Received November 10, 2020, accepted November 16, 2020, date of publication November 20, 2020,
date of current version December 7, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3039616

A Methodology for User Interface Adaptation of Multi-Device Broadcast-Broadband Services

ANA DOMÍNGUEZ[1], JULIÁN FLÓREZ[2], ALBERTO LAFUENTE[3], STEFANO MASNERI[1], IÑIGO TAMAYO[1], AND MIKEL ZORRILLA[1]
[1]Department of Digital Media, Vicomtech, 20009 San Sebastian, Spain
[2]Tecnun, University of Navarra, 20018 San Sebastian, Spain
[3]Distributed System Group, UPV/EHU, 20018 San Sebastian, Spain

Corresponding author: Ana Domínguez (adominguez@vicomtech.org)

The work of Ana Domínguez, Stefano Masneri, Iñigo Tamayo, and Mikel Zorrilla was supported in part by the Basque Government under Grant ZL-2018/00079 (TBTU) and Grant ZL-2019/00029 (TBTU), and in part by the European Union Horizon 2020 Research and Innovation Programme under Grant 883293 (INFINITY). The work of Alberto Lafuente was supported in part by the Spanish Research Council under Grant TIN2016-79897-P, and in part by the Basque Government under Grant IT980-16.

ABSTRACT New audiovisual experiences involve consuming several contents displayed through multiple internet-connected devices. The TV is still the central hub of the living room, but it is often used simultaneously with other screens. Consequently, the user has the chance to consume all different contents at once across multiple devices. However, no existing adaptation models are available to dynamically adapt such a multitude of contents in multi-device contexts. To address this gap, this article proposes a novel multi-device adaptation methodology to build adaptive User Interfaces for multi-screen hybrid broadcast-broadband TV experiences. The methodology is extensible to any kind of content, device and user, and is applicable to different contexts considering technological evolution and other fields of application. The proposed methodology is the outcome of extensive research that arose from a previous multi-device media service deployment with broadcasters.

INDEX TERMS Adaptation, cross-device, methodology, multi-device, multimedia broadband and broadcasting systems, multi-screen, UI adaptation, UI optimisation.

I. INTRODUCTION

The evolution of Internet-connected devices, as well as the changes in the way media is produced, distributed and consumed, have promoted the mobility and ubiquity of broadcast and media services. Consequently, the audiovisual sector has been transformed into a hybrid ecosystem where content is distributed across multiple devices [1]. Moreover, a single user very often consumes content from more than one device at a time, and remotely connected multi-user experiences are becoming popular [2], [3].

This novel context requires highly flexible User Interfaces (UIs) where the content is not only adapted to any target device but also requires an adaptation to multi-device environments, composed of a number of devices that are being used simultaneously by one or multiple end-users. In this context, we call *adaptation* the process of (efficiently) representing the hybrid media content of a TV show across multiple displays of a multi-device environment. Apart from the evident advantages delivered by multi-screen media, such as the enrichment of a TV programme with second screen experiences, more screens demand a higher cognitive load for viewers to understand what they are watching and to correlate the content, such as consuming different multi-view video streams in different devices that complement the TV broadcast mainstream. The required visual attention is also increased and needs to be distributed across multiple displays located at different places in three-dimensional space [4], [5].

Accordingly, the user experience becomes a key factor to facilitate understanding the application and to provide an intuitive interaction method across multiple screens.

On one hand, there are aspects in interface design (such as the functionality and usability) that are well-known dimensions in the field of Human Computer Interaction (HCI), since HCI emerged by focusing on the development of methods and techniques to improve usability [6]. Moreover, in addition to traditional HCI parameters, there is a new wave in this field emphasising the importance of aesthetic aspects in

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

interface design for the users' likability and system acceptability [7], [8]. However, most HCI research is oriented around an end-user using a single device, and therefore lacks an extensive analysis of multi-screen environments.

On the other hand, optimisation methods have been a long-standing topic in HCI research for UI design. Nevertheless, these optimisation methods have been explored as supplemental ways to help speed up the design cycle and improve the design quality. Theoretically, model-based UI optimisation refers to the use of combinatorial methods to solve a UI design problem formulated as a search problem by using predictive models of human behaviour and experience [9]. However, to the best of our knowledge, no existing adaptation models are available to dynamically and seamlessly adapt such a multitude of content to multi-device and multi-user contexts, where a user or a group of users consumes content from a set of devices at the same time.

To address this gap, this article proposes a novel methodology to build adaptive UIs for multi-device media applications and services, based on the COPE (Create Once Publish Everywhere) concept [10].This means that a single application code is developed and the application itself adapts to the multi-device context of the user. To face that challenge, the research is focused on Web technologies [11], since they provide the biggest advantages in terms of interoperability. This is very relevant taking into account the huge amount of devices and operating systems available nowadays. The methodology described in this manuscript, while being applicable to other fields and development environments, will be focused on a hybrid broadcast-broadband environment, since this is the field which currently presents a clear need. Previous work reinforces this assertion, since the research in [12] proposed a Web-based distributed architecture for multi-device adaptation in media applications, while [13] described its implementation and deployment for a TV programme and [14] shows a demo of the experience. However, the architecture proposed in [12] and deployed in [13] required the specific creation of adaptation rules for each TV programme. Instead, a universal, dynamic and automatic multi-device adaptation would be desired for any type of TV show. In fact, the hypotheses and conclusions that arose from the previous research claimed an improvement and optimisation of the usability and universality of the multi-device UI.

Regarding universality, recently we have specified a formal model for UI adaptation in multi-device services [15] that can be applied in multiple fields. In the present work we introduce a methodology that complements this model. The methodology is specifically oriented to provide optimization, efficiency, and usability in hybrid broadcast-broadband programmes, and has been validated through use cases for this scenario. Nevertheless it can be easily adapted to other fields. [16] shows implementations of multi-device experiences in industry and crisis environments where the methodology could be applied.

The main contribution of this article is the definition of a methodology for the adaptation of the UI of multi-device media services, which includes (1) the characterisation of UI elements, including the components, devices and layouts, (2) a two-step adaptation process that assigns components into devices and finds suitable layouts for the assignations, and (3) an evaluation with an illustrative battery of use cases that check the quality, the efficiency and the universality of the aforementioned adaptation process. The code generated for the adaptation and evaluation is available on Github [17] and enables the replicability of the experiments and the application to other fields.

The structure of this article is defined as follows. Section II describes the related work. Section III provides an overview of the proposed methodology. Section IV identifies the UI elements and properties to be considered in the adaptation process. Section V describes the proposed two-step adaptation process. Section VI describes the effect of the context in the adaptation process and Section VII provides examples and evaluation of the quality, efficiency and universality of the proposed approach. Finally, Section VIII provides the conclusions and future work.

II. RELATED WORK

When broadcasters or application developers are creating a single-device UI, they typically define a template to organise the items in the layout, and usually describe a different layout behaviour for each target device. For instance, in Web applications, developers follow the cost-effective trend of Responsive Web Design (RWD) [18] and provide a CSS template with Media Queries [19], which adapts the UI of the application to the features of the device.

However, in a multi-device system where a user can be consuming content from multiple screens at the same time, the typical template-based approach cannot work, as the set of elements to be shown on each device can change over time, depending on the number of components to visualize and the number of devices connected to the system. In a multi-screen environment, such as that addressed in [12], the content is divided into logic elements with the Web components specification [20] following an object-based broadcasting approach [21]. In this case, the list of elements available on each device will depend on the changeable context of the users. In this scenario, it is tedious, very expensive and unaffordable for developers to provide an explicit template to organise the elements on the UI of each target device, considering all the possible combinations. Thus, the solution described in [12] provides generic and arbitrary divisions to arrange the elements in a responsive UI following some hints given in the application code and creates a specific layout template depending on the context of the user.

XDBrowser [22], [23] project investigates how web browsers can be improved to better support parallel multi-device usage and provides a proof-of-concept implementation that segments web pages and distributes the parts across devices. Webstrates [24] synchronises elements across devices operating on the level of the Document Object Model (DOM) while maintaining exact copies on different devices. Sarkis *et al.* [25] allowed for the re-use of existing single-screen applications to automatically create synchronous multi-screen applications that analyse the code of the application and classifies different elements, such as HTML tags or event listeners. Other frameworks such as HuddleLamp [26] and Connichiwa [27] work with graphical aspects of UIs, for instance, when spanning one image across multiple screens. Other prior works on cross-device interfaces, such as the ones described by Frosini and Paternò [28] or Yang and Wigdor [29], have proposed methods for synchronising elements across devices. All of these frameworks allow for the development of new multi-screen applications but they all require developers to explicitly define how to distribute interface components across displays.

