A framework for web applications using an Agile and Collaborative Model Driven Development (AC-MDD)

Breno Lisi Romano[*] **and Adilson Marques da Cunha**

Instituto Federal de Educação, Ciência e Tecnologia de São Paulo, Rua Doutor João Batista Merlin, s/n., 13872-551, São João da Boa Vista, São Paulo, Brazil. *Author for correspondence. Email: blromano@ifsp.edu.br

ABSTRACT. This paper presents, as its main contribution, a Framework for Web Applications named Agile and Collaborative Model Driven Development (AC-MDD), seeking to reduce wastes on the modeling and documentation activities of web applications. The AC-MDD framework was elaborated based on related works that present Agile Model Driven Development processes (AMDD), but it deals with issues not yet considered by these researches. To adopt the AC-MDD framework, some visual elements from a novel Unified Modeling Language (UML) profile named Web-AML was designed, allowing to represent agile models of web applications. Besides that, in order to apply the proposed framework using these new models, a method was defined providing steps to transform agile models into web application source-codes. A case study was performed successfully in a real project that gives support to the population of a city located in the State of São Paulo (Brazil) to deal with the real flood problem they have been facing in recent years. This project was developed at the Federal Institute of Education, Science, and Technology of São Paulo to verify and validate the AC-MDD framework. Finally, a comparative analysis between the AC-MDD Framework and other AMDD processes was presented and highlighted the main aspects of the proposed framework.

Keywords: agile; web applications; floods; development; AMDD.

Introduction

Some authors emphasize that web applications are complex systems based on a variety of hardware and software components, languages, interfaces, and standards, which have evolved into sophisticated computing tools that not only provide stand-alone functions to the end user, but are also integrated with corporate databases and business applications (Pressman & Maxim, 2014; Huang & Chu, 2014).

Additionally, the development of these Web applications is under pressure due to the evolution of businesses' needs, short deadlines, and limited resources. To deal with these issues, a different approach is required, in order to turn business requirements into software artifacts and handle the constant evolution of Internet application workflow (Liang, Marmaridis, & Ginige, 2007).

In this context, agile methodologies began being used to develop Web applications. These methodologies represent project management approaches that guide the production process of computer systems, being flexible to the environment dynamics that are common in software development while generating value and innovation (Qumer & Henderson-Sellers, 2008).

A research conducted in 2017 by VersionOne with people from different software development industries of the world has shown that at least 90% of companies' projects already used agile methodologies. The most common were Scrum, Kanban, and eXtreme Programming (XP) (VersionOne, 2017). This research presented the following greatest concerns about adopting agile in new projects: lack of documentation, lack of up-front planning and loss of management, among others. Besides that, the top three benefits they are looking when adopting agile have remained steady for the past five years: manage changing priorities, team productivity, and project visibility.

As it can be noticed, the lack of a proper documentation is one of the major worries in software development companies. Modeling is an important activity in every software development project and, if it is performed properly, can improve the software documentation.

Given this scenario, we must search for some ways to overcome these difficulties, while developing software to leverage advantages of modeling and documenting without facing the inconveniences. In this

matter, we can use the Agile Modeling (AM) method, which defines a set of values, principles, and practices that describe how to simplify modeling and documenting activities as part of development teams' daily routine (Ambler, 2007).

It is worth mentioning that the goal of AM is to optimize the modeling process to make it more efficient. Due to this reason, AM is a method based on good practices rather than on a prescriptive method (Ambler, 2014). In this sense, it does not define detailed procedures on how to create a certain kind of model. Rather, it provides good practices on how to be efficient as a system analyst, in order to create agile models.

As highlighted by Ambler (2007), what makes AM a catalyst to improve software documentation and modeling are not the modeling techniques, but how to apply them to improve productivity.

Another interesting remark about agile methodologies practices, which was also reported on VersionOne research in 2017, is their constant search for productivity increase in software development (VersionOne, 2017).

To increase productivity, the Agile Model Driven Development (AMDD) was created. It represents the agile version of the Model Driven Development (MDD). MDD is a software development approach in which the extensive creation of models is made before the source-code is written (Ambler, 2008).

Still according to Ambler (2007), the goal of MDD is to offer the creation of understandable models. Generating software from them is not possible to every development team. In this context, AMDD becomes a more realistic approach, since its objective is to describe how developers and stakeholders can work together to create models that are 'just barely good enough'.

Since web applications are the main focus of this paper, its development needs more agile software development methodologies with short iterative cycles – because of short deadlines. We can use the AMDD approach to leverage both techniques based on models and agility. This approach implies shorter iterations, encourages stakeholders' participation and allows requisites to evolve along the project (Liang et al., 2007).

Moreover, there are many cases where the MDD is used to define transformation mechanisms to allow the generation of source-code in a specific programming language from high-level abstraction models (Romano, Silva, Cunha, & Mourao, 2010).

For the generation of source-code from models, they must be conceived in a standardized language. Usually, the Unified Modeling Language (UML) is used. Besides being the most popular software modeling language (Rumpe, 2014), it represents a visual language with well-defined semantics to create software structures used to visualize, specify, build, and document the created artifacts (Moreno, Fraternali, & Vallecillo, 2007).

On the other hand, UML does not encompass every aspect of the web application modeling. One of its deficiencies is the lack of abstractions that represent user interfaces and their interactions.

In this context, we identified the need to search for new solutions that combine the agile modeling practices and MDD to create web applications using a specific language for its representation. We also wanted to solve the difficulties mentioned before and improve productivity by generating source-codes from systems' models.

