CLEI Electronic Journal, Volume 21, Number 2, Paper 4, August 2018

A Model-Driven Approach to Develop Rich Web Applications

Guido Nuñez, Daniel Bonhaure, Magalí González, Nathalie Aquino, and Luca Cernuzzi

Departamento de Electrónica e Informática - DEI
Universidad Católica "Nuestra Señora de la Asunción"
Asunción, Paraguay

Abstract. Many Web applications offer the possibility of distributing their data and their business logic between the client and the server, also allowing an asynchronous communication between them. These features, originally associated with the arrival of Rich Internet Applications (RIA), remain particularly relevant and desirable. In the area of RIA, there are few proposals that simultaneously consider these features, adopt Model-Driven Development (MDD), and use implementation technologies based on scripting. In this work, we start from MoWebA, an MDD approach to web application development, and we extend it by defining a specific architecture model with RIA functionalities, supporting the previously mentioned features. We have defined the necessary metamodels and UML profiles, as well as transformation rules that allow code based on HTML5, Javascript, jQuery, jQuery Datatables and jQuery UI to be generated. The preliminary validation of the proposal shows positive evidences regarding the effectiveness, efficiency and satisfaction of the users with respect to the modeling and code generation processes of the proposal.

Keywords: Model Driven Development, MDD, Model Driven Architecture, MDA, Rich Internet Applications, RIA, Model Oriented Web Approach, MoWebA.

1 Introduction

In the past, Web applications were characterized by a server side processing, where the client was only able to request and display content. This behavior was the cause of limited interactions with the user, the need to reload the entire page to perform navigation, long response times and a difficult way to display animations and multimedia content.

Rich Internet Applications (RIA) came to overcome these limitations [1]. This type of applications has made it possible to significantly improve the user experience, mainly in terms of presentation and interaction. RIAs have four main features: client data storage, client business logic processing, asynchronous communication between client and server, and the improvement of the user interface [1,2]. Currently, regardless of whether they follow a specific RIA architecture or not, a growing percentage of web applications have adopted among their features the possibility of distributing their data and business logic among the client and the server, also allowing an asynchronous communication between them. Many technologies have been proposed to ease the development of RIA, and they have managed to increase developers' productivity. However, certain disadvantages remain, such as the lack of high-level instruments that abstract details of implementation, the propensity to errors, the lack of consideration of the complete life cycle of the software, among others [3].

Model-Driven Development (MDD) [4] approaches offer solutions to some of those limitations, allowing applications to be created through the specification of models and the generation of code from them. If these approaches adopt a standard such as Model-Driven Architecture (MDA) [5], the task of conducting the development process will be facilitated and will guarantee greater interoperability and portability among systems. Several MDD proposals for the implementation of RIA are found in the literature [6,7,8,9]. However, many of such proposals focus on the presentation layer and scarcely consider the features of data distribution, distribution of business logic and asynchronous communication between client and server. Among the approaches that do consider these last features, it can be noted that they have not adopted standards or have not developed an implementation with scripting-based technologies, which are the most common option for RIA development practices [3].

Furthermore, one of the strategies that MDA promotes to facilitate changeability and portability of the conceptual model is the prescription of the Platform Independent Model (PIM) separated from the Platform Specific Model (PSM). However, in practice, current web methodologies tend to cope with architectural evolution trends by extending their modeling notations directly at the PIM level (e.g., WebML RIA [10], UWE for RIA [11], OOH4Ria [12], among others). Such extension usually results in an enriched PIM that includes characteristics and constraints of a certain specific architecture or platform. The enriched PIMs are obtained either by adding marks (but without considering this as a marked PIM), or by adding new entire models, or even by adding new elements in the PIM notation. Although it could be argued that after extending a PIM to support a specific architecture it is no longer a PIM, but a PSM, in general, the literature does not analyze this distinction.

The problem is that this tendency of adapting the PIM to specific architectures goes against the principle of the PIM portability promoted by MDA. The enriched PIM loses part of its independence from the architecture/platform, and becomes increasingly more complex to understand and manage. The consequence is a loss of portability and reusability of the conceptual models.

Another issue related to portability arises when web methodologies tend to enrich the PSM with architectural details, since the same architecture can be implemented in different platforms (e.g., the RIA architecture can be implemented in Backbase, Dojo, GWT, jQuery, among others). In this case, the problem does not concern the conceptual model, but one same architecture requires multiple PSMs, according to the different implementation platforms.

To preserve the independence of the PIM and the portability of models towards different architectures, some authors have already proposed the introduction of an Architecture Specific Model (ASM) (e.g., [13,14]). The ASM allows details related to the architecture to be moved out from the problem space (PIM) to an intermediate model in the solution space (ASM), avoiding the PIM from containing such details and contributing to its so desired portability. Moreover, the ASM model can result in different PSM models, one per each implementation platform.

In this line, MoWebA [13] constitutes an MDD approach to the development of web applications that adopts the MDA standard. Furthermore, MoWebA can be extended through the definition of ASM, used to complement the PIM with information about a specific architecture. In previous work, MoWebA has already been extended to support RIA [15], but this extension has been limited to presentation aspects, focusing exclusively on the user interface, so we considered convenient to make a complementary extension to support other desirable features of rich web applications.

Therefore, this work defines an extension of MoWebA focused on the development of RIA, and more generally, web applications enriched with features of client data, client business logic and asynchronous communication between client and server.

A previous version of this work was presented in CLEI 2017 [16]. This paper enriches and complements the previous work in three main aspects. First, considering the issue of portability of the conceptual model, we extended the related work section to discuss the state of the art on such dimension (see section 2). Second, the current work introduces model-to-model (M2M) transformations from the PIM to an ASM for RIAs (see sections 3 and 4). The PIM-ASM transformations are defined using ATL (Atlas Transformation Language) and are executed automatically, instead of the manual transformation process adopted in the previous proposal. Indeed, such improvement let to save time, effort, and errors during the development process. Third, we have carried out a new experience to improve the validation of the proposal. The new experience was conducted with 10 students and covers the full development process (i.e., PIM-ASM-Code), instead of the previous validation experience that just covered the ASM-Code phase (see section 5).

The rest of this paper is organized as follows: section 2 presents the state of the art of MDD approaches for the development of RIAs; section 3 provides a basic overview on MoWebA, with its processes, putting specific emphasis on the ASM model; section 4 describes the extension of MoWebA, presenting the metamodels, profiles and transformation rules corresponding to this work; section 5 describes the validation experience carried out, and section 6 finalizes the document specifying the conclusion of the work.

2 Related work

We reviewed the literature related to MDD approaches for the development of RIA that consider the features of client data, client business logic and asynchronous communication between client and server, which are the features of interest in this work. The main results are shown in Table 1. This table shows the identified approaches with their references, ordered by year, from least to greatest. It also presents information on the supported RIA features, the adoption of MDA as a standard to conduct the development process, the implementation technology and the type of implementation technology.

We note that the studies included between 2006 and 2010 came to consider the features of client data, client business logic and asynchronous communication between client and server, while the more contemporary studies, going from 2013 to 2016, have hardly considered such features, focusing mostly on the rich user interface feature. While we are aware of the importance of the presentation layer to ensure a positive experience to the user, from the functional and performance point of view, the other features continue to play a very important role and is notorious how they have been scarcely considered in the last years. For this reason, our current proposal focuses on recovering the importance of modeling and supporting the development of these features.

Among the works that cover all our features of interest simultaneously, only Meliá et al.[23] adopts MDA to guide its process. The adoption of MDA is convenient to standardize and improve the software development process, by establishing better interoperability and portability among systems. In addition, it is the only study that generates an implementation using scripting-based technologies. These technologies are mostly used in the developers's community for the implementation of RIA [3], since they favor the development of the client data and client business logic features, as well as the asynchronous communication between client and server. Other scripting-based technologies such as HTML5, jQuery and GWT, are not taken into account for the implementation of the features of interest.

We also reviewed the literature related to architectural aspects at the conceptual modeling level [30]. We found that most approaches consider architectural aspects without making a clear distinction between the independent model and the architectural one. For example, in order to generate RIA, some approaches have extended their notations with additional architecture-specific primitives or patterns (e.g., WebML RIA [10], UWE for RIA [11], RUX-model [18], [29], among others). Other proposals decided to add architectural specific information at the PSM level. In this case, architecture and platform details are included at the same modeling level, loosing portability at the architectural level with respect to different technologies where it can be implemented (e.g., UWA for RIA [27]).