Vistribute [30] proposes a framework that identifies important properties and relations for distributed visualisation interfaces. Vistribute also provides six heuristics that can guide in the automatic distribution of visualisations in changing device setups together with their web-based implementation. However, those heuristics are focused on data analysis visualisations and therefore they are again not general enough to extend them for the adaptation of other contents in different fields.

In the mentioned literature there is a lack of an adaptation process that is able to distribute and adapt such a dynamic amount of content to any device at run-time and in real-time for a multi-device and in a multi-user context. Such a process would be beneficial for existing solutions that try to achieve a seamless integration between traditional TV and new consumption habits, e.g. [31]–[35], for current multi-device platforms that distribute the content among devices only under user interaction, e.g. [25], [36] and finally for multi-screen solutions in other fields such as industry [16], [37] or crisis management [38]. However, the definition of the process itself is not straightforward, which is why this article proposes a methodology to adapt the UI of multi-device media services.

III. OVERVIEW OF THE ADAPTATION METHODOLOGY

In the context of the present work, the aim of adaptation is to provide a suitable distribution of the contents of a hybrid broadcast-broadband programme into different device screens in the living room. Figure 1 shows the target scenario in which a user consumes a TV programme enriched by additional content accessible from different devices. All of these contents should be provided proactively and responsively, according to the context conditions and changes.

Therefore, proposed adaptation methodology aims to adapt the interface of hybrid broadcast-broadband services, avoiding the definition of specific adaptation rules for each TV programme. The methodology aims to be extensible for any kind of situation: any kind of TV programme or content, any device type, and any kind of UI layout template, to be used by any kind of user and in any context, and applicable even if technological changes occur (including new devices beyond

FIGURE 1. A scenario for hybrid broadcast-broadband services.

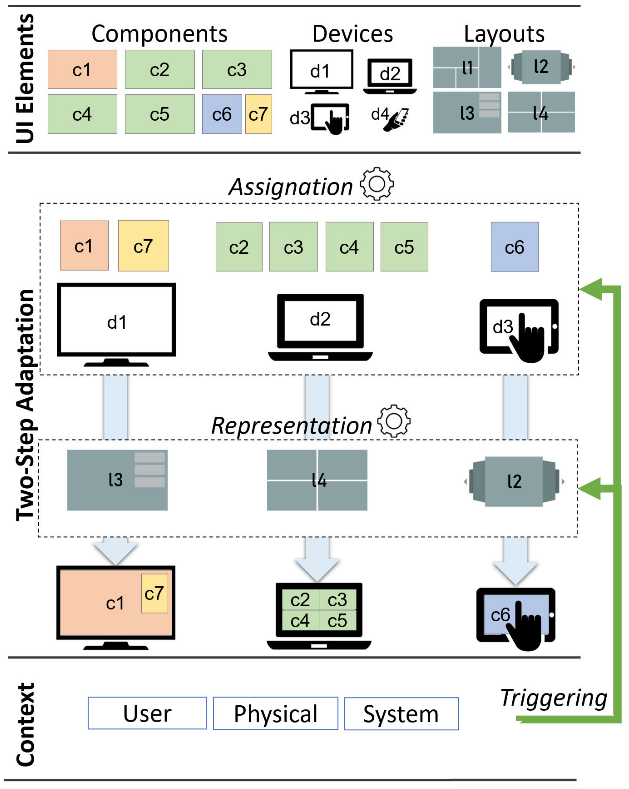


FIGURE 2. Overview of the proposed methodology.

AR/VR headsets or smart watches, new context conditions, new ways of interaction with TV or media, etc.).

This methodology is the outcome of an extensive research that identifies the elements, properties and criteria that should take part in the adaptation process [12], [13], [15], [39]–[41]. More specifically, our approach is based on the definition of a general adaptation process that allows efficient implementations while considering also the effect of the context in the final UI.

Figure 2 shows an overview of the methodology. In the Figure, seven components of four different types have been identified in the TV programme that a user is watching. The content components are assigned to three active devices at user's hand. In a second step, the components are represented in their respective assigned devices using some of the available layout templates of the service. For example, in the figure the smart TV uses a Picture-in-picture (PiP) layout to represent components c1 (conceivably the main programme) and c7. This adaptation process is triggered by changes in the application context, such as a new component to show, a device turned off or the interaction of the user triggering specific interactions.

In the following Sections, the adaptation methodology is described in detail, describing it in three main parts:

1) Characterization of the elements of the User Interface (Section IV).
2) A two-step characterization of the adaptation process based on the aforementioned UI elements (Section V).
3) Finally, the role of the context as a drive of the adaptation process, which will be described on the basis of the implementation example (Section VI).

IV. CHARACTERIZATION OF THE ELEMENTS OF THE USER INTERFACE

In [15], three set of elements in a multi-device service have been identified and formalised: content *components*, target *devices*, and UI *layouts*. In this section we identify and characterize the specific elements to be found in the scope of broadcast-broadband services.

On the one hand, a hybrid broadcast-broadband programme will have different pieces of content, the *components*, such as the main programme (typically the broadcast), other media resources (multi-view live cameras, on-demand complementary content, etc.) and other types of information (statistics, graphics, banners, etc.). On the other hand, the *devices* that end-users will use to consume the content elements are also crucial elements to build an adaptive multi-device UI. Finally, every device taking part in the multi-device experience needs to visually organise the components in the display, assigning a location, size and aspect ratio to each of the components, creating UI *layouts*.

A. COMPONENT TYPES AND PROPERTIES

We have conducted a previous research [40] to provide component typification. On the basis of an analysis of the contents of different TV services, the work proposes a way of componentising hybrid broadcast-broadband media programmes and characterizes the components in terms of component properties, following the model proposed in [15]. The work defines 8 types of components for an interactive multi-device TV programme:

1) **Main programme**: The mainstream audiovisual content that drives the experience.
2) **Advertisements**: Business-related resources advertising something within the TV programme.
3) **Secondary videos**: Additional videos sourced by the broadcaster.
4) **Banners**: Additional information including notifications, headlines and small texts.
5) **Static Data**: Elements that show relevant data, longer texts or images, that might not bring interaction by the user.
6) **Interactive Data**: Elements that show relevant data, diagrams or tables that might bring interaction by the user.
7) **UGC - User Generated Content**: Additional videos generated by the viewers.
8) **Social content**: Additional information coming from the opinion of the viewers (social networks, a quiz, etc.).

Our implementations are based on this set of component types, or a subset of it. Every component in the multi-device experience is labelled with one of the aforementioned types. Then, some properties have been identified in such a way that every component can be evaluated in terms of them [25], [42], such as:

- **Attention**: The demand of attention required for that specific component regarding the entire multi-device media service, from the perspective of the content provider of the broadcaster. For example, the main programme is supposed to require high attention.
- **Interactivity**: The degree of interaction that the component allows depending on its type. For example, a chat component requires high interactivity.
- **Processing Requirements**: The CPU/GPU processing and memory demands of a component (e.g. real-time decoding of H265 video streams).
- **Broadcast Requirements**: The demand of a broadcast tuner to decode the TV content.
- **Confidentiality**: The level of privacy required by a component, since it could include personal or customised information that might not be of interest for other viewers in the same physical space. This is the case, for example, of a secondary video, different from the main programme.

This set of properties could be extended to other properties depending on the use case. Table 1 shows the intuitive parametrisation performed for our reference implementation for all the component types identified in [40]. Each of the values would require an exhaustive analysis which is considered out of the scope of this article.

B. DEVICE TYPES AND PROPERTIES

In the context of hybrid broadcast-broadband media services, such as the one described in [13], there are usually four types of devices involved, also called *core devices* or *core screens* [43]. We are talking about smartphones, laptops, tablets and Smart TVs. These devices will be the basis for our implementation, even though the model we have defined is able to incorporate other types of devices, such as smartwatches, smart speakers, VR/AR headsets or future incoming HCI devices.

Device properties will be generically assigned to the considered device types. Most of the properties can be

TABLE 1. Summary of used component types and properties.

Properties	Attention	Interactivity	Processing requirements	Broadcast requirements	Confidentiality
Main	1	0	0.3	1	0
Sec. videos	0.7	0.3	0.9	0	0.1
Banner	0.8	0.2	0.1	0	0.1
St Data	0.6	0.2	0.2	0	0.3
Dyn Data	0.6	1	0.3	0	0.8
Social	0.3	1	0.3	0	1
UGC	0.3	1	0.7	0	0.9
Adv	0.9	0	0.5	0.8	0

TABLE 2. Summary of used device types and properties.