Therefore, the goal of this paper was to present a Framework for Web Applications named Agile and Collaborative Model Driven Development (AC-MDD). Such framework should improve the productivity in developing applications, be consistent with the requirements, and generate source-codes from agile models. It tackles waste reduction of resources on the modeling and documentation stages of web applications, involving collaborative work among stakeholders.

To reach this goal, some visual constructs from a novel UML profile named Web Agile Modeling Language (Web-AML) was conceptualized and implemented. To apply the AC-MDD Framework using these new visual constructions, a WebAC-MDD Method was defined providing all steps to transform agile models into Web application source-codes.

Related works

This section presents an overview of some processes addressing issues related to agile model driven development. Besides that, some approaches are also presented for Web Application modeling. Finally, a survey of some related works of this research with direct intersection with the AC-MDD Framework were investigated and detailed.

AMDD processes

As the name suggests, AMDD is the agile version of MDD. According to Amber (2007), the goal of MDD is to allow the creation of understandable models. Generating software from these models, however, is not possible for every development team. In this context, AMDD is a more realistic approach so that its goal is to

describe how developers and stakeholders can work together to create models that are 'just barely good enough'.

Another AMDD process is the Sage process that uses MDD for an iterative and incremental approach to develop high confidence reactive multi-agent systems (Kirby Jr, 2006). Kirby Jr (2006) emphasizes that the Sage process takes advantages of the UML class diagrams to represent elements from different models. Besides, the process defines attributes for each class, making it possible for them to interact with one another (Kirby Jr, 2006).

Finally, the System Level Agile Process (SLAP) is an agile methodology based on Scrum, developed and approved by Motorola (Zhang & Patel, 2011). Motorola's MDD is a process based on the V-Model and includes the following development activities: Application and Architecture Requirements; Project Requirements Analysis; Detailed Project; Source-Code Generation; and Tests.

This process seeks to increase the development pace, to guarantee frequent deliver of functionalities and to improve product quality (Zhang & Patel, 2011). Further, this process uses SLAP and MDD, both used by Motorola, and establishes a simple correspondence between Sprints' SLAP and Motorola's MDD process.

Approaches to web application modeling

In order to deal with Web Application development and its aspects, Web Engineering came up aiming to represent applications with systematic and quantifiable approaches, in order to provide economic development and produce high quality solutions (Wakil, Jawawi, & Safi, 2015).

In Web Engineering, there are approaches that can be adopted to develop Web Applications. Some approaches are presented in this subsection. UML-based Web Engineering (UWE) is an approach presented by Wakil et al. (2015). This method represents a Software Engineering approach to Web Applications, based on the Unified Process, aiming to cover the life cycle for the development of these applications.

An UML profile is created by UWE consisting of stereotypes and attributes that describe the modeling aspects required to model different web application elements, such as presentation, navigation, and adaptation (Kroiss & Koch, 2008).

The Web Modeling Language (WebML) is also an approach that defines a visual language to specify both the web application content structure and the organization and presentation of its content in one or more hypertexts. Using WebML, web application design begins with the data schema specification, representing its contents' organization (Wakil, Jawawi, & Safi, 2015). The WebML Data Model adopts the elements of the Entity-Relationship Model. On the other hand, the WebML Hypertext Model allows describing how content, previously specified in the Data Model, is published in the web application.

Finally, the Interaction Flow Modeling Language (IFML) is an approach that was created based on the WebML experience and seeks to generalize its use for any type of software platform. This language was adopted by the OMG standard in March 2013, and its first version was completed in March 2014 (Schauerhuber, Wimmer, Schwinger, Kapsammer, & Retschitzegger, 2007). The IFML was designed to represent the content, the user interaction, and the behavior control of applications belonging to different domains.

Discussions on related works

With the AMDD processes investigation presented in section II-A, it is possible to emphasize that none of them were conceptualized focusing on web applications, which is the purpose of this research and requires specific activities to develop these applications.

The AMDD High-Level Life Cycle process is applied for the agile development of general software projects. The Sage process was proposed for use in agile approaches to high assurance software. The MDD-SLAP process was designed benefiting from both agile and MDD advantages in developing real-time telecommunication systems.

In addition, investigated AMDD processes do not consider an appropriate collaborative work with decision-making among stakeholders involved in software development. It can also be noticed that these processes use UML to model applications contemplated by them. However, it is worth mentioning that UML does not present specific aspects for web application modeling.

Considering the approaches to web application modeling, it can be noticed that all of them focus to model pages structures (contents) and navigability among pages, failing to consider in more detail characteristics of business requirements and software testing. Additionally, these approaches were not proposed to handle agile practices and AM, which are part of the research scope.

In this way, the investigation of the related works presented in this section motivated the design of the AC-MDD Framework, WebAC-MDD Method, and the Web-AML profile, in order to improve the productivity in developing web applications, be consistent with the requirements, and generate source-codes from agile models. Moreover, it tackles waste reduction of resources in the modeling and documentation stages of web applications, involving collaborative work among stakeholders.

The AC-MDD framework

Using theoretical concepts and related works, we conceived a Framework for Web Applications using an Agile and Collaborative Model-Driven Development (AC-MDD). This framework is shown in Figure 1 and corresponds to the agile model driven development of Web application scenario, on the basis of the life cycle of AMDD, and the Scrum and the XP agile development process.

We must start the development of a Web application with the agile modeling activity in a business level (high level abstraction) of the application. We must then model both the technical requirements and user stories (sub activity 0.1) and application's architecture and technology (sub activity 0.2).

