A Roadmap for User Interface Design of Interactive Systems: An Approach Based on a Triad of Patterns

Alexandra Ruíz[1(✉)], William J. Giraldo[1], David Geerts[2], and Jose L. Arciniegas[3]

[1] Universidad del Quindío, Armenia, Colombia
{aruiz,wjgiraldo}@uniquindio.edu.co
[2] KU Leuven, Leuven, Belgium
david.geerts@kuleuven.be
[3] Universidad del Cauca, Popayán, Colombia
jlarciniegas@unicauca.edu.co

Abstract. This article presents a Roadmap for user interface designing based on a triad of patterns (data, interaction, and presentation) that comprehends the principles and guidelines from MBUID, HCI, and DCU with the purpose of promoting usability on the final interface. The roadmap is supported by a method, a tool, and languages that allow interface designing. This study is focused on the method only, and it presents the activities and artifacts in detail through its application in a case study in the educational context.

Keywords: User interface design · Patterns · Model driven · Usability

1 Introduction

The User Interface (UI) is considered one of the main components of the development of interactive systems because it connects end users with functionality. Due to its importance and the increasingly demanding requirements to reach to solutions with the quality, usability, and user experience required, the development of the UI has become complex. For this reason, developers and designers must reuse the knowledge already gained by professionals in previous processes, so that they do not have to think about how to solve design problems that have been solved already. For example, certain presentation structures can be common for multiple businesses (sales, purchases, customers), either in the same or different domains. However, a developer who does not have enough experience would have to generate from scratch the interface prototypes and the data modeling that supports them; unless, there is the possibility of using data and UI patterns to generate different proposals of the final interface without much effort.

On the other hand, the complexity of the development of the UI is also due to the heterogeneity of the deployment devices, use contexts, and users, which represent for the software companies more development time of the interactive applications; thus, a greater monetary investment. One of the approaches that have emerged to solve this problem is the Model Based User Interface Design (MBUID). It is an approach that aims to address the problems of heterogeneity related to devices, users, and use contexts, reducing the necessary effort for the development of the UI.

From the benefits offered by MBUID and the implementation of patterns, the contributions from this article focus on proposing a methodological path that simplifies the work for the developer to generate UI from a triad of patterns (data, interaction, and presentation) that fit the mental model acquired by the user from his own tasks. The proposed path is constructed from a set of proposals that include the principles and the main ideas of the MBUID and the HCI, with the purpose of promoting usability in the final interface. This article is then structured by the following sections: Sect. 2 shows the related works in the context of MBUID and the use of patterns, Sect. 3 presents the fundamentals that characterize the Roadmap, Sect. 4 specifies the Roadmap life cycle and its flow of activities, Sect. 5 presents the application of the Roadmap in a specific case study; and finally, Sect. 6 presents the conclusions and future work.

2 Related Works

The literature reports a variety of approximations based on MBUID, which, according to their characteristics, have been classified into four generations [1]. The first generation is characterized by the development of the UI from a universal model. The second one was characterized by the extension of UI models through the integration of other models. In the third one, developers and designers had to focus on developing user interfaces for different devices with different restrictions. In that generation, MBUID becomes relevant. The fourth generation focuses on the development of context sensitive UI for platforms, devices, and different modalities. Finally, it is observed that as of 2012, more proposals that contemplate the use of patterns have emerged; this way, possibly emanating a fifth generation of MBUID tools. The Fig. 1 presents the MBUID timeline with the most representative proposals of each generation. The graph is an updated version of the one that can be found in [1].

From this wide list of approaches, those that have available information or were current were analyzed. The proposals are: ITS [2], AME [3], MECANO [4], MOBI-D [5], TERESA [6], Supple [7], Maria [8], CIAF/TD_MBUID [9], Cedar [10], MyUI [11], PamGis [12], LIZARD [13] and MODUS [14]. The last four (4) proposals use patterns as a means to provide a certain degree of automation at the usability level. Such is the case of MyUI, a proposal for the generation of individual user interfaces from an interaction model. In this proposal, adaptations are made for different user needs, devices, and environmental conditions, during the execution time. To do this, it uses an extensible repository of design patterns, which includes adaptation rules and modular building blocks for the generation of user interfaces.

On the other hand, PamGis aims to provide a high degree of automation for the generation of artifacts from abstract (UML/XML) and semi-abstract (XML) application models to the source code of the resulting application, through the information inherent to task, user, device, context, and pattern language models.

The central component of the framework is a repository of patterns that contain different types of patterns and pattern languages of different levels of abstraction,

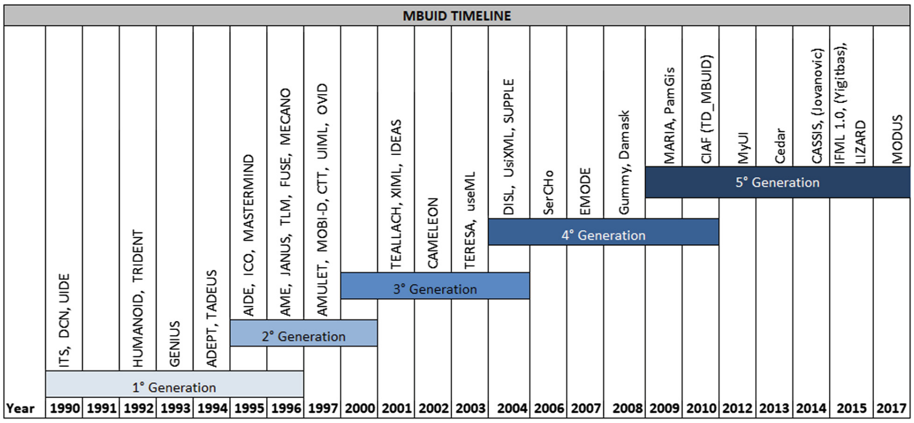


Fig. 1. MBUID [1] proposals timeline, updated version

architecture patterns, design patterns, and HCI patterns. LIZARD allows the definition of applications at a high level of abstraction, based on the specification of domain, task, presentation, device, and data access models. This proposal applies model transformations to generate the target native UI considering the specific features of target platforms. The generated applications follow the UI design guidelines and the architectural design patterns specified by the corresponding operating system manufacturer. Finally, MODUS proposes a tool supported approach for user interface generation directly from the architectural models of the business logic. This proposal transforms the domain model into an ontology in Prolog in order to perform queries to identify data patterns previously stored in a patterns knowledge base. Once the patterns have been selected, the concrete interface is generated and subsequently the final interface, based on transformations of the abstract interface previously associated with each pattern.