Properties	Screen size	Input capabilities	Processing capabilities	Privacy	Broadcast capabilities
Smartphone	0.2	0.7	0.6	1	0
Tablet	0.6	0.8	0.7	1	0
Desktop	0.7	1	1	0.5	0
SmartTV	1	0.2	0.4	0.1	1

theoretically obtained from the scripting capabilities of Web browsers, frameworks or libraries. However, this is sometimes not possible and often provides false positives, making such information unreliable. Additionally, device typification must be performed at run-time.

In a previous research, we used the information about the device made available by the browser to implement device typification. The work described in [41] compares three different methods of Web-based device type detection using the User Agent of the browser, obtaining an overall accuracy of 95%, which is acceptable for the use cases that this article addresses.

Nevertheless, there are device properties that are related to usage patterns, such as the degree of privacy of a device, which cannot be obtained from the browser and have not been integrated in our implementation.

Again, every device in a multi-device experience is labeled as smartphone, tablet, laptop or smart TV and then some properties have been identified in such a way that every device can be evaluated in terms of them, such as:

- **Screen Size**: The size of the display of the device. Trivially, the TV would have a large size and smartphones a small one. However, this property could integrate other features such as resolution or use distance. In this case it would then be better to consider the *apparent size* of the screen device.
- **Input Capabilities**: The features and mechanisms that a device provides for interaction. Current Connected TVs have low input capabilities, since all the interaction is through the remote control, while laptops, through the availability of keyboard and touchpad, have higher input capability.
- **Processing Capabilities**: The processing and memory capabilities of a device.
- **Graphic/video Capabilities**: The rendering and video decoding capabilities of a device. Note that current technologies use GPUs for both processing and graphic/video support.
- **Broadcast Capabilities**: The broadcast tuning and decoding capabilities of a device.
- **Privacy**: The level of privacy that a device provides in terms of enabling the ability to not share the content in the display with other viewers in the same physical space.

This set of properties could be extended to other properties depending on the use case. Table 2 shows the intuitive parametrisation performed in the implementation for all the identified device types. Each of the values would require an exhaustive analysis which is considered out of the scope of this article.

C. LAYOUT TEMPLATES

Today devices present a set of components to the user as efficiently as possible. The way those components are presented in the screen is called layout. Even when considering a single screen, the variety of ways to organise the components is very extensive and suffers combinatorial explosion [44].

In order to simplify the methodology, we choose a set of layout templates of standard use in current applications and devices. These are Picture-in-Picture (Figure 3A), Split (Figure 3B) and Carousel (Figure 3C):

1) **Picture-in-Picture (PiP)**: space sharing template in which a component is shown in full screen and the others are overlapped with a smaller size over the first one. PiP shows all the components at the same time to the detriment of the shown area of the first component (overlapped) and the resolution of the secondary components.
2) **Split**: space sharing template in which the screen is divided into a regular grid according to the number of components. Unlike PiP, there is no overlapping. However, empty space appears when the number of components does not match the number of cells established by the template.
3) **Carousel**: time sharing template in which all the components are organised in a carousel that allows to

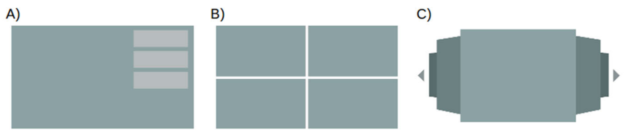


FIGURE 3. Considered layout templates.

TABLE 3. Layout properties for each template.

Property	PiP	Split	Carousel
Time sharing	✗	✗	✓
Space sharing	✓	✓	✗
Overlapping	✓	✗	✗
Scrolling	✗	✗	✗
Distortion	✓	✓	✓
Prior components	✓	✗	✗

left- or right-slide all the elements. Carousel is commonly used for image galleries.

These templates can also be described in terms of a set of properties:

- **Time sharing**: At a given time t, only a subset of components is shown, thus each component is given a time slot of the screen resource.
- **Space sharing**: Components share the screen area, so a component should be reduced in size to fit a fraction of the screen.
- **Overlapping**: A component may overlap other components.
- **Scrolling**: A component may shift smoothly, vertically or horizontally, across the screen. Note that when scrolling, the screen area is shared in both time and space.
- **Distortion**: Components may be distorted to fit the assigned space of the screen both in scale and aspect ratio. If distortion is not provided, either the component should be cropped or an unused area is left.
- **Prioritisation of the components**: Components can be given different ranks, prioritising some over the others. Prioritisation can be performed spatially (assigning more screen size) or temporally (assigning longer time slots).

A summary of the template layout properties is shown in Table 3.

The template to be used in each specific situation will depend on parameters such as the type of device, the number of components to be shown, the nature of the application or the number of devices being used at the same time [47].

V. ADAPTATION PROCESS

Following the adaptation model defined in [15], we divide adaptation into two goals:

1) Maximizing the quality of the distribution of the visual content components across the connected devices.
2) Maximizing the quality of the User Interface layout in each device, given the components assigned to it.

Instead of evaluating all the adaptation solutions, which will lead to an optimal result, our implementations addresses both goals in two sequential steps, that we call *assignation* and *representation* respectively.

Addressing the partial goals separately leads to sub-optimal adaptation solutions, nevertheless we have adopted the two-step approach for performance and responsiveness reasons, as well as for the experience acquired in a previous pilot [13]. Therefore, the adaptation process will execute one common assignation step and as many representation steps as devices connected. The following subsections describe each step.

A. ASSIGNATION

Assignation, also referred as Step 1 of the adaptation process, uses vectors of properties for components and devices to calculate affinities on the basis of an affinity matrix that relates both sets of properties. This affinity matrix is shown in Table 4. In a real deployment, the values of the matrix should be carefully set, which involves extensive user evaluation and/or a continuous adaptive learning process. However, the scope of the present work is aimed at the definition and evaluation of an adaptation methodology. Therefore, we have built the affinity matrix with values obtained by polling a reduced community of users.

As an example for the assignation, if we take into account attention, interactivity and processing requirements properties for the components, the property vector for the Main component obtaining the values from Table 1 would be:

$$P_{Main} = [1, 0, 0.3] \tag{1}$$

In the same way, for devices, if we take into account the screen size, input capabilities and processing capabilities, the property vector for the TV, smartphone and laptop obtaining the values from Table 2 would be:

$$P_{TV} = [1, 0.2, 0.4] \tag{2}$$
$$P_{smartphone} = [0.2, 0.7, 0.6] \tag{3}$$
$$P_{laptop} = [0.7, 1, 1] \tag{4}$$

Lastly, the affinity matrix would be taken from the corresponding cells in Table 4.

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 0.1 & 1 & 0.5 \\ 0.3 & 0.3 & 1 \end{bmatrix} \tag{5}$$

Then, the implementation of Step 1 is quite straightforward. The affinity matrix is used to translate the properties of each device to their level of compliance regarding the requirements imposed by each of the component properties. After that, a multidimensional comparison is performed, as shown in Figure 4. As can be seen the blue line represents the requirements presented by the component while the other curves show the compliance by each of the devices.

In order to assign a given component, for our reference implementation these criteria have been followed:

TABLE 4. Affinity matrix.

Properties	Screen size	Input capabilities	Processing capabilities	Privacy	Broadcast capabilities
Attention	1	0	0	0.1	0
Interactivity	0.1	1	0.5	0.9	0
Processing requirements	0.3	0.3	1	0	0
Broadcast requirements	0	0	0	0	1
Confidentiality	0	0	0	1	0

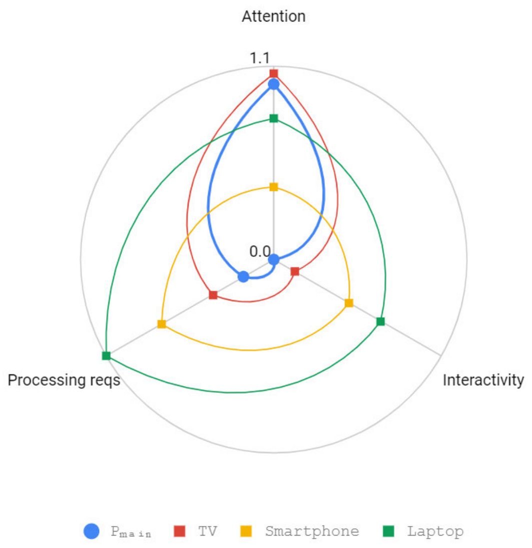


FIGURE 4. Multidimensional comparison of component properties for each device. TV fits every property since all the values are higher than those of P_{main}.