Since this activity occurs only once and aims to better contextualize the main functional and non-functional requirements of Web applications, we say that the business agile modeling is in Sprint 0 (initial phase) of the development. The parts involved in this first moment are those of system analyst and stakeholders, with a collaborative work making decisions in order to minimize waste in requirements gathering. This involvement is justified by the fact that they have specific modeling abilities, as well as domain knowledge to be used to develop the application.
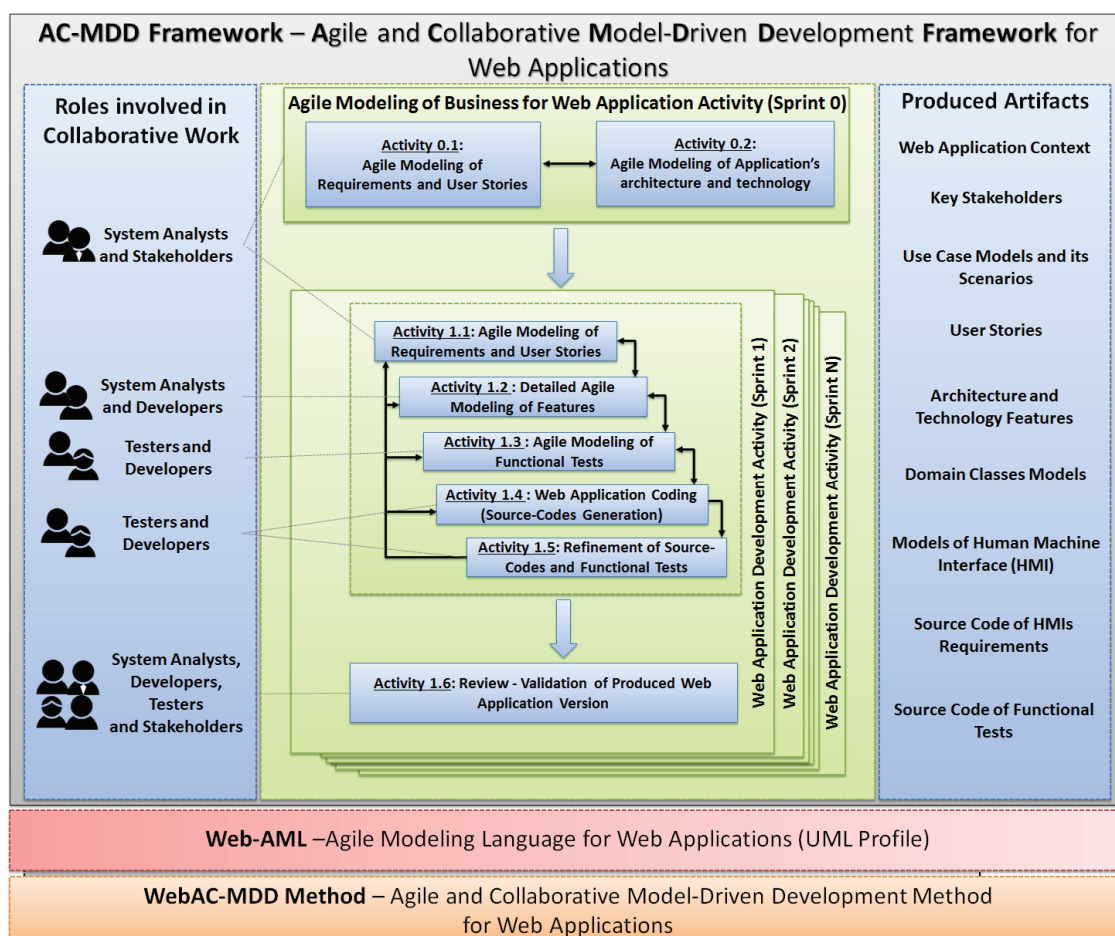
Figure 1. The Agile and Collaborative Model-Driven Development Framework (AC-MDD Framework).

In the first activity, the 'Web Application Business Agile Modeling', we must define a set of technical requirements and user stories to be refined, modeled, developed, and tested in a Sprint. Its definition must be done considering the technical abilities of the development team and their workload. Due to this, there may exist several Sprints, depending on the complexity of the Web application and the hours needed by the development team.

Once the set of technical requirements and user stories is chosen, the system analysts and stakeholders initiate a refinement of their respective modeling in sub activity 1.1 from the 'Web Application Development' activity, applying the principles and practices of AM and focusing only on the current Sprint.

In the next step, and without the participation of the stakeholders, system analysts must work together with developers to fulfill sub activity 1.2, the 'Detailed Agile Modeling of Features', which must be done in the Sprint. If some refinement is required at the end of this sub activity, we suggest returning to sub activity 1.1, 'Agile Modeling of Requirements and User Stories', to reduce the occurrence of future problems in future stages.

The sub activity 'Detailed Agile Modeling of Features' uses the concept of Just in Time (JIT), following the same suggestion from AMDD to model only in time and only enough to solve the problem. The main outputs from this activity are the main classes model and the Computer Human Interfaces (CHIs), considering the features of the application.

Testers and developers do the next sub activity, which represents the 'Agile Modeling of Functional Tests' of the Web application (sub activity 1.3). In this activity, they must link to the user stories of agile test modeling to verify system's behavior. They must focus on the functional tests and browsing experience, as well as the successful and alternative scenarios from the use cases. Once again, if they identify that a refinement is required, at the end of this activity we suggest them to go back to the Sub Activity 1.2.

Once the test modeling of the Web application is finished, developers and testers execute the next two sub activities defined in Figure 1. These sub activities are: 'Web Application Coding' (sub activity 1.4) and 'Refinement of Source-Codes and Functional Tests' (sub activity 1.5).

In sub activity 1.4, developers and testers generate source-codes both for the CHIs and functional tests with the support of Web-ACMDD. Source-codes are generated for specific technologies and defined in the 'Agile Modeling of Business for Web Application Activity'.

