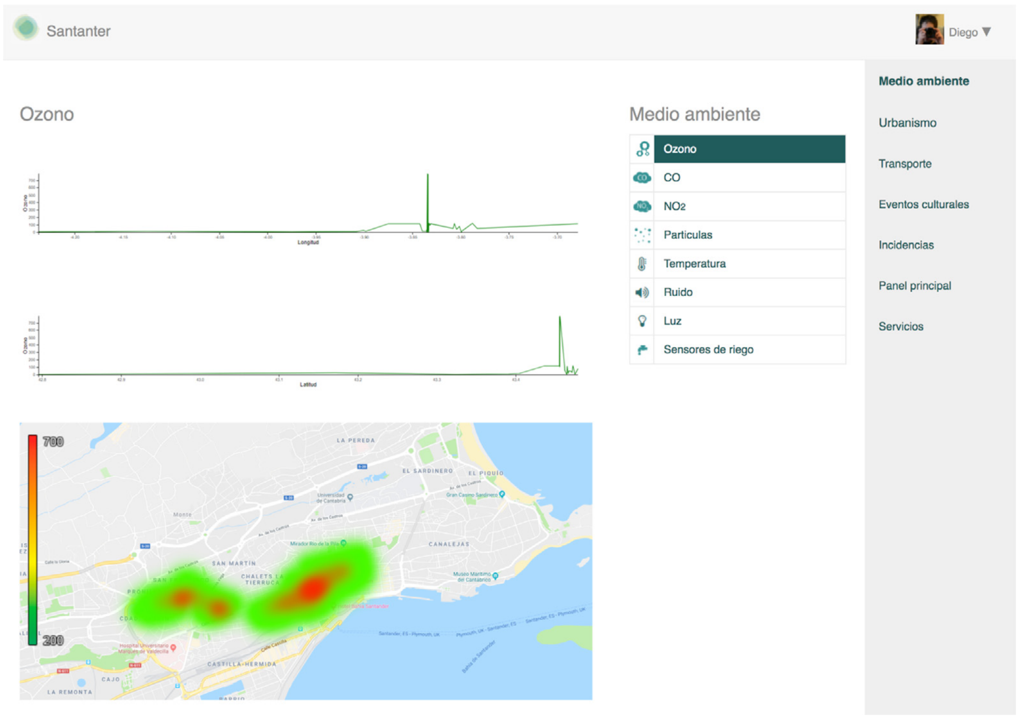


Fig. 13. Final status of the application after the mutation.

In comparison with these other interface systems the results, in terms of problem solving, are the following:

	AIE	TRIDEN	PMT	AOP
(1)	Partial	No	OK	No
(2)	Partial	No	OK	OK
(3)	OK	No	OK	No
(4)	Partial	Partial	No	No
(5)	OK	No	No	OK

One of the points to improve, already identified in [59], is the forecast of a model that optimizes the resources used to meet the performance requirements and energy consumption in devices with very limited resources in IoT environments.

Acknowledgments

This work has been supported by project SPECTRA (Smart PErsonal Co2-free TRAnsport), Spain, co-financed by the CDTI and the European Regional Development Fund (ERDF), Spain. IDI-20150664

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Rajiv Kumar, Amit Sachan, Arindam Mukherjee, Qualitative approach to determine user experience of e-government services, Cit. Data Comput. Hum. Behav. (ISSN: 0747-5632) 71 (2017) 299–306, http://dx.doi.org/10.1016/j.chb.2017.02.023.

[2] G. Cockton, D. Lavery, A. Woolrychn, Inspection-based evaluations, in: J.A. Jacko, A. Sears (Eds.), The Human-Computer Interaction Handbook, second ed., Lawrence Erlbaum Associates, ISBN: 0-8058-3838-4, 2003, pp. 1171–1190.

[3] Eugene Agichtein, Eric Brill, Susan Dumais, Improving web search ranking by incorporating user behavior information, in: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '06), ACM, New York, NY, USA, 2006, pp. 19–26, http://dx.doi.org/10.1145/1148170.1148177.

[4] Christoph Evers, Romy Kniewel, Kurt Geihs, Ludger Schmidt, The user in the loop: Enabling user participation for self-adaptive applications, Future Gener. Comput. Syst. 34 (2014) 110–123, http://dx.doi.org/10.1016/j.future.2013.12.010.

[5] F.H. Priano, C.F. Guerra, Fully smart cities: Analysis of the city to smart city transformation process, in: 2016 IEEE International Smart Cities Conference (ISC2), Trento, 2016, pp. 1–8, http://dx.doi.org/10.1109/ISC2.2016.7580745.

[6] GUI definition. Linux Information Project. October 1, 2004. Retrieved 12 2008. http://www.linfo.org/gui.html.