The UI development processes of the proposals mentioned have focused on providing a certain degree of usability in the interface through the use of patterns. However, the process is not clear in relation to how integration with the development of other aspects such as functionality is made. In our case, we propose a roadmap for the design of the UI based on a triad of patterns that aims; in addition to promoting usability in the final interface, to the integration with the functionality of the interactive system.

3 Roadmap Fundamentals

The fundamentals according to the Roadmap conception, approach, and principles are presented below.

3.1 How Is the Roadmap Built?

The proposed methodological approach finds its essence in existing proposals in the MBUID and HCI contexts. Due to the degree of contribution of the analyzed proposals, the work of the Collaborative Interactive Application Framework (CIAF) [9] is highlighted. It proposes a methodological approximation for the development of Model Based User Interfaces in collaborative environments. The proposal is based on the use of several models to represent collaborative and interactive aspects. For this reason, different notations and techniques are used. The integration of three notations is described particularly: 1. CIAN- which involves aspects of human-computer collaboration and interaction, 2. UML- which specifies the functionality of computer systems, and 3. usiXML- which describes the user interface for multiple platforms. Additionally, how the model is integrated within the Software Engineering process is described. The process of model development and diagram integration is supported by Eclipse tools. CIAF provides a framework of conceptual, methodological and technological development for the advance of interactive systems that can be used directly, extended, or customized. This development framework is formally represented. There is a proposal that considers the development of functionality (OpenUP and UML4), group work (CIAM and CIAN5), and user interface (TD-MBUID and usiXML) with languages and tools to obtain user interfaces. These are the three methodological proposals that have been combined to form a single framework, each one managing a different aspect of the development of interactive systems. Each methodological proposal encapsulates its own roles, activities, artifacts, best practices, and tools, allowing being used in isolation.

Relying on the CIAF development framework as the main foundation, we propose a roadmap that expands and customizes the CIAF for the development of the UI from a triad of patterns. Although the roadmap is based on the CIAF, it adds up artifacts, languages, techniques, tools, and models that satisfy the specific necessities of the UI design from a triad of patterns in the context of a specific domain. Thus, a path is obtained which is supported by three essential components: a method, the notation and modelling tools, and code generators for a specific domain.

3.2 What Are the Roadmap's Pillars?

The main pillar of the roadmap is an engineering approach based on a triad of patterns that seeks to simplify to the developer of the interface, the interpretation of the mental models that the expert and the user have, and the way they interact with that data; in order to promote usability in the generated interface through modeling tools and code generators. Through this approach, it is sought that the development of interfaces does not start from scratch, in such a way that it solves the problems of fast functional prototyping and viable minimum product.

Additionally, the proposed roadmap is based on the pillars of the MBUID, the DCU and the design of the UI. The roadmap shares the philosophy of MBUID, in the sense that the models are the essential artifacts in the development process of the UI. Unlike the abstraction levels described in the CAMELEON [15] framework used by some MBUID proposals, the proposed path does not include the abstract user interface (AUI) or modal-independent interface, at least not in an automated way, since the modality is selected at the beginning of the path.

Similarly, the Roadmap shares the DCU [16] principles, such as: (i) the representative users actively participate from early stages and through the whole development process and the entire life cycle of the UI; (ii) the patterns and the UI are built from an iterative and incremental process and (iii) it raises empirical measures of the use of the product based on the evaluation of experts and representative users.

Finally, the roadmap is based on the pillars for UI designing proposed by [17], which argues that the design of the UI must have three elements: processes and guidelines documents (theories and models), software tools of user interface (algorithms and prototypes), and a review by experts and usability tests (controlled experiments). This way, a process is proposed that promotes usability through prototyping and evaluation techniques and that it is supported by model editing tools and automatic code generators for the development of the UI.

What is a Triad of Patterns and How Is it Built?

In the context of this study, a pattern is a model that represents a repetitive solution to a specific problem domain; that is to say, it describes and specifies how a particular problem can be solved, whether it is a model, an implementation, or another abstraction level [18]. In this regard, the triad of patterns is formed by association of a data pattern (domain model), interaction pattern (interaction model), and presentation pattern (presentation model). Their concepts are presented below:

- *Data pattern*: they describe the concepts and data that support the business natural tasks. In this sense, it is sought to specify modelling solutions that simplify the interpreting of mental models by the user [19]. The data pattern is represented by a data model using the UML class diagram notation.
- *Presentation pattern*: it is represented by DataForm model, which allows to express the form semantic (user interface) and the data semantic. The DataForm model is built from how the user structures and perceives the information [20].
- *Interaction pattern*: it is represented by the dialog pattern [9], which preserves the structure and temporary relations of the CTT (*Concur Task Tree*) [21], notation, specifying for each task, the type of Abstract Interaction Components (AIC) associated with the task. These interaction components represent a canonic expression of the rendering and the manipulation of the domain and function concepts in a way of being the most independently possible from the modality (auditive, visual, tactile, etc.) and the informatic platform [22].

The construction of the triad of patterns begins with an exploration of the data patterns for a specific domain. These data have been recognized; assessed by experts that recognize that they have that structure, those elements, and those relationships (expert's mental model), but that it has never been discussed how they must be presented. For this reason, the presentation and interaction patterns (user's mental model) are built from an iterative and incremental process that starts from a user's study, its surroundings, and the task that are done by a specific context and the application of different lab tests with the participation of representative users. Then, the triad of patterns is evaluated by experts and representative users from interface prototypes that are built having in mind different use scenarios, which represent the pattern variability. As a result, a pattern catalog is obtained that can be used for solving design problems for a specific domain.