In sub activity 1.5, developers must refine the source code by complementing functionalities and applying the 'layout' and 'look & feel' from the CHIs. We stress at this point that the 'look & feel' feature is not yet part of the source-code generation of the AC-MDD Framework and for this reason it must be manually done by developers. In this sub activity, testers must execute the tests and check for the existence of flaws from generated source-codes.

There exists a flow from the 'Refinement of Source-Codes and Functional Tests' (sub activity 1.5) to every other previously done sub activity. This is justified by the fact that this flow aims to reduce the occurrence of flaws identified at the development stage and to correct them in the very same development sprint by returning to the activity responsible for correcting the flaws.

The last proposed sub-activity in the AC-MDD Framework, called 'Review - Validation of Produced Web Application Version' (sub activity 1.6), must be done by every part of the development team. This sub-activity consists of presenting the output of the last development sprint to the stakeholders, aiming to validate the requirements identified in the Sprint 0. If any refinement is needed, new user stories must be created and covered in the next Sprint.

As illustrated in Figure 1, visual constructs available in a new UML profile named Agile Modeling Language for Web Applications (Web-AML) were used throughout the whole AC-MDD Framework. To implement this framework, we conceived the Agile and Collaborative Model-Driven Development Method for Web Applications (WebAC-MDD Method), in order to make it operational. In other words, we followed the steps of the WebAC-MDD method by using different sets of visual constructs from the Web-AML to enable the execution of the AC-MDD Framework.

The web AC-MDD method and web-AML profile

The 12 steps of the WebAC-MDD Method are shown in Figure 2 using the UML Activity Diagram. In each step, we define how to properly go through it and try to document the most relevant elements of the AC-MDD Framework shown in Figure 1.
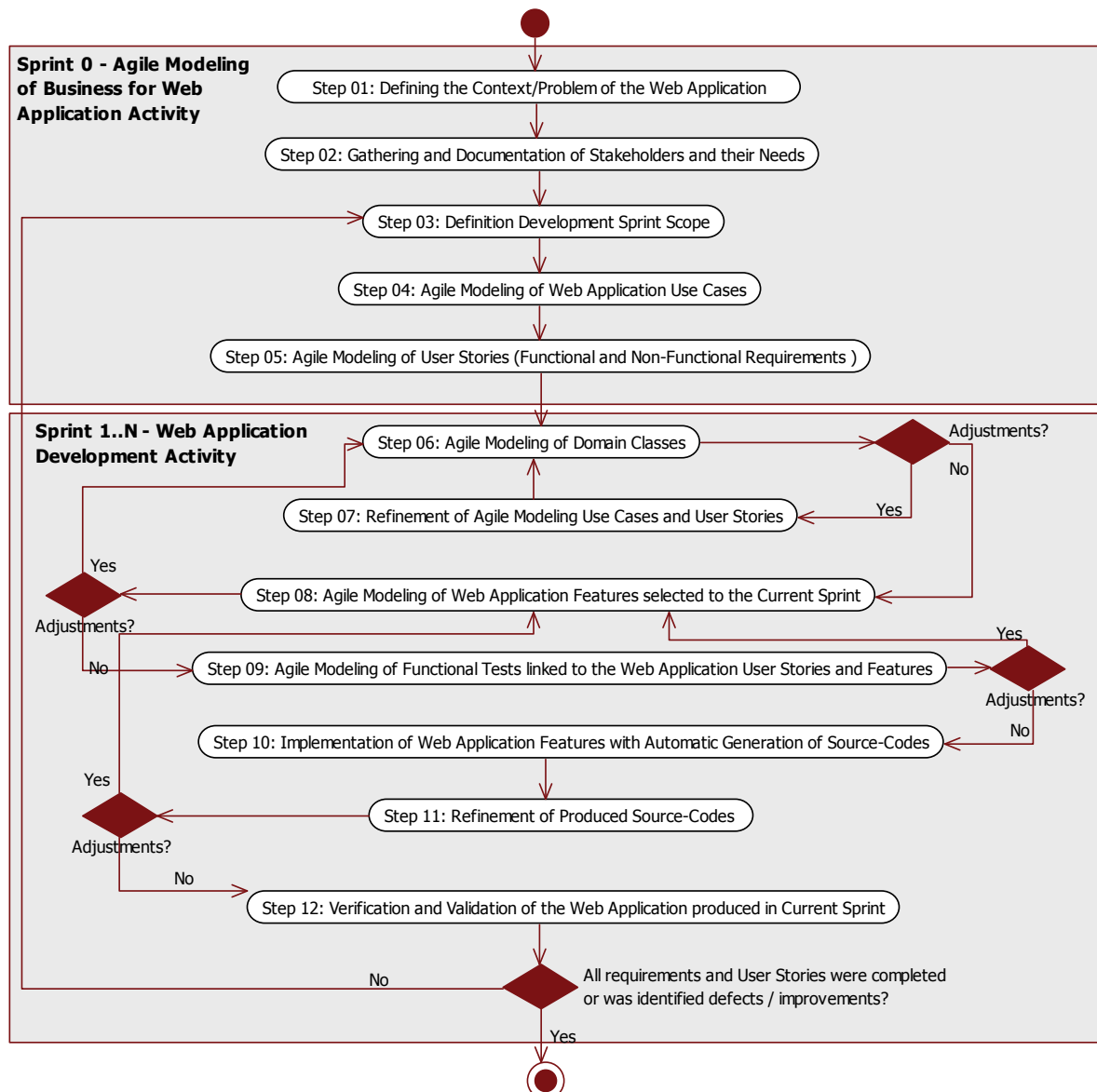
Figure 2. The Web Agile and Collaborative Model-Driven Development Method named WebAC-MDD.