As early as 2004, Mikkonen et al. [14] realized that architectural styles do not have a clear place in MDA. They proposed to modify MDA by adding a new layer called Architecture Specific Model (ASM). This new layer encapsulates architectural properties in it. At the same time, it makes architectural styles,

Table 1. MDD approaches for the development of RIA

	Year	**Client Data**	**Client Business Logic**	**Asynchronous Communication**	**MDA**	**Implementation Technology**	**Type of Implementation Technology**
WebML for RIA [17]	2006	✓	✓	✗	No	OpenLaszlo	Plugin-based
RUX-Model [18]	2007	✗	✗	✓	Yes	Adobe Flex, OpenLaszlo	Plugin-based
WebML for RIA + RUX-Model [19]	2007	✓	✓	✓	No	OpenLaszlo, Adobe Flex, XAML	Plugin-based
WebML for RIA for collaborative web applications [20]	2007	✓	✓	✓	No	OpenLaszlo	Plugin-based
Conceptual design of RIA based on business processes [21]	2008	✓	✓	✓	No	Not specified	-
OOWS Extension [22]	2009	✗	✗	✓	No	Adobe Flex	Plugin-based
OOH4RIA Extension [23]	2010	✓	✓	✓	Yes	RichFaces	Scripting-based
MDD approach for HTML5-based interactive GUI components [24]	2013	✗	✗	✗	No	HTML5, Javascript	Scripting-based
MDD approach for high quality web applications [25]	2013	✗	✗	✓	Yes	Javascript	Scripting-based
MDD approach based on UML for MVP web applications [26]	2014	✗	✗	✗	Yes	GWT	Scripting-based
MDD approach for RIA prototyping [27]	2014	✗	✗	✓	No	RichFaces	Scripting-based
MDUID process integrating HCI patterns [6]	2015	✗	✗	✗	No	JSP, Javascript, CSS	Scripting-based
MDA approach for MVP, DI and DAO patterns for RIA [7]	2015	✗	✗	✗	Yes	GWT	Scripting-based
Approach for the MDD generation of RIA GUI using IFML [28]	2015	✗	✗	✗	Yes	JavaFX	Specific Runtime Environment
MDA approach for web applications [29]	2015	✓	✗	✓	Yes	WCF	Scripting-based
MDD process for MVC RIA focused on GUI [8]	2016	✗	✗	✗	Yes	JavaFX	Specific Runtime Environment
MDD approach for RIA GUI using IFML and Ontologies [9]	2016	✗	✗	✗	Yes	Adoble Flex	Plugin-based

design patterns, and other important design decisions explicit in the model. This interesting proposal was discontinued but inspired the MoWebA proposal, in which the same PIM could be used as a base to generate different ASMs for different architectures (e.g., RIA, REST, client-server, SOA, mobile).

Afterwards, Manset et al. [31] defined an architecture-centric model-driven approach for the automatic generation of grid applications. They merged model-driven development with architecture-centric [32] processes, which considers architecture as the main artifact in the software development process. Also in this direction, Marcos et al. [33] extended MIDAS, a methodological framework for the development of web information systems, by integrating architectural design aspects. MIDAS considers three different viewpoints of web information systems, namely content, hypertext and behavior, which are orthogonal to MDA abstraction levels (Computational Independent Model or CIM, PIM and PSM). Furthermore, the software architecture is conceived as a crosscutting perspective, which is in turn orthogonal to the mentioned three viewpoints. Therefore, both Platform-Independent Architecture and Platform-Specific Architecture models are defined. More recently, these efforts evolved into ArchiMeDes [34], [35], a model-driven framework for the specification of service-oriented architectures. At the PIM level, ArchiMeDes defines a domain-specific language that allows the definition of conceptual service architectures. At the PSM level, different domain-specific languages support the modeling of concrete execution platforms or implementation technologies. ArchMDE [36] is another architecture-centric model-driven engineering approach, and it focuses on the development and validation of embedded real time systems. In this case, the PIM is decomposed into two models: i) an Architectural-Style Independent Model, which defines the structure and behavior of the functional architecture of real time system; and ii) an Architectural-Style Specific Model, which takes into account the functional, non-functional and architectural characteristics.

All of these works constitute architecture-centric efforts, this is to say, they consider architecture as the main artifact in the software development process and, therefore, architectural concepts appear early in the development process (for instance, at the PIM level). In the case of MoWebA, while there is an interest in being able to develop one same application for different architectures, the development process is not based on the architecture. It can also be highlighted that some of these works are specific for the service-oriented architecture ([34], [35]), while in MoWebA, different ASMs for different architectures can be defined. Finally, some of these works are domain specific (grid applications [31], real time systems [36]), as opposed to MoWebA, which allows general management information systems to be developed.

Considering the related work previously mentioned, we perceive the need to provide a solution to the development of RIA, and more generally of enriched Web applications, through an MDD approach that considers the features of client data, client business logic, and asynchronous communication between client and server, adopts standards (such as MDA), preserves conceptual modeling portability and generates an implementation with scripting-based technologies.

In addition, the adoption of MoWebA allows us to address some problems detected in previous work [30]. Specifically, it faces a tendency of the scientific community to set aside some key aspects of MDD, which are: i) M2M transformations, ii) the portability of the PIM, iii) the development of tools that implement M2M transformations, among others.

3 MoWebA in a nutshell

MoWebA is a navigational, role-centric, model-based approach for the development of web and mobile applications [13,37]. It defines methodological aspects (i.e., processes, stages, work products, dimensions) and complements them with an entire development environment, including modeling and transformation tools, automatic code generation, use of standards, and layered architecture, among others.

Figure 1 presents the dimensions of MoWebA that cover its modeling and transformation processes. MoWebA adopts the MDA approach by identifying three different phases related to modeling and transformation activities (the phases dimension): i) the problem space, covered by CIM (Computational Independent Model) and PIM (Platform Independent Model); ii) the solution space, covered by ASM (Architecture Specific Model) and PSM (Platform Specific Model); iii) and the source code definition, covered by the ISM (Implementation Specific Model) and manual code. The levels dimension deals with complementary perspectives to be considered in every phase (content, business logic, navigation, presentation, users). Finally, the aspects dimension addresses structural and behavioral considerations for each perspective.

3.1 Developing applications with MoWebA

In order to develop applications, MoWebA defines two main complementary processes: the first, related to modeling activities, and the second, related to transformation activities. To formalize modeling and transformation processes, it adopts the MOF language for the definition of the abstract syntax, and the UML profile extension for a precise definition of the modeling language (i.e., concrete syntax).

The modeling process encompasses seven stages with their corresponding activities to specify the system-to-be (considering the problem space, architecture/s, and target platform/s). These processes considers the CIM, the PIM, the ASM and the PSM. The definition of the CIM covers late requirements identification, focusing on functional requirements specifications. The PIM specification is based on five models, offering a strong separation of concerns: Domain, Logic, Navigation, Presentation, and User. The ASM enriches the models with information for a specific architecture (e.g., RIA, SOA, REST, among others), and the PSM adds information related to the target platform (e.g., specific programming languages, frameworks, among others).

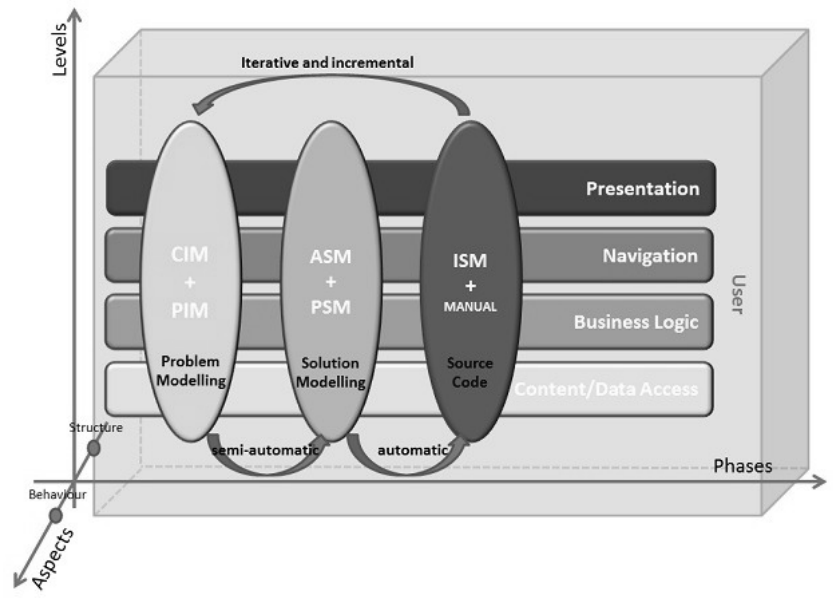


Fig. 1. MoWebA dimensions

The transformation process, on the other hand, is related to the steps, techniques, and tools that allow M2M (i.e., model-to-model) and/or M2T (i.e., model-to-text/code) transformations. This process go through each MoWebA phase (i.e., from CIM/PIM to ASM/PSM, and from ASM/PSM to ISM/manual adjustments). The transformation from CIM/PIM to ASM/PSM should be performed in an automatic way (i.e., having as input the PIM model and a configuration file), precisely what we are looking for in this work. To achieve this goal, MoWebA prescribes a process to define metamodels for specific architectures or platforms, and the corresponding mapping rules for PIM to ASM/PSM transformations. The transformation from ASM/PSM to ISM is performed automatically from models to the application code. Since real experiences have shown that manual changes are necessary sometimes, we consider a phase of "manual adjustments", where additional code can be added to tune the application. Finally, it is worth to mention that the transformation process can be performed iteratively, allowing an incremental application development.

3.2 Defining and applying ASM models

The ASM model enriches previous models with additional information related to the system architecture (e.g., RIA, REST, among others). Complementary, the PSM is oriented to refine models adding information related to the platform and language that were selected for the final system (e.g., Java, .NET, PostgreSQL). At this stage, software engineers are moving from the conceptual definition (CIM/PIM models) to the solution definition (ASM/PSM models).

It is necessary to define the ASM model before using it. This definition encompasses the specification of the corresponding metamodel, among other steps. Brambilla et al. have recommended a process for defining an abstract syntax [38] and MoWebA follows it to define an ASM metamodel. Furthermore, MoWebA complements the suggested process with additional steps that go from the definition of the concrete syntax till the generation of the final code of an application [39]. The steps of this process are synthesized below:

1. Define the ASM metamodel using MOF.
2. Define the corresponding UML Profile.
3. Specify the mapping rules from PIM elements to ASM elements.
4. Define transformation rules from PIM to ASM using standard transformation languages (e.g., ATL or QVT).
5. Define transformation rules from ASM to PSM and PSM to code, or from ASM to code, using M2M and M2T (e.g., Acceleo) transformation languages, respectively.