3.3 Why Those Three Patterns?

In the development of an interactive system, different quality attributes intervene, such as functionality, usability, security, communication, collaboration, etc. Each system attribute requires specific models that describe the information relevant to that attribute. Consequently, a large variety of models are required to develop software.

In our case, we propose an engineering approach that requires only three patterns (models) for the development of the user interface but that uses different models for the construction of each pattern. In this way, the information concerning each quality attribute required is taken into account in that pattern.

The selection of the three types of patterns or models is based on the principles of the development of the functionality and the user interface. Functionality is a process that is developed in parallel with the user interface, which are synchronized. It is related to the domain data (expert opinion), which support the tasks, while the interface is related to how to present and interact with that data but from the user's perspective. In this way, three models (data, presentation and interaction) are required to represent that information.

In the following section, the lifecycle of the proposed methodological roadmap and the main activities of each of its phases are exposed.

4 Proposed Roadmap

4.1 Lifecycle

The Roadmap lifecycle (Fig. 2) is partitioned in the same stages of the software engineering traditional cycle (requirements, analysis and design, implementation, and testing), in such a way that it is easily paired with the development of functionality contemplated in the software engineering conventional methodologies. Likewise, Fig. 2 shows how the user participates actively from early stages and throughout the process, which is iterative and incremental and relies on prototyping and evaluation techniques as one of the mechanisms to promote usability in the final interface.

In the following section, the detailed methodological content of the proposed Roadmap is exposed.

4.2 Detailed Methodological Content

The proposed Roadmap has two main flows: one for creating patterns for a specific domain and the other for the development of the user interface using the created patterns. In Fig. 3 the flow of activities for both cases is observed. According to the stages that make the path, every flow is detailed next.

Flow of Activities for the Creation of Patterns. The flow for the creation of the patterns starts at the plan and research phase and ends at the analysis and design stage. The purpose of this route is to settle the catalog of patterns in such a way that they can be applied to a specific context later. The activities of this flow according to the stages of the Roadmap are presented below.

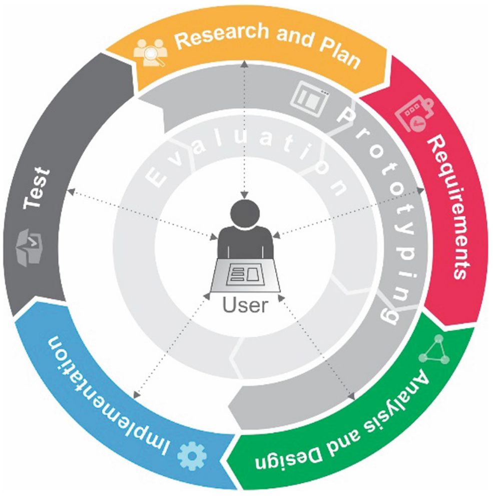


Fig. 2. Roadmap lifecycle

Research and Plan Phase. This stage is aimed at foreseeing the user-centered activities that will be carried out throughout the creation of the patterns and capturing information about the domain of the problem, the user and his tasks, and the product or similar interactive systems. Based on this information, an initial panorama of the context of the type of interactive systems to be designed will be created that will provide information for decision making regarding how the activities of the following stages will be approached. The resulting artifacts of this stage are: usability plan, theoretical frame, data patterns, competitive analysis, user profile, heuristics, and design patterns as presented.

- *Create the usability plan*: it consists of constructing a document that specifies the user-centered activities that will be carried out throughout the construction of the patterns. The plan is a living document that is refined as progress is made in the stages.
- *Know the domain problem*: this activity explores and collects the concepts that characterize the selected domain, in order to identify data patterns that describe the business.
- *Review with experts*: the data patterns found are reviewed with experts, in order to validate domain data in relation to whether they are understandable and complete, both at the concept level and at relationships.
- *Know the product*: tasks that will allow a better understanding of the type of interactive systems to be designed are carried out. For this, there are different techniques that will allow to capture this information such as competitive analysis, literary reviews, user tests, among others. The purpose of conducting activities to get to know the product is to be able to identify main functionalities, design guidelines (e.g. heuristics) and patterns (data, presentation or interaction) already defined for the context of the interactive systems that are being addressed.