As it can be seen in Figure 2, the first 5 steps proposed in the WebAC-MDD Method handle with the execution of Sprint 0 (Agile Modeling of Business for Web Application Activity) defined in the AC-MDD Framework (Figure 1). The following steps of the method (6 to 12) consider the application of the other Web Application Development Activities (Sprint 1 to N), until all user stories have been finished.

Since its conceptualization, we realized that the application of the WebAC-MDD method was not only a set of heuristics and/or good practices, but also a defined method characterized by a set of scientific and technological applications. Then, we identified 12 steps pertaining to the WebAC-MDD method. The goals of each of these steps are described in Table 1.

In each step of the WebAC-MDD Method, visual notations are also detailed. These notations are part of the Web-AML profile, which depends on the needs of WebAC-MDD to make it operational. First, we defined the visual constructs and stereotypes to document both predicted and unpredicted aspects of UML and WebML. Then, the visual constructs were consolidated, highlighting their conceptions in the Web-AML UML extension.

Due to scope, it is presented in Figure 3 and detailed in Table 2, the Web-AML profile metamodel for the elements that represent the CHIs functionalities (Step 8) of the web application.

The adopted case study: IF-Enchentes-SBV

To validate the AC-MDD Framework, the WebAC-MDD Method and the Web-AML Profile proposed, we conducted an experiment performed in a real project case study.

Table 1. Summary of each step proposed by the WebAC-MDD Method.

Step	Step goal
Step 1	Help the development team to understand the web application scope by identifying the problem and the application interactions with others elements. Elements of Web-AML profile: Context diagram
Step 2	Identify and document stakeholders and their macro-requisites. Elements of Web-AML profile: Requirements diagram
Step 3	Define the requirements' scope of a particular development sprint. Elements of Web-AML profile: Development sprint scope
Step 4	Model the use cases and the business rules, using the AM. Elements of Web-AML profile: Use case diagram
Step 5	Model in an agile manner the user stories to detail the requirements identified in the sprint, complementing use cases information. Elements of Web-AML profile: Development sprints' user stories
Step 6	Create the agile models of the web application domain classes, focusing on sprint's use cases and user stories. Elements of Web-AML profile: Domain classes
Step 7	Minimize the flaws found in the agile modes on the business level that may directly impact the development of sprint's functionalities. Elements of Web-AML profile: Adjustments in agile models of the development sprint
Step 8	Create agile models of the web application functionalities to be developed, focusing on the current development sprint, to represent the Computer Human Interfaces (CHIs) and the services proposed by the scenarios. Elements of Web-AML profile: Agile models of functionalities (CHIs)
Step 9	Create agile models of functional tests, linking them to user stories and functionalities. Elements of Web-AML profile: Functionality test cases
Step 10	Generate products from the agile models of value to stakeholders. Elements of Web-AML profile: Database, test, and functionalities source codes
Step 11	Check the source-code and make any necessary adjustments. Elements of Web-AML profile: Source-code refinement
Step 12	Check and validate functionalities of the current development sprint, trying to identify flaws and requirement conformity. Elements of Web-AML profile: Development Sprint delivery

Table 2. Description of the elements proposed in Step 08.

Elements	Description
General elements for organizational structure	
Page	Containers to organize information to end users, aggregating other modeling elements
Value Object	Information transmitted among pages or others modeling elements of the Web Application
Elements to handle with global variables	
Set Unit	Register global variables in browser's sessions to be used in others pages
Get Unit	Get global variables of the browser's sessions to be used in others pages
Elements to model Pages' Contents	
Data Unit	To present a set of attributes from a single entity instance
Entry Unit	To define a form, aiming to collect a set of input values entered by end users
Index Unit	To present a set of attributes from a set of entity's instance
Elements for access control of the Web Application	
Login Unit	To control end-user access in restricted pages
Logout Unit	Exit a web application
Elements to represent content management	
Create Unit	Insert a new instance of an entity into the database
Modify Unit	Update an existing instance of an entity in the database
Delete Unit	Delete an existing instance of an entity in the database
Connect Unit	Insert a new instance of relationship between two entities into the database
Disconnect Unit	Delete an existing instance of relationship between two entities from the database

The real project was called 'IF-Enchentes-SBV' – Development of a collaborative environment to control and alert about floods and meteorological conditions in São João da Boa Vista. This project is currently under development by students at the Federal Institute of Education, Science and Technology (IFSP) in São João da Boa Vista.

This project aims to help people of São João da Boa Vista, located in the interior of the State of São Paulo, Brazil, to deal with a real flood problem they have been facing in the last few years.