1) Choosing a device fitting every property, or the highest amount of them. In our example, the TV would be the selected device for the main programme component.
2) Minimizing under-fitting values. In our example, in absence of a TV, we would assign the component to the laptop.
3) Finally, over-fitting values are considered, choosing the device whose affinity values are the closest to the requirement values.

Note however that other implementations could use different criteria, as for example maximising the usage of resources. In this case, following the example of Figure 4, the TV would be again the preferred device, while the laptop would be last option as choosing it would lead to wasting unused processing and interactivity resources.

All in all, flexible data structures have been implemented to allow for adding or removing components or devices and evaluating all their properties.

B. REPRESENTATION

Representation, also referred as Step 2 of the adaptation process, uses the assignments obtained in Step 1 to calculate the layout template in each device. Our reference implementation for layout selection is based on a set of criteria that allow to evaluate the layout quality according to the general evaluation model described in [15], which defines a function associated to each criteria:

$$\alpha_k = \rho_k(l, d_i, C_i) \tag{6}$$

where α_k is the value for the criteria obtained through its associated expression ρ_k, that depends on the evaluated layout, the target device and the set of components to show. The overall layout quality is then modelled as the weighted product of the criteria:

$$\beta(l, d_i, C_i) = \prod_k \alpha_k^{w_k} \tag{7}$$

where w_k are the weights for each of the criteria.

For the sake of simplicity, the chosen criteria are calculated on the basis of simple geometric parameters and do not consider aspects that would strongly depend on subjective user's likings (e.g., aesthetic) and will require extensive user evaluation. We implemented the following criteria:

- **Rate of the components shown** (α_1): This criterion evaluates the portion of the components that is shown in the display, and refers to three different aspects: (a) the portion of the number of components, (b) the portion of the component area, and (c) the portion of time that a component is shown. Note that Split layouts rank the highest in every aspects, while PiP is penalised in (b) by the overlapped area of the main component, and the time sharing layouts, such as Carousel, are penalised in (c) for the time slice where components are shown.
- **Representation of every shown component in its entirety** (α_2): This criterion is related with how the components shrink to fit to a specific space slot in the space-sharing layouts. The criterion ranks the maximum for Carousel, while it can penalize Split and PiP in two different aspects: (a) shrink in the scale of a component, and (b) distortion of the aspect ratio of the component (the last aspect has not been considered in our example implementation).
- **Efficiency in the use of the screen area** (α_3): Although we are allowing some degree of distortion, in Split layouts the screen is divided in a regular grid that, for practical reasons, excludes extreme configurations (e.g., many components in a single row). As a consequence, for some specific numbers of components, an unused screen area will be generated, which is accounted for as lost space and penalises this criterion.

These criteria depend on the number of components and the screen size property of the devices. To properly define

TABLE 5. Parameters related to apparent areas of screens.

Device	Smartph.	Tablet	Laptop	TV
Screen diagonal (cm)	12.5	25	35	150
User distance (cm)	40	50	60	250
Angle (degrees)	17.36	26.58	30.25	30.93
Apparent area, S (m^2)	0.044	0.100	0.127	0.132
Comp. min size, S_c (m^2)	0.02	0.02	0.02	0.02
N. of components, $\lfloor S/S_c \rfloor$	2	4	6	6

TABLE 6. Screen related layout parameters.

Parameter	PiP	Split	Carousel
Max N. of main comp.	1	S/S_c	1
Insertion frac., F_i	0.33	0	0
Insertion min size, S_s	$S_c/3$	N/A	N/A
Max. N. of insertions	F_iS/S_s	0	0

the screen size property, we considered the *screen apparent area*, denoted as S.

To measure the apparent area of a screen with diagonal h meters and a normalized 1:1 aspect ratio, we situate the screen at the usual watching distance and calculate the projected area of the screen to distance 1 meter. Note that this results in more realistic relations than the ones obtained comparing raw screen areas. For example, a screen with a diagonal size h situated at a distance of l meters from the user has the same apparent area as a screen of size kh at kl meters.

Resolution has been ignored for the screen size definition, since it is not a constraint in current devices. However, some ergonomic aspects should be considered in real deployments. For example, for similar apparent sizes, a TV would provide a more comfortable viewing distance for the user's eyes, while smartphones or tablets can be brought nearer more easily.

Table 5 shows an example of the estimated apparent areas for the four considered device types as well as some related parameters that we explain next.

A relevant parameter for layout generation, specifically for the Split template, is the minimum apparent size of a component to be comfortably seen by the user, denoted as S_c. Indeed, this is a user-related parameter that would deserve extensive user evaluation, in addition to the ability for configuration and adaptation from user context parameters. To provide an example to illustrate the prototype implementation of our adaptation model, we will set S_c to 0.02 square meters. This results, for example, in one half of a smartphone screen at 40cm from the user. For the other devices, Table 5 shows the number of minimum size components that would fit into the screen, i.e., S/S_c, conveniently rounded to an integer number.

For PiP layouts, two additional parameters are required: the fraction of the screen area devoted to inserting the overlapped secondary components, F_i, and the size of the secondary components, S_s which in general can be smaller than S_c. Note that F_iS/S_s denotes the maximum number of insertions in a PiP layout. Table 6 summarizes the values for the three templates adopted in the implementation.

TABLE 7. Layout criteria and coefficients.

Criterion	Description	Coefficient	Coeff. value
α_1	Rate of the components shown	w_1	1.4
α_2	Representation of every shown component in its entirety	w_2	0.3
α_3	Efficiency in the use of the screen area	w_3	1.2

The criteria described previously have been evaluated according to our evaluation model and using empirical parameters in Table 5, Table 6 and Table 7.

Figure 5 shows the quality curves obtained for each layout in terms of the number of components and providing a different result for each device. Observe that time-sharing layouts (Carousel) maintain a significant quality level for a high number of components whereas space-sharing layouts (PiP and Split) are more suitable for a low number of components, depending on the particular election on the device type and the efficient fit of the components in the screen (e.g. the quality decreases sharply for Split layout when the number of components cannot be arranged in a grid occupying the entire screen).

Again, the implementation uses flexible data structures to allow adding or removing other criteria easily. Furthermore, it allows not only for the evaluation of the three pure layout templates we are considering but also for a combination of hybrid layouts, as, for instance, a carousel showing several components simultaneously, as well as many others. Additionally, an exhaustive user evaluation or an adaptive learning process could find the most appropriated values for the specific deployment and user profile, showing at the same time the power of generalization of our model.

VI. THE ROLE OF CONTEXT: TRIGGERING THE ADAPTATION PROCESS

In the previous sections the two-step adaptation process has been described. The question now is when the adaptation process should be launched, either from the beginning or only from the representation step in some specific devices. To address this issue we use context information.

The information about contextual factors and external event occurrences while using a service is usually referred to as context [45]. In our system, context changes are used to trigger the adaptation process in such a way that the system can update and maintain the user experience at convenient levels.

Usually, context is divided in three parts: user context, physical context and system context. The user context would include the user preferences and user state (e.g., mood or stress level). Explicit interaction could also be considered as the user context updates. Physical context parameters can include the location, time, ambient light, and noise level. Finally, the system context is determined by the state parameters of the devices, including the battery level, connectivity conditions, etc.

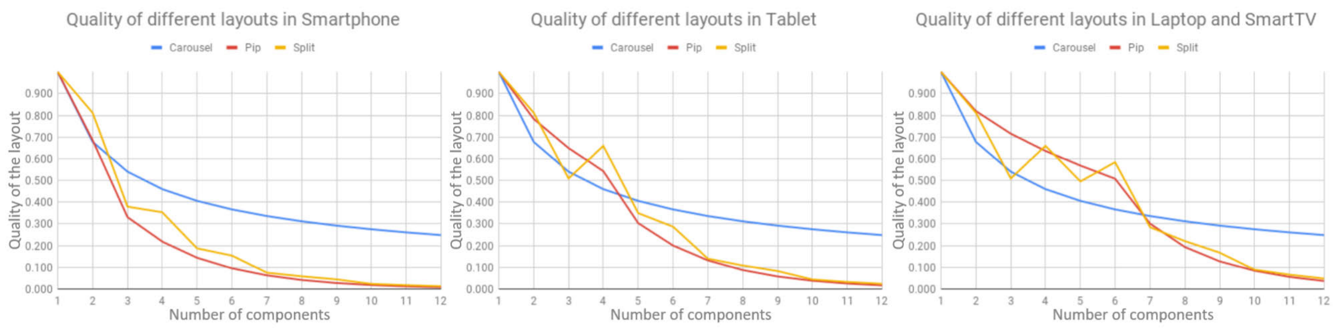


FIGURE 5. **Quality of the layouts in terms of number of components.**

Nevertheless, for the scope of this article, we manage context information as an asynchronous sequence of events. Either directly or indirectly, an event of the sequence is able to promote a run of the adaptation process. To complete the picture (refer to Fig. 2), our multi-device adaptation environment can be implemented as an event-driven system where events can be produced by different situations:

1) A change in the component set available in the service produced by the broadcaster. As an incremental modification, only Step 2 will be executed in the involved devices.
2) Joining or leaving a device, generally involving a rerun of the whole adaptation process.
3) An explicit user interaction, when the user moves a component from one device to another, activates/hides a component which finds interesting/uninteresting or changes the layout of a device. This executes Step 2 most of the times.
4) A dramatic change in a context parameter such as a decrease in the bandwidth available for a device or group of devices. In this case, the whole adaptation process is rerun in order to redistribute the components taking into account the status of the corresponding parameter.