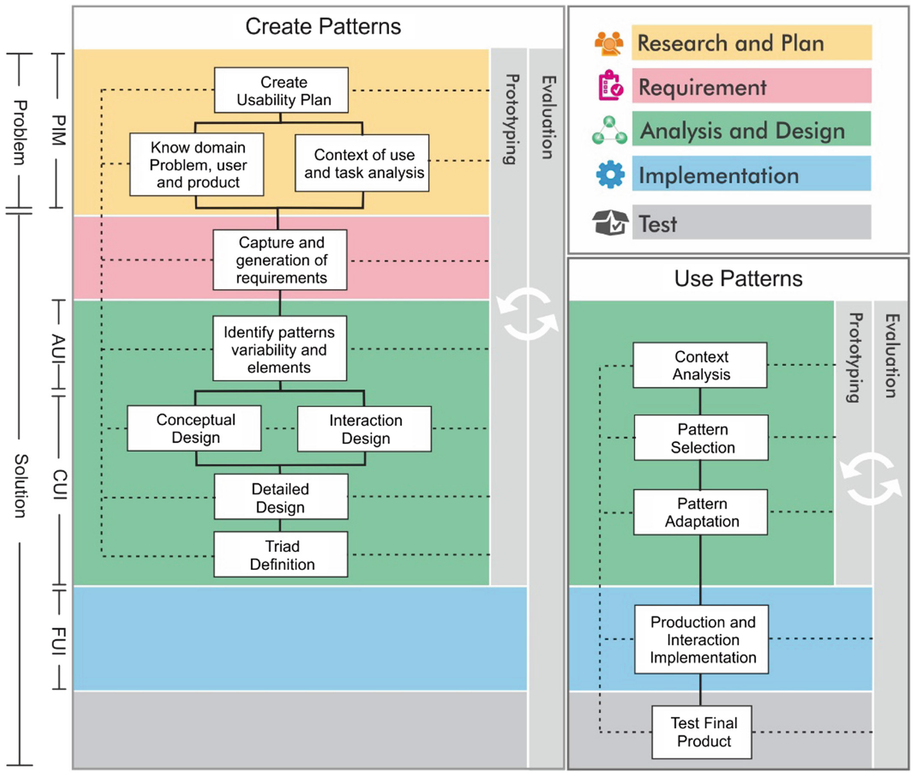


Fig. 3. Detailed methodological content of the proposed Roadmap

- *Know the user*: Finally, once the product is studied, it is necessary to know the potential users of that product. Then, techniques such as interviews, surveys, observation, among others, are applied. The purpose of the application of these techniques is to generate information related to the user's profile, a document that describes and classifies the different potential users in terms of their main characteristics such as skills, limitations, demographic information, appreciations or motivations, etc.
- *Context of use and task analysis*: this activity captures information about the task, how they currently interact with similar systems, what problems they have, what their expectations are, identified needs, etc.

Requirement Phase: As a result of the analysis of the information captured in the previous stage, additional activities are defined or not for the capture of requirements. Subsequently, the information is analyzed and a requirements document containing the user and usability requirements is generated.

Analysis and Design Phase: In this stage, criteria that make the pattern presentation vary are identified. For each variability of the pattern, its Abstract Interface (AUI) is constructed. Afterward, conceptual ideas are generated, and the first prototypes of the pattern presentation are created, which are evaluated to capture the interaction pattern (how the user wishes to interact with the data). Finally, the detailed design is done, and the three patterns are created in the tool: data pattern, presentation pattern (Concrete Interface - CUI) and interaction pattern. The artifacts that are generated in this stage are: validated data, presentation, and interaction patterns, variability tree, sketches, prototypes. The activities of this stage are detailed as follows:

- *Identify patterns variability*: the criteria that can make the pattern vary according to the information related to the use context, task analysis, user profiles, and user requirements are identified. The criteria can be varied and depend on the context that is being addressed.
- *Identify patterns elements*: it corresponds to specify the Abstract User Interface - AUI which is equivalent to identifying the elements that are part of the presentation pattern, according to the selected criteria.
- *Conceptual Design*: in this activity, conceptual ideas are generated for each variability of data pattern. The result can be texts, videos, scenarios, or moodboard. The tasks of this activity are: generate design ideas, create design ideas (images, moodboard, scenarios, sketches, and text documents) and apply metaphors.
- *Interaction Design*: here the first sketches of the data patterns are created, and design patterns are applied if they exist for the context that is being addressed. The evaluation that takes place in this activity is aimed at capturing the interaction pattern and validating the presentation data by the final user. The tasks of this activity are: content and layout design, apply existing design patterns, prototyping, and evaluation.
- *Detailed design presentation*: the detailed design is done here; that is, gestalt laws, heuristics, and other design guides oriented to promote usability in the interface are applied. The evaluation of the prototypes is oriented to evaluate the usability and validate the presentation of the pattern. The tasks of this activity are: apply heuristics or design guidelines, prototyping (digital prototyping), evaluation (usability evaluation, validate patterns presentation).
- *Triad definition for each pattern*: based on the information captured in the previous activities, the triad for each variability of the pattern is created using the model editor. The tasks for this activity are: create data pattern, create interaction pattern, and create presentation pattern.

Flow of Activities Using the Pattern Catalog. The flow for the development of user interfaces using the patterns starts at the analysis and design stage and ends at the testing stage. Below are the activities of this flow according to the Roadmap stages.

Analysis and Design. In this stage, the patterns are selected and adapted according to the business concepts and the context data of the problem. The activities of this stage are:

- *Context Analysis*. The domain of the application is studied to identify the context of the problem; that is to say, to identify the user, the tasks to be performed, and the business concepts and data that are manipulated to execute them. The result of this activity is a general description of the business concepts (Entities) and the context data of the problem.

- *Pattern Selection.* In this activity, the data patterns are selected according to the identified business concepts. The selection of a pattern can be done in different ways: (1) select a pattern from the catalog, (2) create a pattern from the catalog patterns, or (3) create a completely new pattern for the same domain. For the creation of new patterns in the same domain, it is necessary to prototype and evaluate in order to validate the associated presentation of the pattern. Once the data pattern is selected, the variability that fits the problem context is identified for each pattern. Finally, the presentation and interaction pattern associated according with the data that the user requires to perform the task is selected.
- *Pattern Adaptation.* According to their nature, the patterns can be adapted to specific situations of a specific problem; although in the data patterns catalog, possible variations are defined to support different contexts, cases may arise where specific data that were not contemplated are required. Consequently, this activity must validate that the pattern fully supports the user's tasks and data; and if necessary, add or remove data that are not represented, modifying the domain model (initially data pattern) and its respective presentation (CUI) that is obtained from the previous activity.

Implementation Phase. In this stage, the implementation of the productions and the interaction of the CUI is carried out to proceed with its generation. Interaction refers to performing aspects of programming logic that allow exposing functionalities in the prototypes to be generated, which allow defining basic aspects of interaction; for example, autocomplete a text field. Otherwise, the productions refer to generating a set of real data test, instances of the classes of the domain model that supports the interface; which means, generating real data of the business concepts involved. This is done in order to validate the interface with the client early. At the end of this task, the Final Interface (FUI) associated with the selected variability is generated automatically.

Test Phase. In this stage, tests are executed in order to verify the access and interaction to the elements of the screens and the data that are required to carry out each task in the specific context. In other words, the usability of the final interface in the interactive system is evaluated. The resulting information from this evaluation is used as feedback of the triad of patterns.

5 Case Study

As a case study, we propose the development of a web application that assists teachers and students in the teaching-learning process in such a way that it is easy to use by the average teacher and can be paired to a Learning Management System (LMS), specifically to the Moodle platform. The purpose of this application is to provide a didactic strategy model, better known in the jargon of instructional design as "sequence of activities," which allows the teacher to plan their teaching-learning activities in a way that they focus on their work and not in the use of technology. The model allows creating sequences of different didactic activities according to student profiles, previous knowledge, etc. as well as other advantages [23]. The main idea of applying the

Roadmap in this case study is to be able to design the interfaces that will support the didactic activities of the model. As a result, a set of patterns is generated in the educational context that can be used later for the design of interfaces of different interactive systems within the same domain.