It is justified by the fact that in the last few years, São João da Boa Vista has been facing serious problems of floods and overflows in different parts of the town. To help inhabitants avoid material damages and life risks, we proposed the development of a collaborative computational environment (web application, Android app, and Data Acquisition Platform – DAP) to help them get real-time information, both from DAPs and from other inhabitants, about these climate issues.
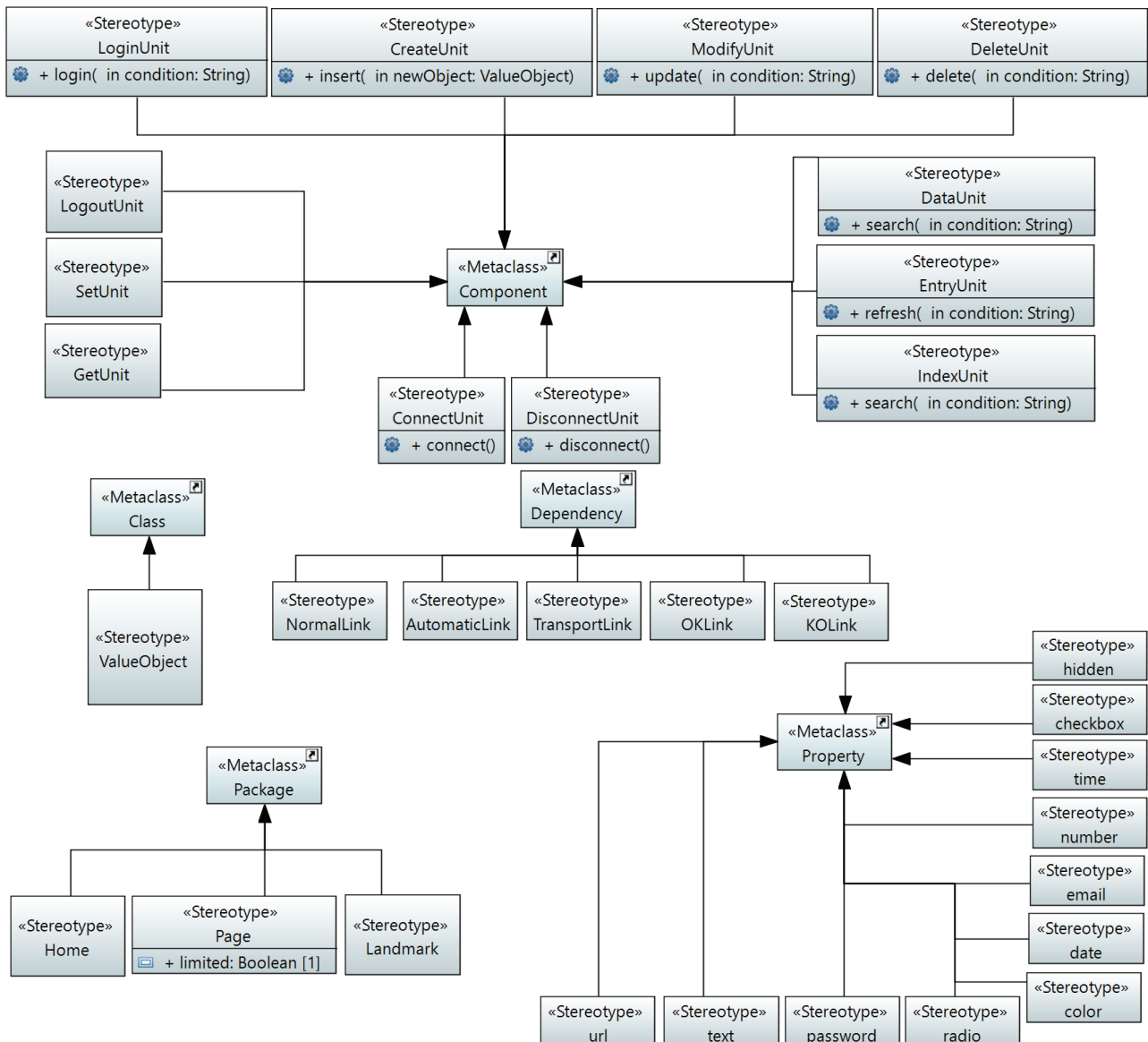
Figure 3. The Web-AML Profile for agile modeling of elements proposed in Step 08.

The idea of the project is to develop a web application and an Android app to allow people from São João da Boa Vista to see real-time information about floods in different parts of the town. Flood data will be measured by meteorological stations (DAPs) and their prototypes will be created during the project.

In this paper, we took the web application part of the collaborative computational environment of the IF-Enchentes-SBV project as the case study, due to research's scope presented with the AC-MDD Framework. In the next section, it is presented some results obtained with the application of AC-MDD Framework, the WebAC-MDD Method, and the Web-AML Profile in the developed case study.

The main results

The web application of the IF-Enchentes-SBV project was developed within five Sprints, focusing on the following similar functionalities: Sprint 01 - User/Administrator Management; Sprint 02 - DAPs Management; Sprint 03 - DAPs Measurements Management; Sprint 04 - Reports and Charts Management; and Sprint 05: DAPs Critical Alerts Management.

Due to scope reasons, in this paper it was chosen to report only the main results from Sprints 01 and 02 to develop the User/Administrator Management and Management of the Data Acquisition Platform (DAP).

In order to contextualize about Sprints 01 and 02, Figure 4 presents the Use Cases Diagram incrementally modeled, defining the main features contemplated in these Sprints. It was modeled after applying Steps 03 to 05. Notice there are highlighted links between use cases and user stories that detail about its developments.
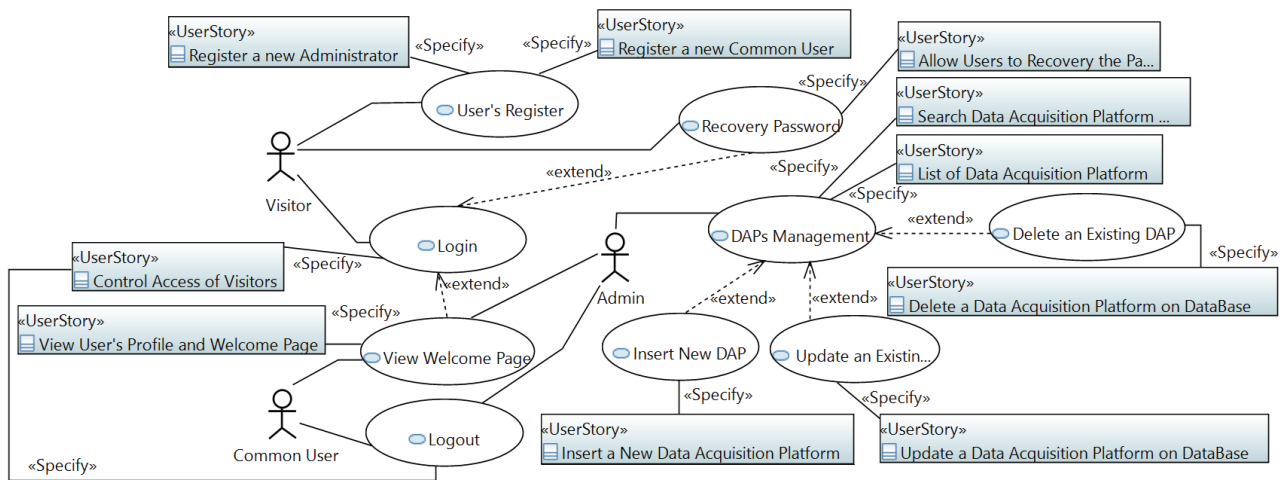
Figure 4. Use Cases Diagram of Sprints 01 and 02.