As it can be seen in the steps of the previous process, the MoWebA approach requires some additional effort as a counterpart of improving the portability of the PIM and facilitating the architectural evolution of applications. This includes the need for the specification of ASM metamodels and the definition of the corresponding transformation rules, in order to achieve automatic transformations on the proposed architecture. In any case, it should be noticed that all previous steps will only be performed once, when targeting to a new architecture for the first time.

Once the ASM for a specific architecture is defined, it can be used to develop an application for the selected architecture following these steps:

1. Define the MoWebA CIM/PIM diagrams following the modeling process (see 3.1).
2. Apply transformation rules in order to obtain the first version of the ASM model.
3. Make manual adjustments (if necessary) to complete the ASM model.
4. Generate the PSM models and/or the final code, applying transformation rules.
5. Include manual adjustments, if necessary.

4 An extension of MoWebA for RIA

In this work, we use MoWebA [13] as a starting point and propose an extension for RIA's own functionalities at the architectural modeling level. We have chosen MoWebA because it has certain advantages compared to other proposals, among them the adoption of standards, the provision of an ASM to extend the approach with functionalities related to a specific architecture, and the conception of a modeling approach based on function-oriented navigation. This function-oriented navigation provides greater cohesion and less coupling with respect to data-oriented proposals and enhances a top-down development, which for architectural purposes provides a clearer and more understandable structure, better reflecting the needs of the user. We present first the ASM definition, following the M2M transformation rules from PIM to ASM. Next we present the M2T transformation rules going through the entire process of code generation.

4.1 ASM RIA for MoWebA

In order to facilitate the modeling of RIA functionalities, we implemented extensions to the metamodel and original profiles of MoWebA. These extensions correspond to functionalities that were not included in the initial version of MoWebA and that allow to facilitate the subsequent modeling of RIA features. We apply the extensions to the logic and content diagrams. More details about the metamodel and original MoWebA profiles can be found in [13].

From the extensions made, we developed the metamodel shown in Figure 2, which is used for the definition of an ASM for RIA. In this metamodel we present the different functionalities incorporated to model the features of client data, client business logic and asynchronous communication between client and server.

To save data in a web client, specifically in a web browser, we created two new classes, a *ClientValueObject*, which extends the value object of the original MoWebA logic diagram, and a *ClientStaticObject*, which extends a static object, a new element introduced in the logic diagram to represent sets of statically defined values as properties of the class. These new objects differ from the original ones at the implementation level, since both the *ClientValueObject* and the *ClientStaticObject*, are mapped to variables stored in the browser. The name of the class corresponds to the name of the variable, and the values of the class with their associated tagged values (*title* and *checked*) correspond to the values of the variable. In addition, these new classes allow a level of *persistence* to be specified, which can be *permanent* or *temporary*, allowing the variable to persist or not at the end of a session or browser close. If the *persistence* level is *permanent*, the HTML *Localstorage*[1] object will be generated, while if the *persistence* is *temporary*, the HTML *Sessionstorage*[2] object will be generated, both accessed through Javascript.

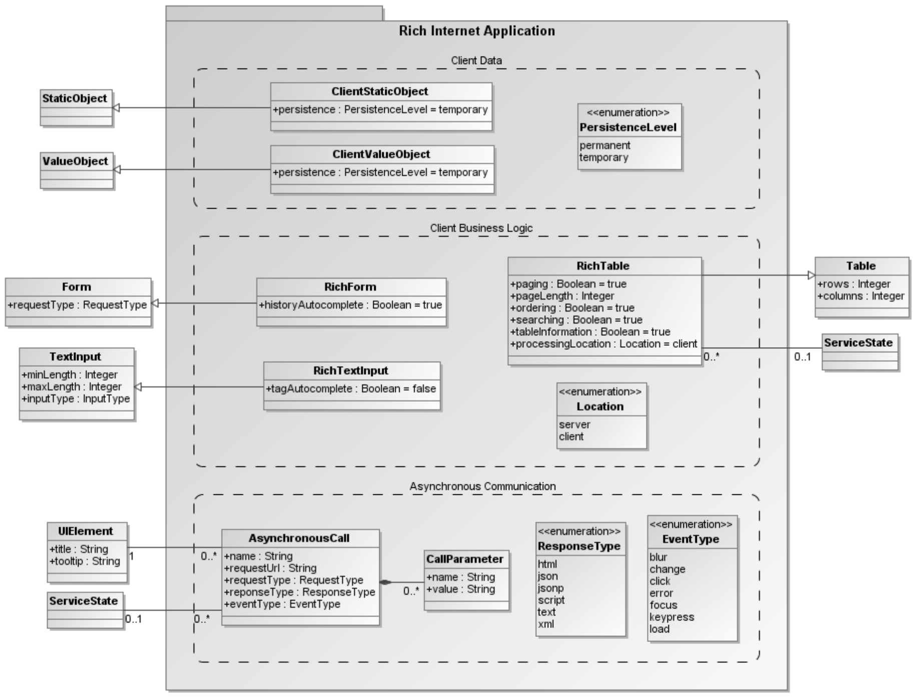
In terms of processes executed on the client, we created three elements that run in the web browser, the *RichForm*, the *RichTextInput* and the *RichTable*.

The *RichForm* is a specialization of the *Form* element of the original MoWebA content diagram. It contains an autocomplete attribute based on history called *historyAutocomplete*, which allows the browser to suggest values to complete the

[1]LocalStorage.`https://www.w3.org/TR/webstorage/#the-localstorage-attribute`

[2]SessionStorage.`https://www.w3.org/TR/webstorage/#the-sessionstorage-attribute`

Fig. 2. Metamodel used for the definition of an ASM for RIA



form's entries, based on previously entered values. From each instance of a *Rich-Form*, an HTML form will be generated with the autocomplete property enabled or not, as established in the model.

The *RichTextInput* extends the *TextInput* of the original MoWebA content diagram. This element introduces the attribute *tagAutocomplete*, which differs from the previous attribute *historyAutocomplete* in that the values to be suggested are obtained from an association with objects of value or static objects and their specializations. Later, using jQuery UI[3], the association between inputs and values will be generated and the autocomplete option will be displayed.

The *RichTable* specializes the class *Table* of the original content diagram of MoWebA. This element allows to specify the service of the node diagram that will be responsible for populating the table. In addition, it adds new properties to the table to allow paging (*paging*), specify number of records per page (*pageLength*), allow ordering by columns (*ordering*), look up words in records (*searching*), display information (index of current page, number of pages, number of records) of the table (*tableInformation*), and specify where these functions

[3]JQuery UI.`https://jqueryui.com/`

will be processed (*processingLocation*), either on the *server* or the *client*. The jQuery plugin Datatables[4] will be used for the implementation of a *RichTable*.

To allow asynchronous communications between client and server we created the class called *AsynchronousCall*. This class is used to make an AJAX request for a service. The *AsynchronousCall* is associated with one or many UI elements in the content diagram and zero or one service of the node diagram. When an event occurs on some UI element, a service is executed. The service can be one specified in the node diagram, or a service running from a URL outside the system (for example, a web service). The instance of the *AsynchronousCall* can be located in the content diagram along with its associated UI elements. The class has a name (*name*); a URL (*requestUrl*), used in case an external service is requested; a type of request (*requestType*), which can be used to retrieve remote data (*retrieve*) or insert or update data (*insert/update*); a type of response (*responseType*), which can be *html*, *json*, *jsonp*, *script*, *text* or *xml* formats; and a type of event (*eventType*), that is applied to the UI element and can take the values *blur*, *change*, *click*, *error*, *focus*, *keypress* or *load*. In addition, the *AsynchronousCall* may attach certain parameters to the request. Each parameter consists of a property of the class that has a name and a value. Every *AsynchronousCall* will generate an *ajax()* method of jQuery[5].

Figure 3 presents the UML profile that corresponds to the described ASM RIA metamodel.

4.2 Transformation rules

As previously mentioned, in this work we introduce M2M transformation rules to automatically produce the ASM from the modeled PIM. Such improvement lets designers and software engineers to reduce effort and time to develop the system-to-be.

For the definition of the M2M transformation rules, we have chosen ATL[6] (Atlas Transformation Language) over QVT[7] (Query/View/Transformation). This choice was based, above all, on the fact that ATL is considered one of the most widely used transformation languages, both in academia and industry, and a mature tool support is available [38].

Another very important choice was the selection of the engine execution mode of the ATL transformation, which has two modes of execution: default and refining [40]. When the source and target metamodels are different, it is mandatory to use the default execution mode, but when the metamodels of the source and target models are the same you can opt for either of the two execution modes [40].