Taking into account the interactive system to be developed, we will begin the Roadmap exploration from the activity flow for creating patterns. Later, the created patterns will be used for the development of the interfaces of the proposed interactive system.

5.1 Activity Flow for the Creation of Patterns in the Educational Context

In the next sections, a summary of the information obtained for every single activity of the flow is presented; the emphasis is on the artifacts directly related to triad of patterns.

Create Usability Plan. The flow for the creation of patterns begins with the creation of the usability plan, in which the following activities were considered: (1) literary reviews to identify the relevant concepts (data patterns) around the didactic act, (2) competitive analysis to get to know similar products, (3) application of surveys to get to know the users profile, (4) application of field journals and contextual observation to get to know about the tasks and the use context, (5) implementation of a workshop to define the AUI, and (6) users tests to validate the presentation and interaction of the pattern. Activities 5 and 6 were defined according to the results of the previous activities.

Know Domain Problem. Once the plan was defined, the execution of a literary review began in order to identify the main concepts and their relationships around the didactic act. The main concepts around the subject that originate from the analysis of different conceptual models of the didactic act such as the ones proposed by Marquès [24], Meneses [25], Rodríguez [26], among others are presented below.

According to Marquès [24], teaching is achieved by the means of didactic acts through which the teacher (Role) or trainer proposes multiple activities to students (Role) to aid in the desired learning. These activities, known as learning activities (LearningActivity), are adapted to the characteristics of the students, the available resources, and the contents under study. However, the teaching process is not only composed of learning activities, but also support activities (SupportActivity) [27], which are usually carried out by the teacher to support the students. Marquès suggests that the effectiveness of the teaching resources will depend to a large extent on the way the teacher guides its use within the didactic strategy (DidacticStrategy) framework that he is using. A strategy is an organized procedure oriented towards the attainment of a clearly established goal. In this way, the didactic strategy is the procedure by which the teacher intends to assist learning in students through activities that contemplate the interaction of students with certain content.

As suggested by Conole [28], the learning activities occur in a specific context, in terms of the environment (Environment) where it is developed, the pedagogical approaches adopted, and the institutional procedures and difficulties. For the development of these tasks, students have a series of resources (LearningObject and services). Conole proposes a taxonomy that defines the components that structure a learning activity [28]. One of the most useful aspects of taxonomy is the detailed description of the nature of the task that students will perform as part of the learning activity to achieve the desired objectives. Thus, the types of learning activities according to the nature of the task can be: assimilative, management of information, adaptive, communicative, productive, experiential, and evaluative. All the aforementioned occurs within a learning unit framework (LearningUnit) as a course, a module, a lesson, etc.

From the analysis of the different conceptual models, a business data pattern was generated. The data that compose the pattern were reviewed by experts in pedagogy and instructional design, in order to validate the concepts of the domain. In summary, Fig. 4 presents the data pattern resulting from the analysis of the didactic act applied to an LMS.

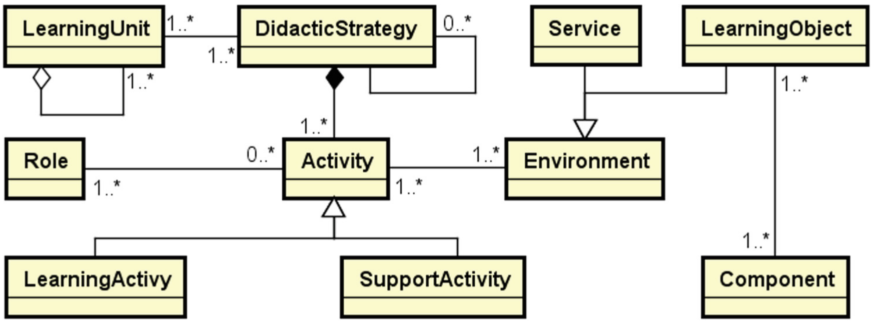


Fig. 4. Didactic act data pattern in an LMS

Know the Users, the Product, and the Tasks. To learn more about potential users, surveys were applied to 50 university students (34 men, 16 women) from different areas of knowledge. Their ages vary between 18 and 54, where 42% have a visual problem. 84% of those surveyed consider themselves to be fast learners and 94% like to work with computers and consider that this tool makes their work easier. 78% have a high experience in the management of computers, but only 12% feel satisfied with the LMS they handle. The most common problems reported with LMS are that their navigation is confusing, unintuitive, and the interface is unfriendly. Finally, 94% consider that the LMS has a high influence on the success of a course.

In order to know more about the LMS, 23 platforms were explored. The exploration was oriented to analyze the main functionalities and the technological components related to the didactic activities. The most common components found were the video player and the document visualizers as the most efficient way to transmit content to the students, the forums and the chat as the way the students discuss the topics learned and contribute to the courses, and Interactive quizzes and online tests as the most common components used to keep track on whether students met the learning objectives or not.

In relation to the tasks and use context, the contextual observation and field journal were used as mechanisms to analyze the didactic activities in the context of a face-to-face class in different university programs. In the contextual observation, the type of didactic activity was analyzed: who performs it, where, when, what resources are used, and the step by step process. In relation to the field journal, it was elaborated by five university students who recorded all their didactic activities inside and outside the university for two weeks. From this information, points of automation were identified in the process in a way that the didactic activity could be taken to a computational environment.

Capture and Generation of Requirement. Once the previous information was obtained, the user and usability requirements that the interfaces that would support the didactic activities must had were defined. The information resulting from this activity will not be described for it is common to most software engineering proposals.