[7] J. Foley, A. van Dam, S. Feiner, J. Hughes, Computer Graphics: Principles and Practice, Addison-Wesley, Reading, MA, 1990, ISBN-13: 978 − 0 − 321 − 39952 − 6.

[8] C. Rusu, V. Rusu, S. Roncagliolo, Usability practice: The appealing way to HCI, in: 1st International Conference on Advances in Computer Human Interaction, 2008, pp. 265–270, http://dx.doi.org/10.1109/ACHI.2008.14.

[9] Y. Kawahara, H. Morikawa, T. Aoyama, Intelligent interface systems utilizing user context-awareness, in: Sixth International Conferences on Machine Learning and Cybernetics, Hong Kong, 2007, pp. 2119–2124.

[10] Ehlert, A.M. Patric, Intelligent User Interface: Introduction and Survey. Research Report DKS03-01 / ICE 01, Version 0.9, Feb. 2003.

[11] S. Zuffi, C. Brambilla, G. Beretta, P. Scala, Human computer interaction: Legibility and contrast, in: 14th International Conference on Image Analysis and Processing (ICIAP 2007), IEEE Computer Society, Modena, Italy, ISBN: 0-7695-2877-5, 2007, pp. 241–246.

[12] J.M. Carroll, Human-Computer Interaction, Second Impression, Pearson Education, 2008, https://www.nature.com/scitable/topicpage/genetic-mutation-1127.

[13] Stephen Greene, Jason Finnegan, Usability of mobile devices and intelligently adapting to a user's needs, in: Proceedings of the 1st International Symposium on Information and Communication Technologies (ISICT '03), Trinity College Dublin, 2003, pp. 175–180.

[14] E. Ross, Intelligent User Interfaces: Survey and Research Directions, 2000, http://www.cs.bris.ac.uk.

[15] N. Ozkan, C. Paris, B. Simpsom, Towards an approach for novel design, in: S.A. Adelaide (Ed.), Australasian Computer Human Interaction Conference, Australia, 1998, pp. 186–191.

[16] K. Yoshida, User commands prediction by using graph- based induction, in: Sixth International Conference on Tools with Artificial Intelligence, New Orleans, LA, USA, 1994, pp. 732–735.

[17] B.A. Goodman, D.J. Litman, Plan recognition for intelligent interfaces, in: 6th IEEE Conference on AI Applications (CAIA-90), Santa Barbara, 1990, pp. 297–303.

[18] D. Sanchez, M. Tentori, J. Favela, Hidden Markov model for activity recognition in ambient intelligence environment, in: 8th Mexican International Conference on Current Trends in Computer Science, 2007, pp. 33–40.

[19] F. Arai, T. Fukuda, Y. Yamamoto, T. Naito, T. Matsui, Interactive adaptive interface using recursive Fuzzy reasoning, in: Proceedings of IEEE Virtual Reality Annual International Symposium, VR '93, Seattle, Washington, USA, 1993, pp. 104–110.

[20] N. Okada, K. Inui, M. Tokuhisa, Towards affective integration of vision, behavior and speech processing, in: Proceedings of the Integration of Speech and Image Understanding, 1999, pp. 49–77.

[21] G.U. Kharat, S.V. Dudul, Neural network classifier for human emotion recognition from facial expressions using discrete cosine transform, in: Emerging Trends in Engineering and Technology, 2008. ICETET'08. First International Conference on, IEEE, 2008, pp. 653–658, http://dx.doi.org/10.1109/ICETET.2008.22.

[22] K. Leichtenstern, E. Andre, User centered development of mobile interfaces to a pervasive computing environment, in: First International Conference on Advances in Computer Human Interaction, 2008, pp. 114–119.

[23] A. Dillon, C. Watson, User analysis in HCI - the historical lessons from individual differences research, Int. J. Hum. Comput. Stud. 45 (6) (1996) 619–637.

[24] Katie Seaborn, Deborah Fels, Gamification in theory and action: A survey, Int. J. Hum.-Comput. Stud. 74 (2014) http://dx.doi.org/10.1016/j.ijhcs.2014.09.006.

[25] F. Bodart, A.M. Hennebert, J.M. Leheureux, I. Provot, B. Sacre, J. Vanderdonckt, Towards a systematic building of software architectures: the trident methodological guide, in: Ph. Palanque, R. Bastide (Eds.), Proc. of 2nd Eurographics Workshop on Design, Specification, Verification of Interactive Systems DSV-IS'95 (Toulouse, 7-9 Juin 1995), Springer-Verlag, Vienne, 1995, pp. 262–278.

[26] J. von Berg, A grammar-based speech user interface generator for structured reporting, Cit. Data Int. Congr. Ser. 1256 (C) (2003) 887–892, http://dx.doi.org/10.1016/s0531-5131(03)00391-1, ISSN: 0531-5131.