VII. EXAMPLES AND EVALUATION OF THE ADAPTATION METHODOLOGY

This section provides an evaluation of the proposed methodology and the underlying model [15] for the UI adaptation of multi-device media services regarding three different aspects: quality, efficiency and universality.

Quality and efficiency, that we address in Subsection VII-A, refer to the performance of the adaptation in the particular scenario of broadcast-broadband media services considering the required responsiveness of a system, possibly including devices of limited computing capabilities. In this context, the quality of the (sub-optimal) 2-step adaptation result is measured in relation with the best possible adaptation solution. To that end, a trade-off between the adaptation quality and the computation effort for a responsive implementation has to be taken. Therefore, in order to set the efficiency of the adaptation solution, we have measured times for both computing the 2-step adaptation process and searching the global optimal solution in different devices.

Universality refers to the flexibility of the adaptation process to fit any context condition, its adaptability to technological changes and new user habits, and its capability to be extended to new multi-device scenarios. Here, in Subsection VII-B, we confine universality to the broadcast-broadband media services arena. In [15] and [37] it is shown how the adaptation model can be applied to other multi-device scenarios. In order to evaluate the universality in broadcasting scenarios we have taken as a reference a real deployment of the adaptation process [13] used during an election programme of the Basque public broadcaster. On the basis of this deployment, we have defined a set of uses cases that consider novel properties for components and devices, as well as new criteria to evaluate the layout quality.

All the code for the evaluation is available in [17].

A. EVALUATION OF THE QUALITY AND EFFICIENCY OF TWO-STEP ADAPTATION PROCESS

To evaluate the two-step adaptation process we have initially defined a set of components, a set of available devices, and a set of available layout templates, as well as representative properties and parameters for all the UI elements. Moreover, the evaluation model presented in [15] has been implemented in order to compare the quality of the two-step adaptation process with the optimal adaptation solution.

To set the quality of the adaptation solutions, we rank the two-step results in relation to the global optimal solutions, according to the evaluation model [15]. Basically, the quality of an adaptation result is computed as the mean of the individual quality of the adaptation for every individual component, and is denoted $\mathcal{E}^{\mathcal{H}}$. In turn, the quality of a single component is calculated as a function of its affinity with the assigned device and its representation quality with the selected layout, as explained in Section V.

Figure 6 shows a picture of a scenario extracted from the deployment described in [13] where a viewer consumes a multi-device TV programme delivered by a broadcaster. That experience has been useful to synthesize the use cases addressed in this section which are seen as a tool to validate the developed adaptation model. Therefore, three different

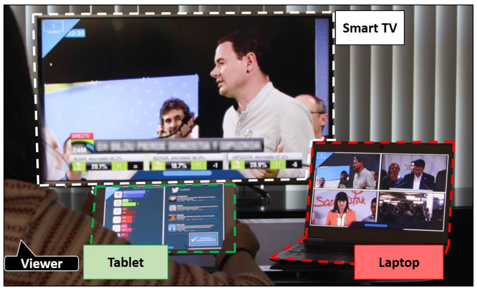


FIGURE 6. A picture of the deployment of a multi-device media service.

TABLE 8. Summary of the use cases.

USE CASE 1	
7 Components	Main Programme, Video 1, Video 2, Video 3, Video 4, Dynamic Data, Social
1 Device	TV
USE CASE 2	
7 Components	Main Programme, Video 1, Video 2, Video 3, Video 4, Dynamic Data, Social
2 Devices	TV, Smartphone
USE CASE 3	
7 Components	Main Programme, Video 1, Video 2, Video 3, Video 4, Dynamic Data, Social
3 Devices	TV, Smartphone, Laptop

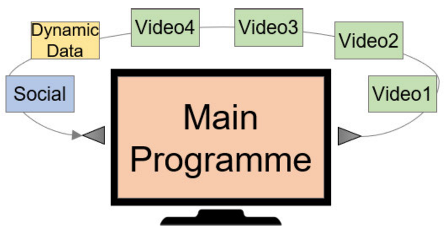


FIGURE 7. Diagram of the adaptation outcome in use case 1.

use cases have been defined, as summarized in Table 8, using a subset of components and devices addressed in the implementation.

1) USE CASE 1 - SEVEN COMPONENTS ON A TV

This use case evaluates the implementation with the aforementioned 7 components having only a Smart TV. As expected, all the components are shown on the TV and the Carousel is the selected layout (see Figure 7).

The overall adaptation quality is $\mathcal{E}^{\mathcal{H}} = 0.28$. In this context, the viewers will be able to mainly follow the TV show, while having the opportunity to perform a kind of content-hopping using the arrows in the remote control to see the other components.

It is worth noting that, in this case, the adaptation quality is severely affected by the inclusion of the *Social* content. This is mainly due to the high interactivity it requires and the low input capabilities the Smart TV is supplied with. Accordingly, if the social content would not be displayed, the Step 2 of the adaptation process would choose the Split layout for

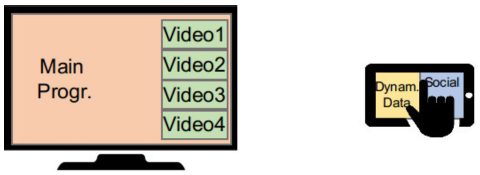


FIGURE 8. Diagram of the adaptation outcome in use case 2.

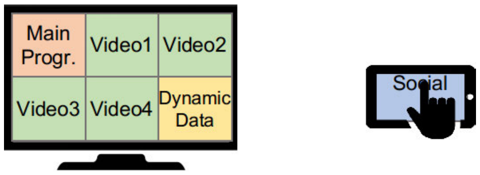


FIGURE 9. Diagram of the best possible adaptation outcome in use case 2.

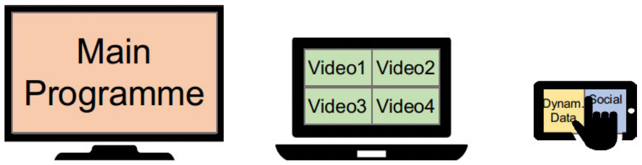


FIGURE 10. Diagram of the adaptation outcome in use case 3.

the remaining six components and the quality figure would increase to $\mathcal{E}^{\mathcal{H}} = 0.52$.

Note also that the adaptation result obtained by the two-step approach for Use case 1 is optimal. Since Step 1 is trivial (one device), only one assignment is possible.

2) USE CASE 2 - ADDING A SMARTPHONE

If a viewer is consuming the aforementioned 7 components using two devices simultaneously (a TV and a smartphone), the recommended assignation is to show the main programme and the four secondary videos on TV, while showing dynamic data and social components on the smartphone (see Figure 8). This scenario represents an overall quality of $\mathcal{E}^{\mathcal{H}} = 0.66$, increasing the previous values. This mainly occurs because the components that were less suitable for the TV are now on the smartphone and also because the TV now has to represent less components, allowing a PiP layout.

Analysing all the adaptation solutions, we found an optimal quality figure $\mathcal{E}^{\mathcal{H}} = 0.76$ showing the social content on the smartphone, while displaying the other components in the TV using the Split layout (see Figure 9).

In this case our two-step solution offers a quality close to 90% (0.66/0.76 = 0.87) of the optimal.

3) USE CASE 3 - SIMULTANEOUSLY USING THREE DEVICES

If a viewer is consuming the 7 components using a TV, a laptop and a smartphone at the same time (as in Fig. 6), with the properties and values specified at the beginning of this sub-section, the two-step adaptation process provides the outcome shown in Fig. 10.