Identify Patterns Variability and Patterns Elements. From the data pattern of the didactic act for an LMS (see Fig. 4), the criteria that make the associated presentation pattern vary in relation to the didactic activities were identified. They are: the type of didactic activity according to the nature of the task [28], synchronous or asynchronous, collaborative or individual, and interactive or passive activities. In accordance with these criteria, the elements that the presentation should have (AUI) associated to a determined variability were defined.

To exemplify the process, we will take the type of assimilative learning activity, according to the types of didactic activity identified in the "Know domain problem" section. An assimilative activity looks for in students to promote their understanding about certain concepts that the teacher presents orally, visually, or written [28]. In this fashion, if we move the environment of a classroom to a computer environment, we can have the video player to present the speech of the teacher, a document viewer to display presentations, or documents that support the speech; and the forum and chat as means of communication between teachers and students or between students. Now, the elements that are part of the interface will depend on the variability in the criteria. For example, if the teacher wishes to carry out an assimilative, asynchronous, individual, and passive activity (without intervention by the student), then he can use the video player and the document viewer. On the contrary, if he makes a variation from passive to interactive (with student intervention) he should use the forum or other asynchronous communication mechanism to receive the student's comments. Table 1 shows some examples of the different elements that the interface must have for an assimilative activity according to the variation in the identified criteria.

Once identified all the variability for each didactic act, the pattern variability tree is obtained as a result.

Conceptual, Interaction, and Detailed Design. From the variability tree, one has to start with the conceptual and interaction design of the presentation pattern for each identified variability. To achieve this, a workshop was held with teachers from different areas of knowledge (including educators) and students. Its purpose was to capture design ideas and obtain the first sketches from the perspective of users and experts. The workshop consisted of forming a scenario based on the variation of the didactic activity. Later, for each scenario, they generated ideas or made a sketch of how the student could interact with the proposed elements. With the results of the workshop, the user and usability requirements were fed back. Likewise, the first prototypes of the

Table 1. Interface elements for an assimilative activity according to the criteria variability

Criteria					
Asynchronous (A) or Synchronous (S)	A	A	A	S	S
Individual (I) or Collaborative (C)	I	I	C	I	C
Interactive (I) o Passive (P)	P	I	I	P	I
Interface elements					
Video player (pre-recorded)	X	X	X		
Video player (live)				X	X
Document viewer	X	X	X	X	X
Chat			X		X
Forum		X	X		

presentation patterns were created, which were evaluated by users in order to identify the way they would interact with each element of the interface in a use context.

Triad Definition. As a result of the previous activities, the triad of patterns is obtained; that is, for each variability of the data pattern, a presentation pattern (CUI) and an interaction pattern are obtained. Each pattern is modeled in the tool and all together contain the pattern catalog for the educational context. In summary, Fig. 6 presents the three patterns for a variability of the assimilative didactic activity. As it is

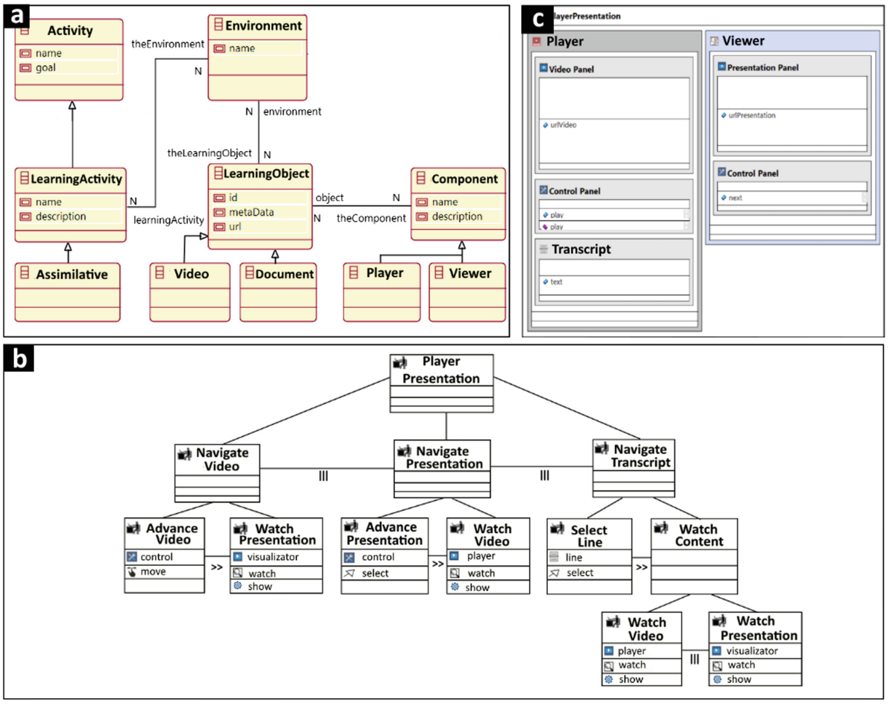


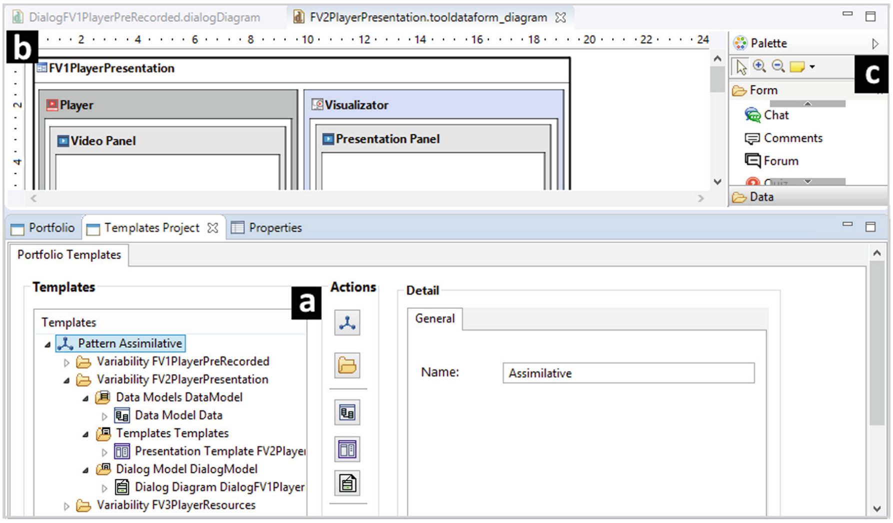
Fig. 5. Triad of patterns for a variability of an assimilative activity

observed, the data pattern (Fig. 5a) only has the data related to the presentation (Fig. 5c), and the interaction pattern (Fig. 5b) only has the actions that the user can perform with the elements of the interface.