[27] D. Roberts, D. Berry, S. Isensee y J. Mullaly, Designing for the User with OVID: Bridging User Interface Design and Software Engineering, New Riders Publishing, 1998.

[28] F. Paternò, Model-Based Design and Evaluation of Interactive Applications, Springer, 1999.

[29] G. Mori, F. Paternò, C. Santoro, CTTE: Support for developing and analyzing task models for interactive system design, IEEE Trans. Softw. Eng. (2002) 797–813.

[30] N. Koch, Software Engineering for Adaptive Hypermedia Systems: Reference Model, Modelling Techniques and Development Process (Ph.D. thesis), Ludwig-Maximilians-Universität Munchen, 2001.

[31] N. Nunes, J. Cunha, Wisdom: a UML based architecture for interactive systems, in: Ph. Palanque, F. Paternò (Eds.), Interactive Systems: Design, Specification, and Verification (7th International Workshop DSV-IS, Limerick, Ireland, June, 2000), in: LNCS, vol. 1946, Springer, 2000.

[32] N. Nunes, Object Modeling for User-Centered Development and User Interface Design: The Wisdom Approach (Tesis doctoral), Universidad de Madeira. Abril, 2001.

[33] Pierre Taner Kirisci, Klaus-Dieter Thoben, A method for designing physical user interfaces for intelligent production environments, Adv. Hum.-Comput. Interact. 2018 (2018) 6487070, http://dx.doi.org/10.1155/2018/6487070, 21 pages.

[34] M. Wooldridge, N.R. Jennings, Agent theories, architectures, and languages: A survey, in: M.J. Wooldridge, N.R. Jennings (Eds.), Proc. ECAI-Workshop on Agent Theories. Architectures and Languages, Amsterdam, The Netherlands, 1994, pp. 1–32.

[35] P. Rodríguez, V. Tabares, N. Duque, D. Ovalle, R. Vicari, BROA: An agent-based model to recommend relevant learning objects from repository federations adapted to learner profile, Int. J. Interact. Multimedia Artif. Intell. 2 (1) (2013) 6–11.

[36] O. Salazar, P. Rodríguez, N. Ovalle, Interfaces adaptativas personalizadas para brindar recomendaciones en repositorios de objetos de aprendizaje, Rev. Tecnura 21 (53) (2017) 107–118, http://dx.doi.org/10.14483/22487638.9287.

[37] B. Smith, L. McGinty, J. Reilly, K. McCarthy, Compound critiques for conversational recommender systems, in: Proceedings of the 2004 IEEE/WIC/ACM International Conference on Web Intelligence, Washington, 2004, pp. 145–151.

[38] B. Steichen, H. Ashman, V. Wade, A comparative survey of personalised information retrieval and adaptive hypermedia techniques, Inf. Process. Manag. 48 (4) (2012) 698–724.

[39] E. Uruchrutu, L. MacKinnon, R. Rist, User cognitive style and interface design for personal, adaptive learning. what to model?, in: 10th International Conference, UM 2005. Edinburgh: Proceedings, 2005.

[40] Line Kolås, Arvid Staupe, A Personalized E-Learning Interface, 2007, pp. 2670–2675, http://dx.doi.org/10.1109/EURCON.2007.4400362.

[41] M. Nivethika, I. Vithiya, S. Anntharshika, S. Deegalla, Personalized and adaptive user interface framework for mobile application, in: International Conference on Advances in Computing, Communications and Informatics (ICACCI), IEEE, 2013, pp. 1913–1918.

[42] Dargent Lauren, Branthomme Arnaud, Kou Paul, Girod Hervé, Morellec Olivier, Chapter 53 - using machine learning algorithms to develop adaptive man–machine interfaces, in: Neuroergonomics - the Brain At Work and in Everyday Life, Academic Press, 2018, pp. 237–238, http://dx.doi.org/10.1016/B978-0-12-811926-6.00053-1.

[43] Jiří Šebek, Michal Trnka, Tom Černý, On Aspect-Oriented Programming in Adaptive User Interfaces, 2015, pp. 1–5, http://dx.doi.org/10.1109/ICISSEC.2015.7371024.

[44] Aqueasha Martin-Hammond, Foad Hamidi, Tejas Bhalerao, Christian Ortega, Abdullah Ali, Catherine Hornback, Casey Means, Amy Hurst, Designing an adaptive web navigation interface for users with variable pointing performance, in: Proceedings of the Internet of Accessible Things (W4A '18), ACM, New York, NY, USA, 2018, 31, http://dx.doi.org/10.1145/3192714.3192818, 10 pages.

[45] Jiangfan Feng, Yanhong Liu, Jiangfan feng yanhong liu intelligent context-aware and adaptive interface for mobile LBS, Comput. Intell. Neurosci. 2015 (2015) 489793, http://dx.doi.org/10.1155/2015/489793, 10 pages.