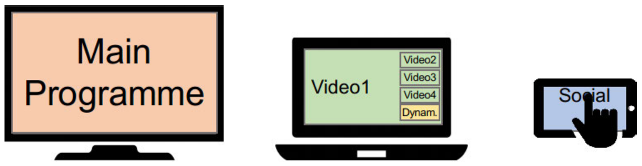


FIGURE 11. Diagram of the best possible adaptation outcome in use case 3.

TABLE 9. Results of the use cases.

Use case	Two-step $\mathcal{E}^{\mathcal{H}}$	Optimal $\mathcal{E}^{\mathcal{H}}$	Performance
Use case 1	0.28	0.28	100%
Use case 2	0.66	0.76	90%
Use case 3	0.81	0.85	95%

This scenario ends with an overall quality outcome of $\mathcal{E}^{\mathcal{H}} = 0.81$. It is interesting to note that with more devices, the outcome quality increases because there are components that fit (almost) perfectly to each of the devices and, with less components on each device, the quality of the representation improves. A discussion of how easy it is for a viewer to consume a content from several devices simultaneously could arise here. Thus, a parameter to penalise the use of multiple devices at the same time could be added, due to the different factors that could contribute to cognitive load, e.g., those listed in [46].

In use case 3, the overall optimal solution (shown in Fig. 11) results in a quality of $\mathcal{E}^{\mathcal{H}} = 0.85$. Therefore, in this case the two-step solution offers a quality of 95% of the optimal. While the optimal quality figure is marginally better than the one obtained with the two-step process, it requires evaluating ($N_D^{N_C} = 3^7 = 2187$) combinations, which is much more time and energy consuming, as shown in the next Subsection. As a summary, Table 9 shows the relative quality figures obtained by the two-step adaptation method.

4) EVALUATION OF THE COMPUTATIONAL EFFICIENCY

To analyse the computational efficiency, we ran specific performance tests using both the two-step adaptation process and the optimal solution search. Two scenarios have been analysed: scenario 1 considers 7 components, 3 devices, 5 properties per component, 5 properties per device, 3 layout types and 3 layout criteria; scenario 2 considers 10 components, 3 devices, 10 properties per component, 10 properties per device, 3 layout types and 6 layout criteria. Note that scenario 1 evaluates the efficiency of the Use Case 3 presented in the previous section while scenario 2 is a variation in order to measure the efficiency in more complex cases. Table 10 shows the time required by each device to provide a solution to the given scenarios.

In the first scenario, even if the two-step adaptation is much more efficient, neither method would impact the responsiveness of the user experience. However, in the second scenario, the times required to calculate the optimal solution dramatically increase, resulting in unacceptable poor responsiveness.

As a summary of the evaluation of the two-step adaptation process, the experiments show that, when parameterising the proposed methodology with reasonable figures, the outcome fits adequate multi-device UIs, mostly coinciding with the outcome of the specific adaptation rules developed by broadcasters and researchers during the application deployment in [13] and providing a responsive (although sub-optimal) outcome efficiently.

TABLE 10. Computational cost.

Device	Tablet	Laptop	TV
Latencies for two-step adaptation (s)			
Scenario 1	0.011	0.003	0.063
Scenario 2	0.026	0.006	0.137
Latencies for optimal adaptation (s)			
Scenario 1	0.671	0.240	1.196
Scenario 2	9.744	3.179	28.680

B. EVALUATION OF THE UNIVERSALITY OF THE METHODOLOGY

One of the goals of our adaptation methodology is to be general enough to be adapted to technological evolution and new broadcast trends or user habits while simplifying developers' work. Therefore, the adaptation process can be modified or oriented to other specific use cases and circumstances, while still following the proposed methodology. We have identified three main dimensions that could be added or modified:

- UI elements (types of components, devices or layouts) and properties to adapt the adaptation process to technological and broadcasters' changes;
- Evaluation criteria, to adapt the methodology to the user;
- Additional context parameters that can improve the user experience. These include parameters of user context (e.g., location in the room), physical context (e.g., ambient light), and infrastructure context (e.g., saturation of device capabilities due to an excess of assigned components).

In this sub-section, different examples are provided to evaluate its universality.

1) ADDING A NEW PROPERTY TO THE COMPONENT AND DEVICE TYPES

Consider, following the deployment performed in [13], that the *Dynamic Data* component shows vote counting through a complex 3D graph type and that the *Secondary Videos* have a high resolution with high decoding demands. In these case, strong graphic capabilities would be useful.

This will require an additional property to be considered, called *Graphic/video requirements*, for the component types, having a value of 1 for *Dynamic Data* and 0.3 for the *Secondary videos* (see Table 11). In the same way, an additional property could be considered for the device types, called *Graphic/video capabilities*, where TVs and mobile devices (both smartphones and tablets) would have a poor value (see Table 12) and laptops would have the highest possible value. The affinity matrix will emphasise the impact of the *Graphic/video requirements* with the *Graphic/video capabilities* (see Table 13).

TABLE 11. Components parametrisation for Graphic/video requirements.

Properties	Graphic/video Requirements
Main	0
Secondary videos	0.3
Banner	0
Static data	0
Dynamic data	1
Social	0
UGC	0
Advertisement	0

TABLE 12. Devices parametrisation for Graphic/video capabilities.

Properties	Graphic/video cap.
Smartphone	0.2
Tablet	0.2
Laptop	1
SmartTV	0

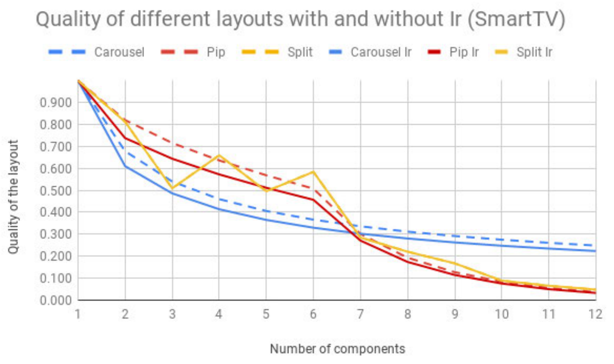


FIGURE 12. Quality of the layouts in terms of number of components with interaction criteria (straight line) and without (dashed).

With the aforementioned novel scenario, if we replicate the environment provided by *use case 3*, the outcome of the two-step adaptation shown in Figure 11, which is the optimal solution, is obtained instead of the previous one (shown in Figure 10), with a quality $\mathcal{E} = 0.85$.

Additionally, the appropriate combination of properties would allow the methodology to support other relevant use cases such as shared scenarios where different second screen contents are provided to family members watching the same show.

Apart from adding new properties to component and devices as explained above, new component types and device types could also be added. This would allow, for instance, to work with the increasingly adopted 3D formats and displays, smaller devices such as smartwatches or even smart speakers. New contents would be treated as additional component types in Table 1 and new displays as additional device types in Table 2. In that case, the property set for both components and devices will possibly be modified in Table 1 and 2. Finally, the affinity matrix in Table 4 would be extended to define the relation between the added properties. Therefore, the methodology allows extending component types, device types or properties without adding complexity.

2) ADDING A NEW CRITERION FOR USER INTERFACE LAYOUTS

Apart from the screen area related criteria used to select the best UI layout in each case, a broadcaster or content provider may want to obtain a balance between the interactivity required by the layout itself and the input capabilities of the target device. For instance, a carousel layout that requires the interaction of the viewer to spin the components will be the most difficult to manage with the remote control of a TV, while easiest with a touch-screen device. In the same way, PiP layouts could benefit from this interaction facilities. Regarding the Split layout, however, the interaction may not be a relevant parameter.

To take these aspects into consideration a fourth criterion α_4 has been considered, named *Interactivity compliance* and which depends on the layout property *Interactive Requirements* (Ir) added to those in Table 6 and applicable to Carousel and PiP layouts. This criterion has been weighted with a coefficient $w_4 = 1$.

Figure 12 shows the representation of the quality of the layouts in terms of the number of components for Smart TV devices, after adding the criterion of interaction of the layouts. As can be seen, the interaction affects to the result since the quality decreases for both the Carousel and PiP layouts, reinforcing Split as the best layout.

Other UI layouts criteria could also be considered, for example those oriented to assign bigger areas to some components against others. This could be done including a component type prioritisation and/or considering the resolution required by each component.

As a summary, this section have shown that the proposed methodology is based on an adaptation model that is efficient, flexible and adaptable to different multi-device media services. Therefore, the methodology could also be tuned to

TABLE 13. Affinity matrix considering Graphic/video capabilities and requirements.