In our case, both, the metamodels of source and target models are the same because MoWebA's implementation is based on profiles [13]. Moreover, the M2M

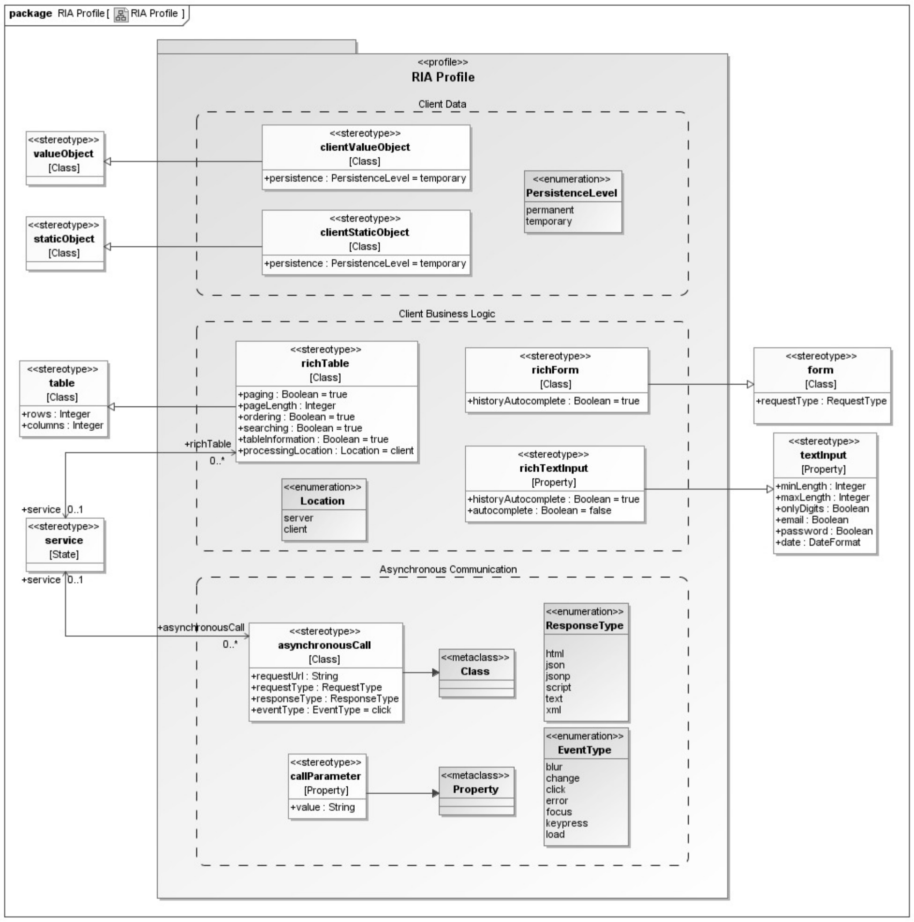
[4]Datatables.`https://datatables.net/`

[5]JQuery.`https://jquery.com/`

[6]ATL.`http://www.eclipse.org/atl/`

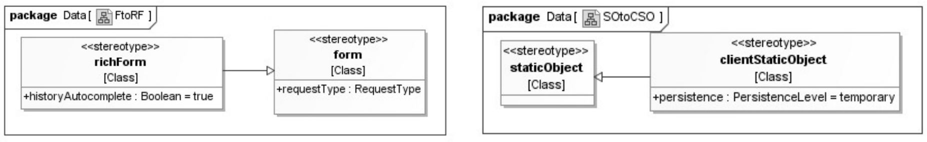
[7]QVT.`http://www.omg.org/spec/QVT/About-QVT/`

Fig. 3. ASM RIA Profile for MoWebA


package RIA Profile [RIA Profile]
<<profile>>
RIA Profile
Client Data
<<stereotype>> valueObject [Class]
<<stereotype>> clientValueObject [Class]
+persistence : PersistenceLevel = temporary
<<enumeration>> PersistenceLevel
permanent
temporary
<<stereotype>> staticObject [Class]
<<stereotype>> clientStaticObject [Class]
+persistence : PersistenceLevel = temporary
Client Business Logic
<<stereotype>> table [Class]
+rows : Integer
+columns : Integer
<<stereotype>> richTable [Class]
+paging : Boolean = true
+pageLength : Integer
+ordering : Boolean = true
+searching : Boolean = true
+tableInformation : Boolean = true
+processingLocation : Location = client
<<stereotype>> richForm [Class]
+historyAutocomplete : Boolean = true
<<stereotype>> form [Class]
+requestType : RequestType
<<stereotype>> richTextInput [Property]
+historyAutocomplete : Boolean = true
+autocomplete : Boolean = false
<<stereotype>> textInput [Property]
+minLength : Integer
+maxLength : Integer
+onlyDigits : Boolean
+email : Boolean
+password : Boolean
+date : DateFormat
<<enumeration>> Location
server
client
+richTable 0..*
+service 0..1
<<stereotype>> service [State]
+service 0..1
Asynchronous Communication
+asynchronousCall 0..*
<<stereotype>> asynchronousCall [Class]
+requestUrl : String
+requestType : RequestType
+responseType : ResponseType
+eventType : EventType = click
<<metaclass>> Class
<<enumeration>> ResponseType
html
json
jsonp
script
text
xml
<<enumeration>> EventType
blur
change
click
error
focus
keypress
load
<<stereotype>> callParameter [Property]
+value : String
<<metaclass>> Property

transformation from PIM to ASM fits conceptually better to the refining mode, since the PIM is justly refined to obtain the ASM. For all this, we opted for the refining execution mode. This choice has greatly reduced the number of necessary transformation rules. However, this also led to some complications, since the application of profiles in ATL is performed in the imperative block and turns out this option when using refining mode.

For the above reason, we were forced to change the compiler. The default compiler in Eclipse is the EMF-specific Virtual Machine (EMF-specific VM), but it also provides other compilers. We opted for the EMF Transformation Virtual Machine (EMFTVM). Although the main reason for this choice was that it allows the use of the imperative block in the refining mode, which is necessary to work with profiles, there are some other advantages. For example,



(a) (b)

Fig. 4. Mapping-RIA-Inheritance

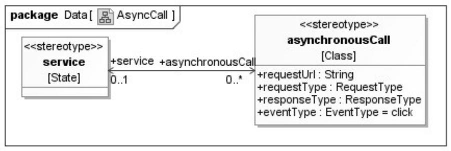


Fig. 5. Mapping-RIA-Asynchronous service

its performance is roughly 80% better than the EMF-specific VM [41], and allows us to invoke native Java methods [42].

The PIM-ASM mapping phase: Before the definition of the transformation rules, a mapping between the elements of the PIM and the elements of the ASM has been made, seeking to identify which elements of the PIM model must be transformed and, above all, to which elements of the ASM model.

This mapping was performed through a visual analysis of the defined profile for RIA (see Figure 3).

During the mapping phase we noticed that, in general, when the relation between classes from source and target models is an inheritance, the transformation that allows to obtain the corresponding target element is very simple and consists of the application of the correct stereotype. Thus, for example, in the RIA profile, a class of type Table (or a class with the stereotype Table applied) in the PIM model, becomes a class of type RichTable in the ASM model (by applying the RichTable stereotype). A class of type Form, becomes a RichForm class (see Figure 4a). A class of type TextInput is transformed into a RichTextInput class. A StaticObject class, becomes a ClientStaticObject class (see Figure 4b). A ValueObject class is transformed into a RichValueObject class.

When the relation between two classes in one profile is not an inheritance, the mapping becomes a bit more difficult. In the case of the RIA profile, we found two relations of this type, one between the ServiceState and AsynchronousCall classes, and the other between the ServiceState and RichTable classes.

Analyzing these relationships, we could conclude that the AsynchronousCall class must be created when it is related to a ServiceState class, which, in addition, must represent an asynchronous service (see Figure 5). This condition can be detected automatically, allowing the automation of the transformation.

```
mod: 1
classes:
    - package      : package_containing_the_class
      class        : class_name
properties:
    - package      : package_containing_the_property
      class        : class_containing_the_property
      property     : property_name
```

Fig. 6. ArchConfTransformacion.yaml - Example

Configuration Files: Configuration files allow the designer to control some transformation aspects, allowing the achievement of two important capabilities: on the one hand, the possibility of capturing specific design decisions of the system under design that would otherwise not be automated, and on the other hand, the possibility that the designer has some level of influence over the transformation rules.

Two different configuration files have been considered. One that allows to indicate which elements of the model are transformed and which are not, called "ArchConfTransformacion.yaml", and another one that allows to specify some properties for the classes created automatically when executing the M2M transformation rules, called "ArchConfAsynchronousCall.yaml".

The "ArchConfTransformacion.yaml" configuration file has two modes of operation. Mode 1 specifies which elements of the PIM must be transformed and mode 2 specifies which elements must not. If the file does not exist, all elements of the PIM that are referenced by the M2M transformation rules are transformed. Its structure is very simple. It has three sections, the first indicates the operation mode, the second, the affected classes and the third, the affected properties. A class is referenced indicating the package containing it and its name, while a property is referenced indicating the package and class containing it and the property name (e.g., Figure 6).

The "ArchConfAsynchronousCall.yaml" configuration file specifies the properties to be added to the automatically created asynchronous classes.

It has a single section that allows to identify the classes to which the properties should be linked. Thus, since these classes must be specified before they are created, we identify them from elements of the PIM (i.e., the class and attribute that contains the asynchronous service that justifies the creation of the asynchronous class) (e.g., Figure 7).

Challenges of the M2M transformation: Among the challenges of the M2M transformation that have been tackled, we can mention:

1. The inability to work with profiles in the refining mode, which forced us to use a compiler different from the one proposed by default.
2. The asynchronous services identification, indispensable for automating the creation of asynchronous classes.

```
classes:
    - classPIM   : class_name_in_the_PIM_model
      atributePIM: property_name_in_the_PIM_model
      properties:
        - name      : name_of_the_new_property
          stereotype: stereotype_of_the_new_property
```

Fig. 7. ArchConfAsynchronousCall.yaml - Example

3. The correct configuration of the IDE in order to access native java methods created expressly for the processing of the configuration files.

After the automatic generation of the ASM of the application it is possible to go through the process of generating code. To do this, we have defined transformation rules from model to text using the Acceleo[8] tool.