The flow for the generation of interfaces from the created patterns is explained next.

5.2 Activity Flow for the Development of Interfaces from Patterns in the Educational Context

The flow for the development of interfaces begins with the analysis of the context, where the information related to the patterns and their different variabilities is analyzed with the purpose of knowing the contexts in which this pattern can be used. Once the patterns that can be used are identified, we select the pattern associated with a didactic activity from the variability tree of the pattern catalog (Fig. 6a). As it is shown, when displaying the selected pattern (Patterns assimilative), their different variabilities appear, which have a data pattern (Data Model), a presentation pattern (Template), and an interaction pattern (Dialog Model) associated. After selecting a certain variability, we proceed to configure the different properties associated with the elements of the interface (Fig. 6b) according to the specific context of use. For example, you can activate or deactivate functionalities (transcription, subtitles, quality, etc.), specify colors, fonts, among others for a video player. Once the interface elements (CUI) have been configured, a production with real test data is created and the programming logic that exposes functionalities in the final interface is implemented. Finally, the FUI is automatically generated. Currently, the tool generates interfaces in HTML5 and JavaScript.



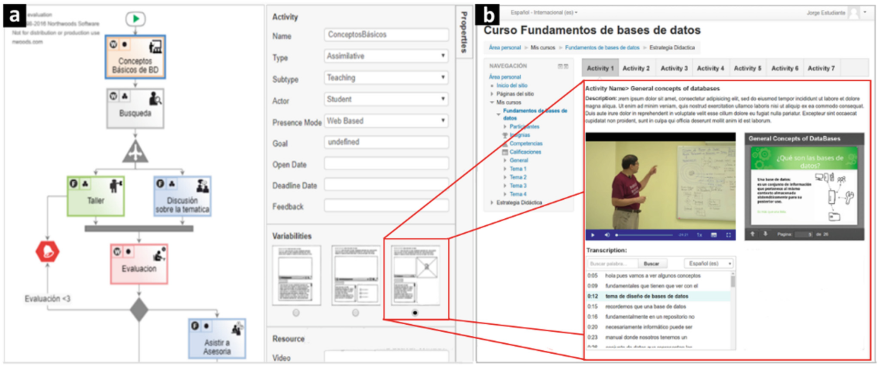
Templates
- Pattern Assimilative
  - Variability FV1PlayerPreRecorded
  - Variability FV2PlayerPresentation
    - Data Models DataModel
      - Data Model Data
    - Templates Templates
      - Presentation Template FV2Player
    - Dialog Model DialogModel
      - Dialog Diagram DialogFV1Player
  - Variability FV3PlayerResources

Name: Assimilative

Fig. 6. Development environment of the user interface from patterns

An alternate flow to this procedure is modifying the presentation pattern, adding new widgets or components of the palette (Fig. 6c). To do this, you must bear in mind that changes in the presentation may affect the other models and the usability of the final interface. The interfaces generated in the development environment are proven functional prototypes that can be used in any interactive system in the educational context.

Once the required interfaces were generated, they were integrated into the application as a means to the execution of the didactic activities of the model. In Fig. 7, you can see the model editor integrated in the Moodle platform (Fig. 7a) and the visualization for the student (Fig. 7b). Teachers use the didactic strategy model editor to plan the flow of activities that the student must complete. For each activity, the type of activity, the general data (delivery date, description, etc.), the interface desired for the student to observe, and the resources of the activity (videos, documents, etc.) are specify by teachers in the area of properties. In this way, the teacher focuses on his planning work, and does not have to deal with design or configuration problems of technological components. Likewise, students have an easy-to-use interface that supports them in the tasks they must perform.



Activity

Name	ConceptosBásicos
Type	Assimilative
Subtype	Teaching
Actor	Student
Presence Mode	Web Based
Goal	undefined
Open Date	
Deadline Date	
Feedback	

Variabilities

Resource

Video

Curso Fundamentos de bases de datos

Área personal > Mis cursos > Fundamentos de bases de datos > Estrategia Didactica

Activity Name> General concepts of databases

Description: ›rem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Una base de datos: es un conjunto de información que pertenece al mismo contexto almacenada sistemáticamente para su posterior uso.

Transcription:

```
0:05  hola pues vamos a ver algunos conceptos
0:09  fundamentales que tienen que ver con el
0:12  tema de diseño de bases de datos
0:15  recordemos que una base de datos
0:16  fundamentalmente en un repositorio no
0:20  necesariamente informático puede ser
0:23  manual donde nosotros tenemos un
```

Fig. 7. Didactic strategy model editor and student visualization in the Moodle platform

6 Conclusion and Future Work

This article presents a Roadmap for the design of user interfaces based on a triad of patterns that includes the principles and guidelines of MBUID, HCI, and DCU. The application of the Roadmap in the case study allowed to validate the method as a possible solution towards the automation of the usability in the MBUID tools. Similarly, it shows how it is possible to generate interfaces that although start from the information of three patterns (data, interaction, and presentation), they take into account information from other contexts, namely: from the user and his tasks, the use context, and the experts, as do other MBUID proposals using more models. Finally, this initiative aims for the software industry to apply this approach in such a way that the catalog of created patterns is gradually broadened. In this way, through the use of the catalog, developers and designers can speed up the development process of the user interface and promote usability in the final interface.

As future work, the validation of the Roadmap in other application contexts is proposed, as well as the promotion of this approach in the software industry through the incorporation of the triad of patterns as the main artifact. Additionally, at the tool level that supports the Roadmap, there is an extensive work that can be done such as: creating new code generators, expanding the palette of components and/or widgets, generating databases from data patterns, among others. Likewise, it would be useful to work in a web-based browser that allows the navigation and learning of the Roadmap and the reusing of the methods content for the definition of other process structures that rely on the triad of patterns.