	Screen size	Input cap.	Processing cap.	Privacy	Broadcast cap.	Graphic/video cap.
Attention	1	0	0	0.1	0	0
Interactivity	0.1	1	0.5	0.9	0	0
Processing requirements	0.3	0.3	1	0	0	0
Broadcast requirements	0	0	0	0	1	0
Confidentiality	0	0	0	1	0	0
Graphic/video requirements	0	0	0	0	0	1

different fields of applications beyond hybrid TV by modifying the sets of elements or properties and adjusting parameter values, according to the needs in each field.

VIII. CONCLUSION AND FUTURE WORK

This article proposes a methodology for multi-device User Interface adaptation in media services, such as a hybrid broadcast-broadband TV programme. The methodology is based on a universal adaptation model, that identifies and characterises all the elements of the User Interface for broadcast-broadband services, analyses the role of the context, and provides an efficient two-step implementation to make the adaptation process responsive.

The aforementioned research has allowed to provide a comprehensive example of how the methodology could be implemented. Reasonable parameters have been used for this, which provide an outcome that coincides with the output of the adaptation rules developed by the broadcasters and researchers during a real application deployment. Moreover, the quality, efficiency and universality have been evaluated and the following conclusions have been drawn:

- Suboptimal solutions obtained by the two-step adaptation process are reasonably good and meet the expectations of broadcasters and researchers. In the use cases tested they offer a quality of approximately 90% of the optimal solution.
- The two-step adaptation process is much more efficient than the search of the global optimal solution in terms of the processing time and therefore of the power consumption. Responsiveness and low latencies are guaranteed through two sequential steps, but not through the exploration of the entire solution tree in search of the optimal adaptation.
- The developed methodology is general enough to be extended to any type of content, device, context or evaluation criteria and ready for technological changes as well as continuous adaptive learning processes.

The methodology allows to simplify the adaptation process of hybrid broadcast-broadband services. This can be beneficial for broadcasters or content providers in other fields since their adaptation tasks will be reduced to: a) tag the contents to classify them in component types; and b) parametrize the properties or criteria they would like to consider. Once these tasks are accomplished the framework will be able to adapt the UI for many different scenarios automatically.

The exhaustive analysis of each parameter, intuitively or reasonably assigned, requires exploring research lines in the user interaction field, which is left for future work. Moreover, specific requirements or limitations could be added for each use case, such as the possibility of creating duplicated components during the *assignation*, as well as not showing specific non-critical components depending on the context. The methodology could easily assimilate these types of requirements by treating it as another component of the same type or removing a specific one.

An extensive user evaluation from the point of view of the TV viewer could be performed to adjust parameters and coefficients. Performing such an evaluation is probably premature, as multi-device broadcast-broadband applications are still uncommon and few users are familiar with them. When these types of applications are commonplace, it will be easier to select a user base to validate the presented methodology. Additionally, this could lead to learning processes that allow for modifying or refeeding the adaptation process with context information: the interaction of the viewers in general, personalisation for each viewer, analysing how the environmental factors impact the adaptation preferences, etc.

Finally, the methodology proposed in the present work has validated the underlying adaptation model in hybrid broadcast-broadband scenarios. However it could also be applied to completely different fields, such as Human-Computer Interfaces for Industry 4.0 [37] or decision-making videowalls.

REFERENCES

[1] L. Claudy, "The broadcast empire strikes back," *IEEE Spectr.*, vol. 49, no. 12, pp. 52–58, Dec. 2012.
[2] M. Obrist, P. Cesar, and S. Basapur, "Forward to the theme issue on interactive experiences for television and online video," *Pers. Ubiquitous Comput.*, vol. 19, nos. 5–6, pp. 741–742, Aug. 2015.
[3] M. McGill, J. H. Williamson, and S. A. Brewster, "A review of collocated multi-user TV," *Pers. Ubiquitous Comput.*, vol. 19, nos. 5–6, pp. 743–759, Aug. 2015.
[4] T. Neate, M. Jones, and M. Evans, "Interdevice media: Choreographing content to maximize viewer engagement," *Computer*, vol. 49, no. 12, pp. 42–49, Dec. 2016.
[5] A. Bulling, "Pervasive attentive user interfaces," *Computer*, vol. 49, no. 1, pp. 94–98, Jan. 2016.
[6] F. Carroll, "Exploring past trends and current challenges of human computer interaction (hci) design: What does this mean for the design of virtual learning environments?" in *User Interface Design for Virtual Environments: Challenges and Advances*. Hershey, PA, USA: IGI Global, 2012, pp. 60–75.
[7] M. Bauerly and Y. Liu, "Effects of symmetry and number of compositional elements on interface and design aesthetics," *Int. J. Human-Computer Interact.*, vol. 24, no. 3, pp. 275–287, Mar. 2008.
[8] A. Altaboli and Y. Lin, "Investigating effects of screen layout elements on interface and screen design aesthetics," *Adv. Hum.-Comput. Interact.*, vol. 2011, May 2011, Art. no. 659758.
[9] A. Oulasvirta, "User interface design with combinatorial optimization," *Computer*, vol. 50, no. 1, pp. 40–47, Jan. 2017.
[10] (Dec. 2012). *NEM (Networked & Electronic Media) Connected TV Position Paper*. Accessed: Oct. 14, 2020. [Online]. Available: https://nem-initiative.org/wp-content/uploads/2013/12/NEM-PP-015.pdf
[11] *W3C Standards*. Accessed: Nov. 9, 2020. [Online]. Available: https://www.w3.org/standards/
[12] M. Zorrilla, N. Borch, F. Daoust, A. Erk, J. Flórez, and A. Lafuente, "A Web-based distributed architecture for multi-device adaptation in media applications," *Pers. Ubiquitous Comput.*, vol. 19, nos. 5–6, pp. 803–820, Aug. 2015.
[13] A. Dominguez, M. Agirre, J. Florez, A. Lafuente, I. Tamayo, and M. Zorrilla, "Deployment of a hybrid broadcast-Internet multi-device service for a live TV programme," *IEEE Trans. Broadcast.*, vol. 64, no. 1, pp. 153–163, Mar. 2018.
[14] *A Multiscreen Experience for a Elections Night TV Programme*. Accessed: Oct. 7, 2020. [Online]. Available: https://www.youtube.com/watch?v=NwixgA_M144
[15] A. Dominguez, "Optimisation of the user experience across multi-screen media services," Ph.D. dissertation, Distrib. Syst. Group, UPV/EHU Univ., San Sebastian, Spain, 2020. [Online]. Available: https://addi.ehu.es/handle/10810/46221

[16] *Flexible Solution for Multimedia Information Management*. Accessed: Oct. 7, 2020. [Online]. Available: https://www.youtube.com/watch?v=s0p_k_hKPGo

[17] *Adaptation Model Implementation*. Accessed: Oct. 14, 2020. [Online]. Available: https://github.com/tv-vicomtech/adaptationModel

[18] J. Bryant and M. Jones, "Responsive Web design," in *Pro HTML5 Performance*. Cham, Switzerland: Springer, 2012, pp. 37–49.

[19] *Media Queries W3C Recommendation*. Accessed: Oct. 14, 2020. [Online]. Available: https://www.w3.org/TR/mediaqueries-4/

[20] *Web Components W3C Specification*. Accessed: Oct. 14, 2020. [Online]. Available: http://w3c.github.io/webcomponents/

[21] M. Armstrong, M. Brooks, A. Churnside, M. Evans, F. Melchior, and M. Shotton, "Object-based broadcasting-curation, responsiveness and user experience," BBC, London, U.K., Tech. Rep. WHP 285, 2014.

[22] M. Nebeling, "XDBrowser 2.0: Semi-automatic generation of cross-device interfaces," in *Proc. CHI Conf. Hum. Factors Comput. Syst.*, 2017, pp. 4574–4584.

[23] M. Nebeling and A. K. Dey, "XDBrowser: User-defined cross-device Web page designs," in *Proc. CHI Conf. Hum. Factors Comput. Syst.*, 2016, pp. 5494–5505.

[24] C. N. Klokmose, J. R. Eagan, S. Baader, W. Mackay, and M. Beaudouin-Lafon, "Webstrates: Shareable dynamic media," in *Proc. 28th Annu. ACM Symp. User Interface Softw. Technol.*, 2015, pp. 280–290.

[25] M. Sarkis, C. Concolato, and J.-C. Dufourd, "A multi-screen refactoring system for video-centric Web applications," *Multimedia Tools Appl.*, vol. 77, no. 2, pp. 1943–1970, Jan. 2018.