These transformation rules follow a template-based approach in which text templates are specified with entries for data to be extracted from the model diagrams. The MTL[9] language was used for the definition of the templates, and OCL[10] to make queries to the model. In addition, services defined in Java were used to extend MTL with greater functionalities.

These rules are responsible for transforming elements defined in the logic and content diagrams to code in HTML5 and Javascript languages and in the jQuery, jQuery UI, and Datatables libraries. The complete transformation rules can be found in [43], Appendix 2.

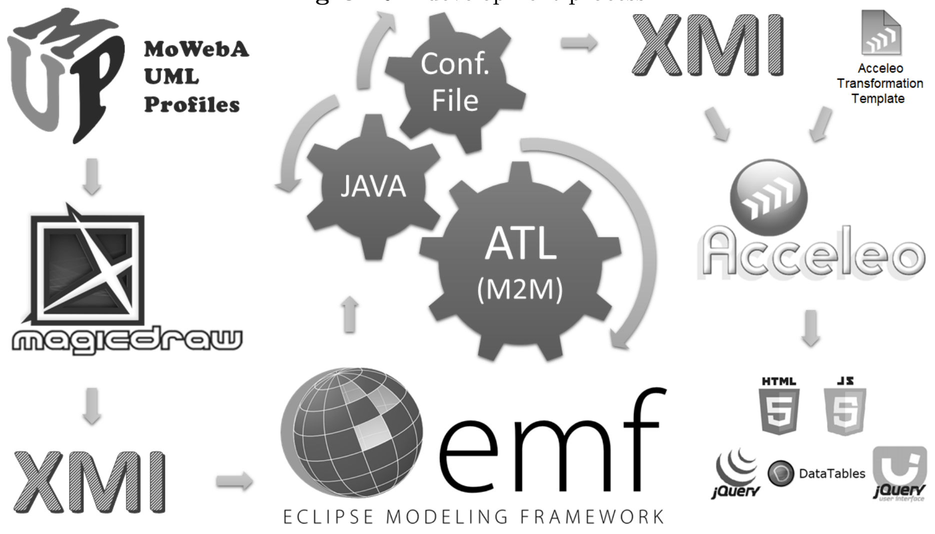
4.3 RIA development process

Figure 8 illustrates the proposed process for the development of RIA.

In order to model and subsequently generate a RIA, we start by using the MagicDraw[11] tool. After importing the MoWebA profiles into this tool, it is possible to create the initial PIM model. This model is exported to a file in XMI format, which is imported into the EMF[12] tool, where, through the mapping of the defined M2M transformation rules, it is transformed into a new XMI with diagrams corresponding to the ASM. Then this new XMI is imported into the Acceleo tool. The latter uses the transformation rules described in the previous section to perform transformations from model to text, generating the code in HTML5, Javascript, jQuery, Datatables and jQuery UI technologies, corresponding to the final implementation of the RIA. The proposed tools are available at `http://www.dei.uc.edu.py/proyectos/mddplus/herramientas/mowebaria/`.

[8]Acceleo.`https://eclipse.org/acceleo/`
[9]MTL.`http://www.omg.org/spec/MOFM2T/1.0/`
[10]OCL.`http://www.omg.org/spec/OCL/`
[11]MagicDraw.`http://www.nomagic.com/products/magicdraw.html`
[12]Eclipse Modeling Framework.`https://www.eclipse.org/modeling/emf/`

Fig. 8. RIA development process



4.4 Modeling and code generation example

This section presents an example of how to model and generate code of an application using this proposal. The application consists of a system of employee dialing, in which a guest user can register as an employee to later log in and be able to make inbound or outbound dialing. If the user is a supervisor, he is able to observe the entry and exit markings made by employees. For reasons of space, we only exemplify one diagram, which contemplates the RIA's feature of client business logic. The complete example that covers all the features can be found in [43], Chapter 4 (regarding the M2T transformations), and in [44], Chapter 4 and 5 (regarding the M2M transformations).

Figure 9 presents a diagram of the ASM for RIA, which takes advantage of the client business logic using the enriched table. There is a class with stereotype <<*richTable*>>, representing a table enriched with operations that can be performed on the client side. The tagged values are used to enable ordering, paging and searching on the client side. In addition, the maximum number of records per page (ten) is specified and the information in the table is enabled. Finally, the service *Get Markings* is set as the responsible for providing records to the table.

On the left of Figure 10 we can see the code that was generated from the previous diagram. This contains jQuery DataTables code associated with the enriched table *Marcaciones*, allowing to provide the different functionalities specified. On the right of Figure 10 we can see the rich table displayed.

Fig. 9. ASM for RIA diagram with client business logic operations

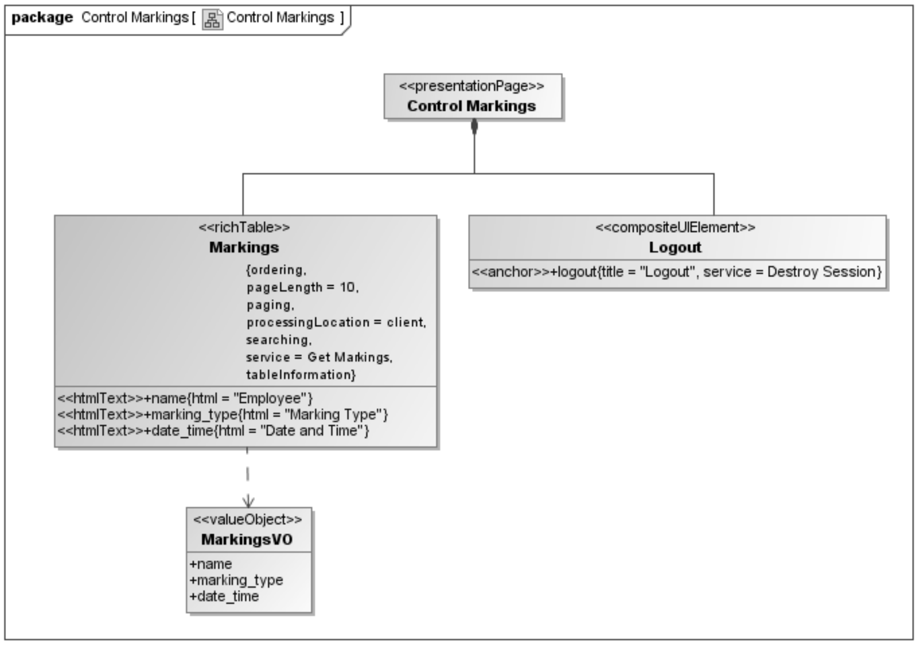
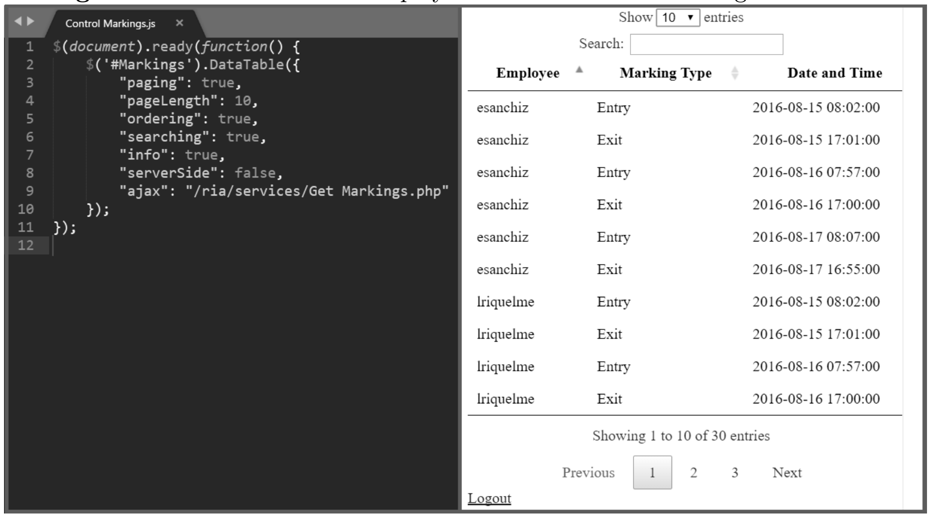

package Control Markings [Control Markings]
<<presentationPage>>
Control Markings
<<richTable>>
Markings
{ordering,
pageLength = 10,
paging,
processingLocation = client,
searching,
service = Get Markings,
tableInformation}
<<htmlText>>+name{html = "Employee"}
<<htmlText>>+marking_type{html = "Marking Type"}
<<htmlText>>+date_time{html = "Date and Time"}
<<compositeUIElement>>
Logout
<<anchor>>+logout{title = "Logout", service = Destroy Session}
<<valueObject>>
MarkingsVO
+name
+marking_type
+date_time

Fig. 10. Generated code and displayed table from the ASM diagram for RIA



```
$(document).ready(function() {
    $('#Markings').DataTable({
        "paging": true,
        "pageLength": 10,
        "ordering": true,
        "searching": true,
        "info": true,
        "serverSide": false,
        "ajax": "/ria/services/Get Markings.php"
    });
});
```

Show 10 entries

Search:

Employee	Marking Type	Date and Time
esanchiz	Entry	2016-08-15 08:02:00
esanchiz	Exit	2016-08-15 17:01:00
esanchiz	Entry	2016-08-16 07:57:00
esanchiz	Exit	2016-08-16 17:00:00
esanchiz	Entry	2016-08-17 08:07:00
esanchiz	Exit	2016-08-17 16:55:00
lriquelme	Entry	2016-08-15 08:02:00
lriquelme	Exit	2016-08-15 17:01:00
lriquelme	Entry	2016-08-16 07:57:00
lriquelme	Exit	2016-08-16 17:00:00

Showing 1 to 10 of 30 entries

5 A validation experience

We have performed two preliminary validation experiences of the proposal focusing on usability. The results of the former experience were presented in [16], and the results of the latter are presented in this paper. Even though both validations considered the evaluation of usability, we can mention the following differences between them: i) number of participants, and ii) scope of validation. The first experience was oriented to analyze the ASM-Code generation phase with 5 participants. In the second experience, 10 participants started the modeling process at the PIM level and got through every phase of the MoWebA development process.