[46] R.B. Freeman, On the origin of species, in: The Works of Charles Darwin: An Annotated Bibliographical Handlist, second ed., Wm Dawson & Sons Ltd, Cannon House, Folkestone, Kent, England, 1977.

[47] L. Loewe, Genetic mutation, Nat. Educ. 1 (1) (2008) 113.

[48] Christoph Adami, Digital Genetics: Unravelling the Genetic Basis of Evolution. Nature Reviews Genetics, Nature Publishing Group, 2006, http://dx.doi.org/10.1038/nrg1771.

[49] S. Sulaiman, I.S. Sohaimi, An investigation to obtain a simple mobile phone interface for older adults, in: Proceedings of the International Conference on Intelligent and Advanced Systems (ICIAS '10), 2010, pp. 1–4.

[50] C. Bundesen, S. Vangkilde, A. Petersen, in: J.K. Tsotsos, M.P. Eckstein, M.S. Landy (Eds.), Recent Developments in a Computational Theory of Visual Attention (TVA), Vol. 116, Vision Research, 2015, pp. 210–218.

[51] John D. Gould, Clayton Lewis, Designing for usability - key principles and what designers think, in: En A. Janda (Ed.), Conference on Human Factors in Computing Systems, ACM New York, Boston, Massachusetts, 1983, pp. 50–53.

[52] F. Rosenblatt, Principles of Neurodynamics; Perceptrons and the Theory of Brain Mechanisms, Spartan Books, Washington, EE.UU, 1962.

[53] Halder, A. Jati, G. Singh, A. Konar, A. Chakraborty, R. Janarthanan, Facial action point based emotion recognition by principal component analysis, in: SocProS (2), 2011, pp. 721–733.

[54] A.P. Gosavi, S.R. Khot, Facial expression recognition using principal component analysis, Int. J. Soft Comput. Eng. (IJSCE) 3 (4) (2013).

[55] S. Milborrow, F. Nicolls, Active shape models with SIFT descriptors and MARS, in: VISAPP, 2014.

[56] C. Gackenheimer, What is react?, in: Introduction to React, Apress, Apress, Berkeley, CA, Berkeley, CA, ISBN: 978-1-4842-1246-2, 2015, http://dx.doi.org/10.1007/978-1-4842-1245-5_1.

[57] Brad Green, Shyam Seshadri, J.S. Angular, O'Reilly Media, Inc. 2013. ISBN:1449344852 978144934485.

[58] What are extensions? Chrome developer manual. https://developer.chrome.com/extensions.

[59] Daniele De Sensi, Tiziano De Matteis, Marco Danelutto, Simplifying self-adaptive and power-aware computing with nornir, Future Gener. Comput. Syst. 87 (2018) http://dx.doi.org/10.1016/j.future.2018.05.012.

Antonio J. Sánchez Martín. He begins her studies of Ph.D. in University of Salamanca in 2014. He obtained a Technical Engineering in Systems Computer Sciences degree in 2002, an Engineering in Computer Sciences degree in 2013 (both at the University of Salamanca) and a Bachelor Degree in Design in the High Institute of Art (Valencian Community) en 2008. As a researcher, his interests are focused on Internet of Things, Human Interfaces, Edge Computing, digital animation, context-aware systems. He has been be employed in recognized international companies, and now he works as senior developer in ElParking. He has participated as author in papers published in recognized international conferences and symposiums.

Sara Rodríguez González (Ph.D.). Received a Ph.D. in Computer Science from the University of Salamanca in 2010. She pursued her studies of Ph.D. in this University. She obtained a Technical Engineering in Systems Computer Sciences degree in 2004, an Engineering in Computer Sciences degree in 2007 at the University of Salamanca. She is Associate Professor at the University of Salamanca and researcher at the BISITE research group (http://bisite.usal.es). She has participated as a co-author in papers published in recognized international conferences and symposiums.

Fernando De La Prieta Pintado. He is Assistant Professor in the Department of Computer Science and Automation at the University of Salamanca. At the research level, he has been published in around thirty articles in journals and more than 80 articles in books and at recognized international conferences. He has worked on 50 research projects and around 20 research contracts. It is worth mentioning that he has been active in the organization of international scientific congresses (PAAMS, CEDI, FUSION, ACM-SAC, etc.).

Alfonso González-Briones He currently works at the Department of Computer Science and Automatics, at the University of Salamanca. He is also member of the BISITE Research Group since September 2014. He earned his Ph.D. in Computer Engineering from the University of Salamanca in 2018. At the same University, he obtained his degrees of Technical Engineer in Computer Engineering (2012), Degree in Computer Engineering (2013) and Master in Intelligent Systems (2014).