[26] R. Rädle, H.-C. Jetter, N. Marquardt, H. Reiterer, and Y. Rogers, "Huddlelamp: Spatially-aware mobile displays for ad-hoc around-the-table collaboration," in *Proc. 9th ACM Int. Conf. Interact. Tabletops Surf.*, 2014, pp. 45–54.

[27] M. Schreiner, R. Rädle, H.-C. Jetter, and H. Reiterer, "Connichiwa: A framework for cross-device Web applications," in *Proc. 33rd Annu. ACM Conf. Extended Abstr. Hum. Factors Comput. Syst.*, 2015, pp. 2163–2168.

[28] L. Frosini and F. Paternò, "User interface distribution in multi-device and multi-user environments with dynamically migrating engines," in *Proc. ACM SIGCHI Symp. Eng. Interact. Comput. Syst. (EICS)*, 2014, pp. 55–64.

[29] J. Yang and D. Wigdor, "Panelrama: Enabling easy specification of cross-device Web applications," in *Proc. SIGCHI Conf. Hum. Factors Comput. Syst.*, 2014, pp. 2783–2792.

[30] T. Horak, A. Mathisen, C. N. Klokmose, R. Dachselt, and N. Elmqvist, "Vistribute: Distributing interactive visualizations in dynamic multi-device setups," in *Proc. CHI Conf. Hum. Factors Comput. Syst.*, New York, NY, USA, 2019.

[31] D. Gómez, J. A. Núñez, M. Montagud, and S. Fernández, "ImmersiaTV: enabling customizable and immersive multi-screen TV experiences," in *Proc. 9th ACM Multimedia Syst. Conf.*, 2018, pp. 506–508.

[32] F. Boronat, D. Marfil, M. Montagud, and J. Pastor, "HbbTV-compliant platform for hybrid media delivery and synchronization on single-and multi-device scenarios," *IEEE Trans. Broadcast.*, vol. 64, no. 3, pp. 721–746, Sep. 2018.

[33] C. Gavrila, V. Popescu, M. Fadda, M. Anedda, and M. Murroni, "On the suitability of HbbTV for unified smart home experience," *IEEE Trans. Broadcast.*, early access, Mar. 16, 2020, 10.1109/TBC.2020.2977539.

[34] C. Gavrila, V. Popescu, M. Alexandru, and M. Fadda, "Unifying the smart home experience through HbbTV-enabled devices," in *Proc. IEEE Int. Symp. Broadband Multimedia Syst. Broadcast. (BMSB)*, Jun. 2019, pp. 1–5.

[35] H. Ajam and M. Mu, "A middleware to enable immersive multi-device online TV experience," in *Proc. Adjunct Publication ACM Int. Conf. Interact. Experiences TV Online Video (TVX Adjunct)*, 2017, pp. 27–32.

[36] S. Oh, A. Kim, S. Lee, K. Lee, D. R. Jeong, S. Y. Ko, and I. Shin, "FLUID: Flexible user interface distribution for ubiquitous multi-device interaction," in *Proc. 25th Annu. Int. Conf. Mobile Comput. Netw.*, 2019, pp. 1–16.

[37] J. Posada, M. Zorrilla, A. Dominguez, B. Simoes, P. Eisert, D. Stricker, J. Rambach, J. Dollner, and M. Guevara, "Graphics and media technologies for operators in industry 4.0," *IEEE Comput. Graph. Appl.*, vol. 38, no. 5, pp. 119–132, Sep. 2018.

[38] P. Diaz, T. Onorati, and S. del Olmo Pueblas, "Analyzing and visualizing emergency information in a multi device environment," in *Proc. Int. Conf. Inf. Syst. Crisis Response Manage. Medit. Countries*. Cham, Switzerland: Springer, 2016, pp. 181–194.

[39] M. Zorrilla, I. Tamayo, A. Martin, and I. G. Olaizola, "Cloud session maintenance to synchronise HbbTV applications and home network devices," in *Proc. IEEE Int. Symp. Broadband Multimedia Syst. Broadcast. (BMSB)*, Jun. 2013, pp. 1–6.

[40] A. Dominguez, I. Tamayo, M. Zorrilla, J. Florez, and A. Lafuente, "Componentizing a hybrid broadcast-Internet multi-device media service," in *Proc. IEEE Int. Symp. Broadband Multimedia Syst. Broadcast. (BMSB)*, Jun. 2018, pp. 1–6.

[41] A. Dominguez, J. Florez, A. Lafuente, S. Masneri, I. Tamayo, and M. Zorrilla, "Methods for device characterisation in media services," in *Proc. ACM Int. Conf. Interact. Experiences TV Online Video (TVX)*, 2019, pp. 118–128.

[42] *Designing for Second Screens: The Autumnwatch Companion*. Accessed: Oct. 14, 2020. [Online]. Available: http://www.bbc.co.uk/blogs/researchanddevelopment/2011/04/the-autumnwatch-companion—de.shtml

[43] W. Nagel, *Multiscreen UX Design: Developing for a Multitude Devices*. San Mateo, CA, USA: Morgan Kaufmann, 2015.

[44] A. Sears, "Layout appropriateness: A metric for evaluating user interface widget layout," *IEEE Trans. Softw. Eng.*, vol. 19, no. 7, pp. 707–719, Jul. 1993.

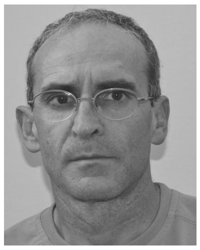
[45] J. Hussain, A. Ul Hassan, H. S. Muhammad Bilal, R. Ali, M. Afzal, S. Hussain, J. Bang, O. Banos, and S. Lee, "Model-based adaptive user interface based on context and user experience evaluation," *J. Multimodal User Interfaces*, vol. 12, no. 1, pp. 1–16, Mar. 2018.

[46] S. G. Hart and L. E. Staveland, "Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research," *Adv. Psychol.*, vol. 52, pp. 139–183, 1988.

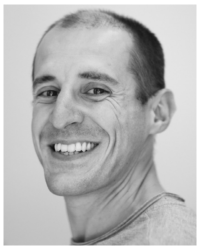
[47] M. Zorrilla, I. Tamayo, A. Martin, and A. Dominguez, "User interface adaptation for multi-device Web-based media applications," in *Proc. IEEE Int. Symp. Broadband Multimedia Syst. Broadcast. (BMSB)*. IEEE, 2015, pp. 1–7.

ANA DOMÍNGUEZ received the degree in telecommunication engineering from Tecnun, University of Navarra, Spain, and the Ph.D. degree from the University of the Basque Country, Spain, in April 2020. Her Ph.D. thesis was on Optimisation of the User Experience Across Multi-Screen Media Services. Since 2015, she has been working with Vicomtech, where she is currently with the Department of Digital Media. Her research interest includes interactive media technologies.

JULIÁN FLÓREZ received the degree in industrial engineering from the University of Navarra, in 1980, and the Ph.D. degree in adaptive control from the University of Manchester Institute of Science and Technology (UMIST), in 1985. He is currently a Professor of control systems engineering with TECNUN, University of Navarra. Since 2001, he has been the General Director of Vicomtech. He has a strong background in digital television and telecom infrastructures.



ALBERTO LAFUENTE is currently an Associate Professor with the Computer Architecture and Technology Department, UPV/EHU. He is also a Co-Founding Member with the Distributed Systems Group, UPV/EHU. His recent research interests include distributed systems and algorithms, dependable computing, pervasive systems, wireless sensor networks, and mobile systems.



STEFANO MASNERI received the B.Sc. degree in information technology and the M.Sc. degree in telecommunications from the Università degli studi di Brescia, Italy, in 2005 and 2008, respectively. He is currently with the Department of Digital Media, Vicomtech. His research interests include signal processing, computer vision, and interactive technologies.

IÑIGO TAMAYO received the degree in computer science engineering from the University of Mondragon, Spain, in 2007, and the Advanced degree in computational engineering and intelligent systems from the University of Basque Country, Spain, in 2017. He is currently with the Department of Digital Media, Vicomtech. His research interests include distributed computing and web technologies.

MIKEL ZORRILLA received the Telecommunication Engineering degree from the University of Mondragon, Spain, and the Ph.D. degree from the University of the Basque Country, Spain, in September 2016. His Ph.D. thesis was on Interoperable Technologies for Multi-Device Media Services. He is currently with the Department of Digital Media, Vicomtech. He is also the Head of the Digital Media Department.