As shown in Figure 4, a visitor can register by itself in the Web Application and log in it. If the visitor has forgotten the password, it can be recovered with limited attempts. Once logged in, the user has one of the following profiles: Common User (one of the inhabitants of the São João da Boa Vista city) or System Administrator.

As Common User, Sprint 01 considers only the view of his/her profile page (welcome page) or exit of the system. On the other hand, as Administrator, the user can perform all DAPs' management and basic operations of CRUD (Create, Read, Update, or Delete) in the database (Sprint 02).

Due to limitations of the CASE tool we used, it is not possible to view the details of each User Stories defined in Figure 4. For this reason, to illustrate a User Story's definition, Table 3 lists the User Story 'Search Data Acquisition Platform by Users', proposed in Step 05 of the WebAC-MDD Method.

Step 08 of the WebAC-MDD Method is performed to produce agile models of the Web Application Features. Figure 5 presents the main results of the agile model for control access of visitors produced in Sprint 01.

Figure 5 highlights the components and parameters used to allow visitor to log in to the Web Application from a home page. If visitors get successfully logged in, they are redirected to the Welcome Page, where some personal information are shown to the user and they are registered in the session of the browser for future uses. Finally, from the Welcome Page, user can exit the system returning to the home page of the 'IF-Enchentes-SBV' Web Application.

Additionally, some agile models for the DAPs' Management produced in Sprint 02 are presented in Figure 6. These models were also created in Step 8 of the WebAC-MDD method and highlight the components and parameters used to build CRUD's functionalities previously mentioned. So, Figure 6 shows a main page for listing existing DAPs in the Database. From this page, System Administrator can update information from an existing DAP, delete an existing one, or insert a new one.

The agile modeling of functional tests of Sprint 01 produced two test cases after the execution of the WebAC-MDD Method. The result is shown in Figure 7.

Test cases modeled in Figure 7 aimed at verifying two questions: Test Login With Correct User - an existing user can successfully log in to the web application; and Test Login With User Not Found - a user not registered in the database does not have access to restricted pages.

Once the agile models of the functionalities defined in Sprints 01 and 02 are done, we execute Step 10 of the WebAC-MDD Method to generate source-code from these models.

Given the agile modeling in Figure 6, we execute Step 10 of the Web-ACMDD method, which is responsible for the implementation of the web application functionalities with automatic source-code generation. The CHIs representing the CRUD from the DCP Management are shown in Figure 8 and 9.

Discussion

This section presents some discussions about the developed case study to validate the AC-MDD Framework, the WebAC-MDD Method, and the Web-AML Profile. There is also a comparison between the

AC-MDD Framework and existing AMDD processes. The AC-MDD Framework adoption causes the development team deals more with agile modeling of web applications, rather than working directly with source-codes. It brought both positive and negative points that are discussed below.

A negative aspect is that as new agile models are proposed in the Web-AML Profile, there is a learning curve to make the best use of these models and represent the proposed solutions in a more appropriate way. In addition, the presence of new agile models may appear to be difficult to document Web applications, and in some cases making this documentation more complex.

Table 3. Definition of the User Story named 'Search Data Acquisition Platform by Users'.