The ISO 9241-11 [45] provides a guide on usability and defines it as "the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use". In addition, effectiveness is defined as "the accuracy and completeness with which users achieve specified goals". Efficiency is related to "the resources expended in relation to the accuracy and completeness with which users achieve goals". Satisfaction is defined as "the freedom from discomfort, and positive attitudes towards the use of the product".

Using the GQM template (Goal-Question-Metric) [46], the goal of this validation is stated as follows: *analyze* the MoWebA approach for the development of RIA, *for the purpose of* assessing its usability, *with respect to* effectiveness, efficiency and satisfaction, *from the viewpoint of* the developer, *in the context of* last year Computer Science students at the Catholic University (Asunción, Paraguay), who modeled the PIM and performed the PIM-ASM transformation and the ASM-code generation of the RIA code using this proposal.

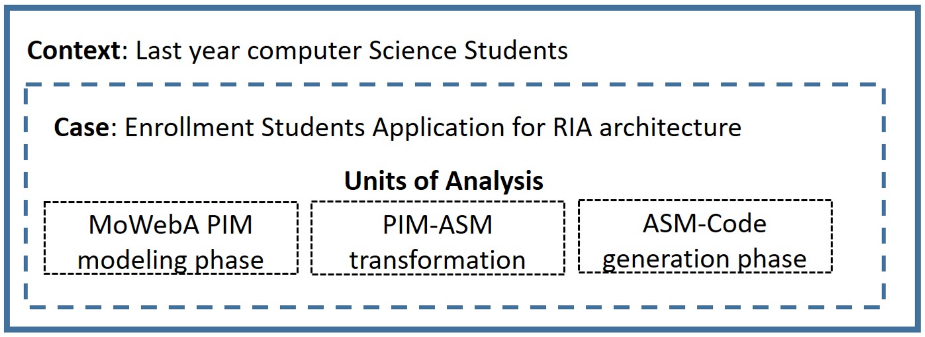
Based on this goal and the fact that an MDD approach can be separated into PIM modeling, PIM-ASM transformation, and code generation processes, the following research questions were established:
• **PI1**: What effectiveness, efficiency and satisfaction does the PIM modeling process of the proposed approach present?
• **PI2**: What effectiveness, efficiency and satisfaction does the PIM-ASM transformation process of the proposed approach present?
• **PI3**: What effectiveness, efficiency and satisfaction does the code generation process of the proposed approach present?

5.1 Case and units of analysis

The validation experience was structured taking into account a framework that Runeson et al. [47] have defined for case studies. According to Runeson, a case can be composed of one or more units of analysis. Furthermore, Yin distinguishes between holistic and embedded cases [48]. In this experience, we used a "unique-embedded case study" approach.

The case consists of a RIA development project using the proposed MDD approach. The requested application consists of a system for enrolling students

Fig. 11. Case and units of analysis


Context: Last year computer Science Students
Case: Enrollment Students Application for RIA architecture
Units of Analysis
MoWebA PIM modeling phase
PIM-ASM transformation
ASM-Code generation phase

in the Computer Science degree, composed of two pages with RIA elements. The experience was performed by ten students who were asked to model the necessary PIM diagrams, execute the PIM-ASM transformation rules to obtain the ASM models, make manual adjustments to the ASM models, generate code from them, and make manual adjustments to the generated code. The units of analysis are the PIM modeling phase, PIM-ASM transformation phase, and the code generation phase (see Figure 11).

5.2 Procedures

The experience was performed by students of the last year of the Computer Science degree at the Catholic University (Asunción, Paraguay). These students have skills in model-based processes, acquired in previous courses, and in model-driven processes, acquired in the course in which this validation experience was carried out.

In a first working session of two hours, the MoWebA for RIA proposal was presented and explained to the students, and they have had a supervised development of the PIM modeling, PIM-ASM transformation, and code generation process, using the employee dialing system example discussed above.

In a second working session of also two hours, the students were asked to create the PIM model for the enrolling students application, execute the PIM-ASM transformation rules, and generate the RIA code corresponding to the case presented in the previous section. The 10 students received a document with the requirements of the system. Then, they started the modeling process, which was timed to obtain data related to effectiveness. Subsequently, the participants proceeded to execute the PIM-ASM transformation rules, make manual adjustments to the ASM, generate the code of the application, and make manual adjustments to obtain the final and functional RIA. Again, the effort were measured in terms of time.

The models and code generated and adjusted by the students were corrected in order to obtain success rates (in relation to efficiency) for the modeling process and for the code generation process.

Furthermore, the 10 students answered three After Scenario Questionnaires (ASQ) [49], one for each task (PIM modeling, PIM-ASM transformation, and code generation). These questionnaires allow us to know the perception of the satisfaction of the participants with respect to the accomplishment of every task.

5.3 Data collection

From the validation experience, we obtained quantitative data to answer the research questions, and qualitative data (based on personal opinions) to improve the manifested perception.

We consider three information sources for data collection: i) project documentation: including PIM models without corrections, PIM model with corrections, generated code, and code of the modifications made to the application; ii) the time sheet of the working sessions; and iii) the questionnaires. The quantitative data were obtained from the project documentation, the time sheets and the ASQ questionnaires. On the other hand, the qualitative data were obtained from the opinions and/or comments of the participants and researchers.

The project documentation allowed us to determine: correct PIM elements modeled without researcher's intervention, number of corrections made to the models, code generated, code manually added. These data helped us to determine success rates for PIM modeling, PIM to ASM transformation and code generation. Subsequently, the time sheet allowed us to determine average completion time for PIM modeling, PIM to ASM transformation and code generation. Finally, the ASQ questionnaires were used to calculate the average satisfaction for the three phases.

For each phase, usability was measured by:

– Effectiveness: Average success rate for PIM modeling, PIM to ASM transformation and code generation (considering the project documentation).
– Efficiency: Average completion time for PIM modeling, PIM to ASM transformation and code generation (considering time sheets).
– Satisfaction: Average satisfaction for PIM modeling, ASM-Code transformation and code generation (considering ASQ questionnaires).

5.4 Data analysis and results

Next we discuss the experience and the results according to the research questions.

PI1: What effectiveness, efficiency and satisfaction does the PIM modeling process of the proposed approach present?

Table 2 presents the usability measurements for the PIM modeling process carried out during the experience. On average, we observed that students completed the application model with a success rate of 95.2%, in 17 minutes, and with an ASQ score of 2.87 (ASQ scores range from 1 to 7, and values closer to 1 are those that indicate a higher level of satisfaction).

Table 2. Usability measurements for the PIM modeling process

Process	**Average Success Rate**	**Average Completion Time**	**Average Satisfaction**
Modeling	95.2%	17 min.	2.87

Table 3. Usability measurements for the PIM-ASM transformation process

Process	**Average Success Rate**	**Average Completion Time**	**Average Satisfaction**
Modeling	100%	9.2 min.	2.70

Analyzing these results, we can see that a satisfactory success rate was obtained. The biggest difficulty at the time of modeling was given by misapplying some labeled values, stereotypes and forgetting to include a submit button on the forms. In these cases, intervention from researchers were made in order to correct errors.

Regarding modeling time, although we can not affirm with certainty that it corresponds to a favorable or unfavorable time, because it is necessary to compare this approach against another, from our experience in modeling and development, it seems to us that this time is reasonable.

The score obtained from the ASQ questionnaire reflects a good level of satisfaction from the students using the proposed modeling process.

From the above, we can derive that good effectiveness, efficiency and satisfaction were obtained in the modeling process.

PI2:What effectiveness, efficiency and satisfaction does the PIM-ASM transformation process of the proposed approach present?

The PIM-to-ASM transformation phase of the experience included two tasks: applying the ATL rules to automatically generate the ASM, and performing some manual adjustments to the ASM model in order to change default values established by the transformation rules (e.g., change the value of the autocomplete tagged valued to TRUE, since the default value of the transformation rule is set to FALSE). The first task was performed without researchers intervention, but the second task needed some help from researchers, in order to identify the tagged values that needed to be changed.

Table 3 presents the usability measurements for the PIM-ASM transformation process of the experience. On average, the automatic MTM transformation was completed with a success rate of 100%, in 9.2 minutes, and with an ASQ score of 2.70.

From the above, we can derive that very good levels of effectiveness, efficiency and satisfaction were obtained in the PIM-ASM transformation process.

Some additional considerations arise from comparing the present results with those obtained in the previous validation experience [16]. In the previous experience, the scope of the modeling process was limited to generate the ASM. Mean-

Table 4. Usability measurements for the code generation process

Process	**Average Success Rate**	**Average Completion Time**	**Average Satisfaction**
Code Generation	90%	8.1 min.	2.87

while, in the present experience, the modeling process covered both, the PIM models and the PIM-to-ASM transformation process (i.e., the results presented in PI1 and PI2). Moreover, despite being quite reduced, the present experience doubled the number of participants, making the measurements more significant. Hence, comparing the average modeling time of the previous experience (i.e., 44,2 minutes) with the present results (i.e., sum of the average time from Table 2 and Table 3; that is 26,2 minutes) it is worth noting that the efficiency was improved. A natural explanation for such improvement comes from the automatic transformation from PIM-to-ASM, which reduced the modeling effort. In a complementary way, the satisfaction level is very good and quite similar in both experiences.