References

1. Meixner, G., Paterno, F., Vanderdonckt, J.: Past present and future of model-based user interface development. iCom **10**, 2–11 (2011)
2. Wiecha, C., Bennett, W., Boies, S., Gould, J., Greene, S.: ITS: a tool for rapidly developing interactive applications. ACM Trans. Inf. Syst. **8**, 204–236 (1990)
3. Märtin, C.: Model-based software engineering for interactive systems. In: Albrecht, R. (ed.) Systems: Theory and Practice, pp. 187–211. Springer, Vienna (1998). https://doi.org/10.1007/978-3-7091-6451-8_9
4. Puerta, A.R.: The MECANO Project: Comprehensive and Integrated Support for Model-Based Interface Development (1996)
5. Puerta, A.R.: A model-based interface development environment. IEEE Softw. **14**, 40–47 (1997)
6. Berti, S., Mori, G., Paternò, F., Santoro, C.: A transformation-based environment for designing multi-device interactive applications. Presented at the Proceedings of the 9th International Conference on Intelligent User Interfaces, Funchal, Madeira, Portugal (2004)
7. Gajos, K.Z., Weld, D.S., Wobbrock, J.O.: Automatically generating personalized user interfaces with SUPPLE. Artif. Intell. **174**, 910–950 (2010)
8. Paterno', F., Santoro, C., Spano, L.D.: MARIA: a universal, declarative, multiple abstraction-level language for service-oriented applications in ubiquitous environments. ACM Trans. Comput.-Hum. Interact. **16**, 1–30 (2009)
9. Giraldo, W.J.: Marco de Desarrollo de Sistemas Groupware Interactivos Basado en la Integración de Procesos y Notaciones – CIAF. Ph.D. Doctoral, Universidad de Castilla - La Mancha (2010)
10. Akiki, P.A., Bandara, A.K., Yu, Y.: Cedar studio: an IDE supporting adaptive model-driven user interfaces for enterprise applications. Presented at the Proceedings of the 5th ACM SIGCHI Symposium on Engineering Interactive Computing Systems, London, United Kingdom (2013)
11. Peissner, M., Häbe, D., Janssen, D., Sellner, T.: MyUI: generating accessible user interfaces from multimodal design patterns. Presented at the Proceedings of the 4th ACM SIGCHI Symposium on Engineering Interactive Computing Systems, Copenhagen, Denmark (2012)
12. Engel, J., Märtin, C.: PaMGIS: a framework for pattern-based modeling and generation of interactive systems. In: Jacko, J.A. (ed.) HCI 2009. LNCS, vol. 5610, pp. 826–835. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-02574-7_92

13. Marin, I., Ortin, F., Pedrosa, G., Rodriguez, J.: Generating native user interfaces for multiple devices by means of model transformation. Front. Inf. Technol. Electron. Eng. **16**, 995–1017 (2015)
14. Machado, M., Couto, R., Campos, J.C.: MODUS: model-based user interfaces prototyping. Presented at the Proceedings of the ACM SIGCHI Symposium on Engineering Interactive Computing Systems, Lisbon, Portugal (2017)
15. Calvary, G., Coutaz, J., Thevenin, D., Limbourg, Q., Bouillon, L., Vanderdonckt, J.: A unifying reference framework for multitarget user interfaces, interacting with computers. Interact. Comput. **15**, 289–308 (2003)
16. Courage, C., Baxter, K.: Understanding Your Users: A Practical Guide to User Requirements Methods, Tools, and Techniques. Morgan Kaufmann Publishers, Burlington (2005)
17. Shneiderman, B., Plaisant, C.: Designing the User Interface: Strategies for Effective Human-Computer Interaction. Pearson Addison Wesley, Boston (2004)
18. Schmidt, D.C., Stal, M., Rohnert, H., Buschmann, F.: Pattern-Oriented Software Architecture, Patterns for Concurrent and Networked Objects, vol. 2. Wiley, Hoboken (2013)
19. Triviño, J.I.: Generación de la interfaz de usuario de negocio a partir de patrones de negocios basada en los fundamentos metodológicos de TD_MBUID. Magister, Informatica Universidad EAFIT (2015)
20. Giraldo, O.W.J., Arias, M.R., Collazos, O.C.A., Molina, A.I., Ortega, C.M., Redondo, M.A.: Fast functional prototyping of user interfaces based on DataForm models, a tool (ToolDataForm). In: 2015 10th Computing Colombian Conference (10CCC), pp. 172–179 (2015)
21. Paternó, F., Mancini, C., Meniconi, S.: ConcurTaskTrees: a diagrammatic notation for specifying task models. Presented at the Proceedings of the IFIP TC13 International Conference on Human-Computer Interaction (1997)
22. Limbourg, Q.: Multi-path development of user interfaces. Ph.D. Université catholique de Louvain, Louvain-la-Neuve, Belgium (2004)
23. Ruiz, A., Giraldo, W.J., Arciniegas, J.: Method for the integration of the didactic strategy model in virtual learning platforms in the university context: a mechanism that takes into account the professor's work. Presented at the 12 Congreso Colombiano de Computación - 12CCC, Cali, Colombia (2017)
24. Marqués, P.: La Enseñanza Buenas Prácticas. La Motivación (2011). http://peremarques.net/
25. Meneses, G.: NTIC, Interacción y Aprendizaje en la Universidad. Doctorado, Departament de Pedagogia, Universitat Rovira I Virgili (2007)
26. Rodríguez, J.L.: Curriculum, acto didáctico y teoría del texto. Anaya, ed España (1985)
27. Koper, R., Olivier, B., Anderson, T.: IMS learning design information model. IMS Global Learning Consortium (2003)
28. Conole, G.: Describing learning activities: tools and resources to guide practice. In: Beetham, H., Sharpe, R. (eds.) Rethinking Pedagogy for a Digital Age: Designing and Delivering E-learning. Routledge, Abingdon (2007)