PI3: What effectiveness, efficiency and satisfaction does the code generation process of the proposed approach present?

Table 4 presents the usability measurements for the code generation process of the experience. On average, automatic code generation was completed with a success rate of 90%, in 8.1 minutes, and with an ASQ score of 2.87.

From these results, we can note that the success rate is just as satisfactory as the success rate obtained for the modeling process. However, we believe that the success rate for the code generation process could have obtained a better score, since this was directly affected by the input model used. Although several lines of code were generated from all models, these lines were not complete, due to imperfections of the models developed by participants, used as input for the code generator. We emphasize that in-situ corrections of the elaborated models were carried out, but not all the necessary corrections were detected due to the limitation of time in the working session.

In relation to the time of code generation, we can observe that a quite reduced time was achieved. This time could be generalized for the generation of all types of applications since it requires strictly mechanical and predefined steps, without great variation in the intervention of the developer. Also, this time is independent from the size of the application to be generated, because an increase in the size of the application would not generate a significant increase in time.

Therefore, we can conclude that the code generation process was carried out with good effectiveness, efficiency and satisfaction.

5.5 Threats to validity

Regarding internal validity, which has to do with the degree of confidence in a cause-effect relationship between the factors of interest and the observed results, it can be said that the validation experience was carried out with students, who all have the same level of experience in terms of an MDD process, thus avoiding participants with unbalanced knowledge. To avoid plagiarism the students were supervised and communication between pairs was forbidden.

The external validity, which represents the degree to which the results achieved can be generalized, is affected by the fact that the experience was developed with students, which does not allow us to ensure that the results can be generalized to a target population corresponding to the web application developers that use MDD. In addition, the number of participants involved in this experience was ten (doubling the number of the first experience presented in [16]), and although it does not correspond to a sufficient amount for statistical purposes, it reaches at least to get first judgments, which then must be corroborated with additional rigorous experiments and / or experiences with greater number of participants. In addition, the developed case consisted of a limited case, however, this case contemplates the development of a RIA with all the features that have been mentioned in this work.

Regarding the validity of the construct, which reflects the extent to which the measures have been adapted to what the researcher has in mind and what is being investigated, we selected data that are normally used to measure quality aspects. We also used standard questionnaires, which are considered reliable and valid [49,50] to evaluate the students' perceptions without the intervention of the researcher.

In relation to reliability, which indicates the dependence of the data and its analysis on a specific researcher and the ability to replicate the same study and obtain the same results, we respected the literalness of the data obtained in the documentation, in the measured times and in the questionnaires, avoiding the introduction of biases through interpretation.

6 Conclusion

In this work we presented a solution based on MDD for the development of RIA, and more generally, for rich Web applications, incorporating features of client data, client business logic and asynchronous communication between client and server, adopting the MDA standard and achieving, through the execution of the defined M2M transformation rules, the automatic generation of the ASM from the PIM.

The proposal is based on MoWebA and extends its models and tools through an Architecture Specific Model (ASM). We present the PIM-ASM transformation process with ATL. For the generation of the code corresponding to the final RIA, unlike the other existing proposals, scripting-based technologies (HTML5, Javascript, jQuery, jQuery Datatables and jQuery UI) have been adopted, which are the most commonly technologies used for the development of RIA.

The preliminary validation, considering the PIM modeling process, PIM-ASM transformation process and the subsequent code generation of the resulting rich Web application, yields positive results in relation to the completion of the modeling tasks, the development times, and the satisfaction of the participants.

Further validation of the proposal is still pending, such as formal experiments or case studies in an industrial or commercial context.

Acknowledgements

This work has been funded by CONACYT through the PROCIENCIA program, with resources from "Fondo para la Excelencia de la Educación e Investigación - FEEI", FONACIDE. This work has been developed under project "Mejorando el proceso de desarrollo de software: propuesta basada en MDD" (14-INV-056).

References

1. P. Fraternali, G. Rossi, and F. Sánchez-Figueroa, "Rich internet applications," *Internet Computing, IEEE*, vol. 14, no. 3, pp. 9–12, 2010. [Online]. Available: https://doi.org/10.1109/mic.2010.76
2. M. Busch and N. Koch, "Rich internet applications. state-of-the-art," Ludwig-Maximilians-Universität München, München, Germany, techreport 0902, 2009.
3. G. Toffetti, S. Comai, J. C. Preciado, and M. Linaje, "State-of-the-art and trends in the systematic development of rich internet applications," *Journal of Web Engineering*, vol. 10, no. 1, pp. 070–086, 2011.
4. M. Brambilla, J. Cabot, and M. Wimmer, "Model-driven software engineering in practice," *Synthesis Lectures on Software Engineering*, vol. 1, no. 1, pp. 1–182, 2012. [Online]. Available: https://doi.org/10.2200/s00441ed1v01y201208swe001
5. OMG, "Mda specifications," available at http://www.omg.org/mda/specs.htm. [Online]. Available: http://www.omg.org/mda/specs.htm
6. E. Yigitbas, B. Mohrmann, and S. Sauer, "Model-driven ui development integrating hci patterns." *LMIS@ EICS*, vol. 2015, pp. 42–46, 2015.
7. R. Esbai and M. Erramdani, "Model-to-model transformation in approach by modeling: From uml model to model-view-presenter and dependency injection patterns," in *Information and Communication Technologies (WICT), 2015 5th World Congress on.* IEEE, 2015, pp. 1–6. [Online]. Available: https://doi.org/10.1109/wict.2015.7489648
8. S. Roubi, M. Erramdani, and S. Mbarki, "Modeling and generating graphical user interface for mvc rich internet application using a model driven approach," in *Information Technology for Organizations Development (IT4OD), 2016 International Conference on.* IEEE, 2016, pp. 1–6. [Online]. Available: https://doi.org/10.1109/it4od.2016.7479249
9. N. Laaz and S. Mbarki, "A model-driven approach for generating ria interfaces using ifml and ontologies," in *Information Science and Technology (CiSt), 2016 4th IEEE International Colloquium on.* IEEE, 2016, pp. 83–88. [Online]. Available: https://doi.org/10.1109/cist.2016.7805005
10. A. Bozzon, S. Comai, P. Fraternali, and G. T. Carughi, "Capturing RIA concepts in a web modeling language," in *Proceedings of the 15th international conference on World Wide Web, WWW 2006, Edinburgh, Scotland, UK, May 23-26, 2006*, L. Carr, D. D. Roure, A. Iyengar, C. A. Goble, and M. Dahlin, Eds. ACM, 2006, pp. 907–908. [Online]. Available: http://doi.acm.org/10.1145/1135777.1135938

11. N. Koch, M. Pigerl, G. Zhang, and T. Morozova, "Patterns for the Model-Based Development of RIAs," in *Web Engineering, 9th International Conference, ICWE 2009, San Sebastián, Spain, June 24-26, 2009, Proceedings*, ser. Lecture Notes in Computer Science, M. Gaedke, M. Grossniklaus, and O. Díaz, Eds., vol. 5648. Springer, 2009, pp. 283–291. [Online]. Available: https://doi.org/10.1007/978-3-642-02818-2_23
12. S. Meliá, J. Gómez, S. Pérez, and O. Díaz, "A model-driven development for gwt-based rich internet applications with OOH4RIA," in *Proceedings of the Eighth International Conference on Web Engineering, ICWE 2008, 14-18 July 2008, Yorktown Heights, New York, USA*, 2008, pp. 13–23. [Online]. Available: http://dx.doi.org/10.1109/ICWE.2008.36
13. M. González, L. Cernuzzi, and O. Pastor, "A navigational role-centric model oriented web approach-moweba," *International Journal of Web Engineering and Technology*, vol. 11, no. 1, pp. 29–67, 2016. [Online]. Available: https://doi.org/10.1504/ijwet.2016.075963
14. T. Mikkonen, R. Pitkänen, and M. Pussinen, "On the Role of Architectural Style in Model Driven Development," in *Software Architecture, First European Workshop, EWSA 2004, St Andrews, UK, May 21-22, 2004, Proceedings*, 2004, pp. 74–87. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-24769-2_6
15. I. López, M. González, N. Aquino, and L. Cernuzzi, "Una propuesta basada en model driven architecture para el soporte de rich internet applications," in *Memorias de la XIX Conferencia Iberoamericana de Software Engineering*, Ecuador, Quito, Apr. 2016, pp. 25–38.
16. G. Nunez, M. González, N. Aquino, and L. Cernuzzi, "A model-driven approach to develop rich web applications," in *2017 XLIII Latin American Computer Conference, CLEI 2017, Córdoba, Argentina, September 4-8, 2017*, 2017, pp. 1–10. [Online]. Available: https://doi.org/10.1109/CLEI.2017.8226424
17. A. Bozzon, S. Comai, P. Fraternali, and G. T. Carughi, "Conceptual modeling and code generation for rich internet applications," in *Proceedings of the 6th international conference on Web engineering.* ACM, 2006, pp. 353–360. [Online]. Available: https://doi.org/10.1145/1145581.1145649
18. M. Linaje, J. C. Preciado, and F. Sánchez-Figueroa, "A method for model based design of rich internet application interactive user interfaces," in *Web Engineering.* Springer, 2007, pp. 226–241. [Online]. Available: https://doi.org/10.1007/978-3-540-73597-7_18
19. J. C. Preciado, M. Linaje, S. Comai, and F. Sanchez-Figueroa, "Designing rich internet applications with web engineering methodologies," in *Web Site Evolution, 2007. WSE 2007. 9th IEEE International Workshop on.* IEEE, 2007, pp. 23–30. [Online]. Available: https://doi.org/10.1109/wse.2007.4380240
20. G. T. Carughi, S. Comai, A. Bozzon, and P. Fraternali, "Modeling distributed events in data-intensive rich internet applications," in *Web Information Systems Engineering–WISE 2007.* Springer, 2007, pp. 593–602. [Online]. Available: https://doi.org/10.1007/978-3-540-76993-4_51
21. M. Brambilla, J. C. Preciado, M. Linaje, and F. Sanchez-Figueroa, "Business process-based conceptual design of rich internet applications," in *Web Engineering, 2008. ICWE'08. Eighth International Conference on.* IEEE, 2008, pp. 155–161. [Online]. Available: https://doi.org/10.1109/icwe.2008.22
22. F. Valverde and O. Pastor, "Facing the technological challenges of web 2.0: A ria model-driven engineering approach," in *International Conference on Web Information Systems Engineering.* Springer, 2009, pp. 131–144. [Online]. Available: https://doi.org/10.1007/978-3-642-04409-0_18

23. S. Meliá, J. Gómez, S. Pérez, and O. Díaz, "Architectural and technological variability in rich internet applications," *IEEE Internet Computing*, vol. 14, no. 3, pp. 24–32, 2010. [Online]. Available: https://doi.org/10.1109/mic.2010.63
24. M. Huber and P. Brune, "Model-driven development of interactive web user interfaces with html5." in *MODELSWARD*, 2013, pp. 249–252. [Online]. Available: https://doi.org/10.5220/0004311202490252
25. J. L. H. Agustin and P. C. Del Barco, "A model-driven approach to develop high performance web applications," *Journal of Systems and Software*, vol. 86, no. 12, pp. 3013–3023, 2013. [Online]. Available: https://doi.org/10.1016/j.jss.2013.07.028
26. R. Esbai, M. Erramdani, and S. Mbarki, "Model-driven transformation for gwt with approach by modeling: From uml model to mvp web applications," *International Review on Computers and Software (I. RE. CO. S.)*, vol. 9, no. 9, pp. 1612–1620, 2014. [Online]. Available: https://doi.org/10.15866/irecos.v9i9.3361
27. M. L. Bernardi, G. A. Di Lucca, and D. Distante, "Model-driven fast prototyping of rias: From conceptual models to running applications," in *Advances in Computing, Communications and Informatics (ICACCI, 2014 International Conference on.* IEEE, 2014, pp. 250–258. [Online]. Available: https://doi.org/10.1109/icacci.2014.6968522
28. S. Roubi, M. Erramdani, and S. Mbarki, "A model driven approach to generate graphical user interfaces for rich internet applications using interaction flow modeling language," in *Intelligent Systems Design and Applications (ISDA), 2015 15th International Conference on.* IEEE, 2015, pp. 272–276. [Online]. Available: https://doi.org/10.1109/isda.2015.7489237
29. J. L. H. Agustin, "Model-driven web applications," in *Science and Information Conference (SAI), 2015.* IEEE, 2015, pp. 954–964. [Online]. Available: https://doi.org/10.1109/sai.2015.7237258
30. D. Bonhaure, M. González, N. Aquino, L. Cernuzzi, and C. Pons, "Exploring model-to-model transformations for ria architectures by means of a systematic mapping study," *CLEI Electronic Journal*, vol. 20, no. 3, 12 2017, (to apper). [Online]. Available: https://doi.org/10.19153/cleiej.20.3.5
31. D. Manset, H. Verjus, R. McClatchey, and F. Oquendo, "A formal architecture-centric model-driven approach for the automatic generation of grid applications," in *ICEIS 2006 - Proceedings of the Eighth International Conference on Enterprise Information Systems: Databases and Information Systems Integration, Paphos, Cyprus, May 23-27, 2006*, 2006, pp. 322–330. [Online]. Available: https://doi.org/10.5220/0002443503220330
32. M. Broy, "Architecture driven modeling in software development," in *9th International Conference on Engineering of Complex Computer Systems (ICECCS 2004), 14-16 April 2004, Florence, Italy*, 2004, pp. 3–12. [Online]. Available: http://dx.doi.org/10.1109/ICECCS.2004.1310898
33. E. Marcos, C. J. Acuña, and C. E. Cuesta, "Integrating software architecture into a MDA framework," in *Software Architecture, Third European Workshop, EWSA 2006, Nantes, France, September 4-5, 2006, Revised Selected Papers*, 2006, pp. 127–143. [Online]. Available: http://dx.doi.org/10.1007/11966104_10
34. M. L. Sanz and E. Marcos, "Archimedes: A model-driven framework for the specification of service-oriented architectures," *Inf. Syst.*, vol. 37, no. 3, pp. 257–268, 2012. [Online]. Available: http://dx.doi.org/10.1016/j.is.2011.11.002
35. M. López-Sanz and E. Marcos, *Modeling Platform-Independent and Platform-Specific Service Architectures with UML and the ArchiMeDeS Framework.* IGI Global, 2014, ch. 11, pp. 254–277. [Online]. Available: https://doi.org/10.4018/978-1-4666-6026-7.ch011

36. N. Elleuch, A. Khalfallah, and S. B. Ahmed, "Archmde approach for the formal verification of real time systems," in *11th IEEE International Conference on Computer and Information Technology, CIT 2011, Pafos, Cyprus, 31 August-2 September 2011*, 2011, pp. 533–538. [Online]. Available: http://dx.doi.org/10.1109/CIT.2011.39
37. M. González, J. Casariego, J. J. Bareiro, L. Cernuzzi, and O. Pastor, "A MDA approach for navigational and user perspectives," *CLEI Electron. J.*, vol. 14, no. 1, 2011. [Online]. Available: http://www.clei.org/cleiej/paper.php\?id=208
38. M. Brambilla, J. Cabot, and M. Wimmer, *Model-Driven Software Engineering in Practice*, 1st ed., ser. Synthesis Lectures on Software Engineering. Morgan & Claypool Publishers, 2012. [Online]. Available: https://doi.org/10.2200/S00441ED1V01Y201208SWE001
39. M. González, L. Cernuzzi, N. Aquino, and O. Pastor, "Developing web applications for different architectures: The moweba approach," in *Tenth IEEE International Conference on Research Challenges in Information Science, RCIS 2016, Grenoble, France, June 1-3, 2016*, 2016, pp. 1–11. [Online]. Available: http://dx.doi.org/10.1109/RCIS.2016.7549344
40. F. O. García Rubio, J. M. Vara Mesa, and C. V. Chicote, *Desarrollo de Software Dirigido por Modelos: Conceptos, Métodos y Herramientas.* Ra-Ma Editorial, 2013.
41. F. Jouault and D. Wagelaar, "ATL EMF Transformation Virtual Machine (research VM) - Performance," https://wiki.eclipse.org/ATL/EMFTVM\#Performance, 2017, accessed at 19-02-2017.
42. ——, "ATL EMF Transformation Virtual Machine (research VM) - Invoking native Java methods," https://wiki.eclipse.org/ATL/EMFTVM\#Invoking_native_Java_methods, 2017, accessed at 19-02-2017.
43. G. Nuñez, M. González, and L. Cernuzzi, "Un enfoque MDD para el desarrollo de RIA," Proyecto final, Universidad Católica "Nuestra Señora de la Asunción", 2017, available at http://www.dei.uc.edu.py/proyectos/mddplus/documentos/.
44. D. Bonhaure, M. González, N. Aquino, and C. Pons, "Una propuesta de transformación M2M para el análisis de la fase ASM de MoWebA," Master Thesis, Universidad Católica "Nuestra Señora de la Asunción", Universidad Nacional de La Plata, 2017, available at http://www.dei.uc.edu.py/proyectos/mddplus/documentos/.
45. S. ISO, "9241-11. 1998," *Ergonomic Requirements for Office Work with Visual Display Terminals (VDTs)–Part II Guidance on Usability*, 1998.
46. V. R. Basili and H. D. Rombach, "The tame project: Towards improvement-oriented software environments," *IEEE Transactions on software engineering*, vol. 14, no. 6, pp. 758–773, 1988. [Online]. Available: https://doi.org/10.1007/3-540-27662-9_8
47. P. Runeson, M. Höst, A. Rainer, and B. Regnell, *Case Study Research in Software Engineering - Guidelines and Examples.* Wiley, 2012. [Online]. Available: http://eu.wiley.com/WileyCDA/WileyTitle/productCd-1118104358.html
48. R. K. Yin, "Case study research: Design and methods . thousands oaks," *Sage. Young, LC and Wilkinson, IR (1989). The role of trust and co-operation in marketing channels: a preliminary study. European Journal of Marketing*, vol. 23, no. 2, pp. 109–122, 2003.
49. J. R. Lewis, "Psychometric evaluation of an after-scenario questionnaire for computer usability studies: the asq," *ACM SIGCHI Bulletin*, vol. 23, no. 1, pp. 78–81, 1991. [Online]. Available: https://doi.org/10.1145/122672.122692
50. J. Sauro, "Measuring usability with the system usability scale (sus)," 2011.