User Story Search Data Acquisition Platform by Users	
Actor	System Administrator
Goal	Search for DAP already registered and installed in the city. The search can be done by one of the attributes: City, State or Description.
Reason	Update information from a specific DAP and keep the database always up-to-date, allowing Administrators keep track of DAPs' locations.
Most Important Qualities	Portability in browsers most used by the population.
Less Important Qualities	Ordering the list of PCDs presented, according to the user's choice.
Desirable Time	1 Month
Priority	5 (High)
Estimated Effort	4 hours
Sprint	Sprint 02
Team	Breno Lisi Romano
Comments	This user story can only be developed after the development of User Story named 'List of Data Acquisition Platform'.
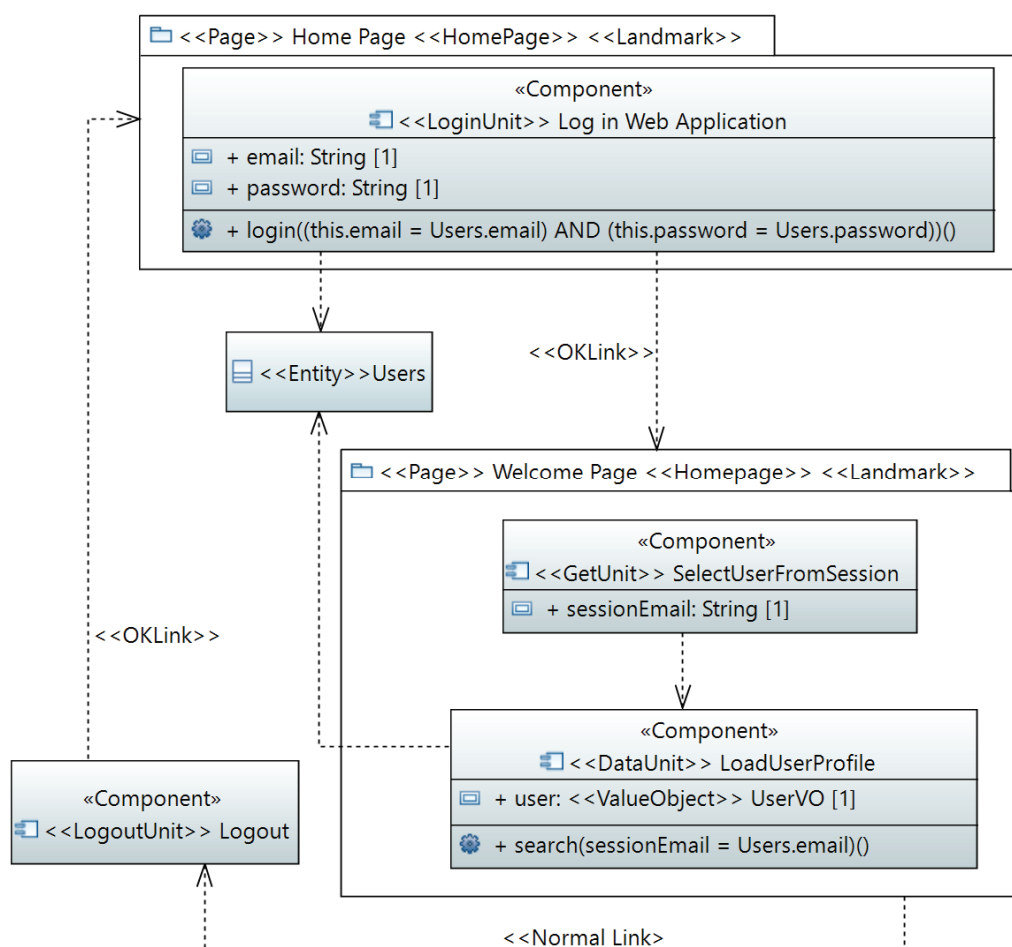
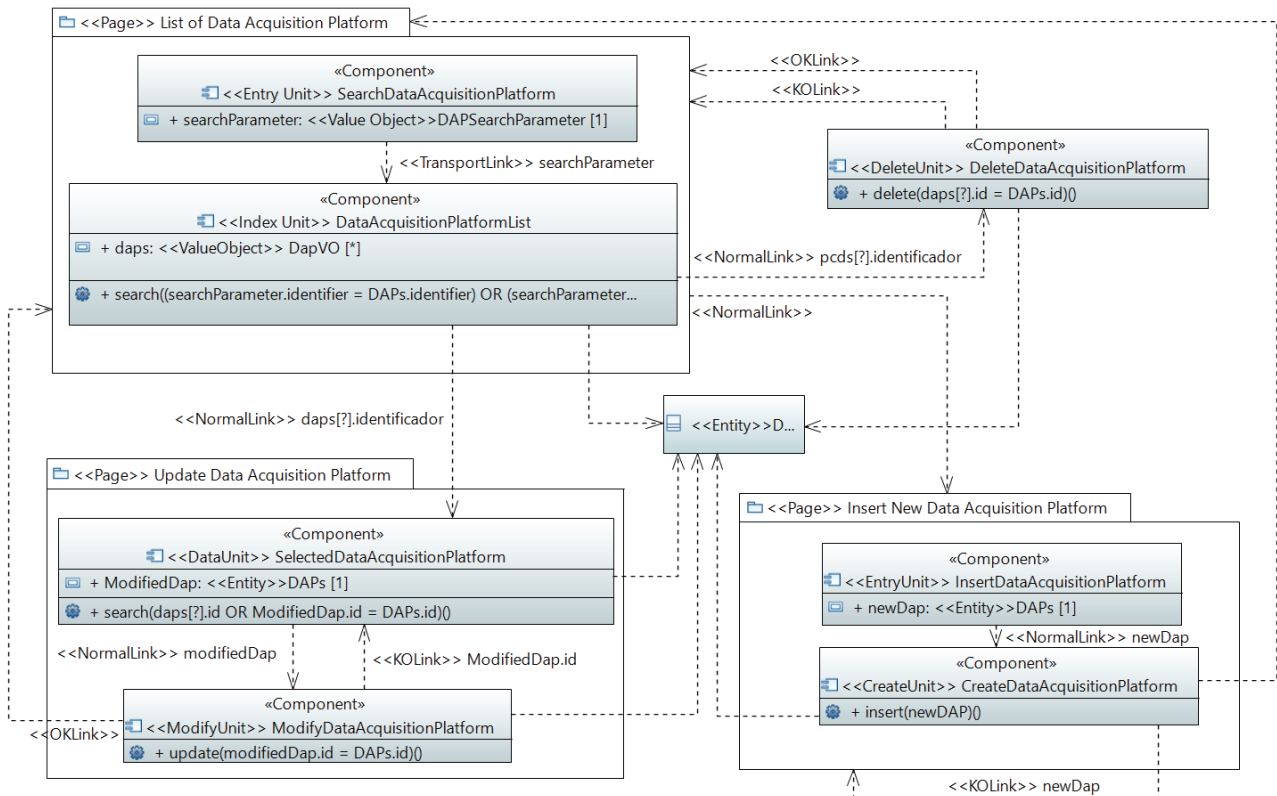**Figure 5.** The agile models of the Control Access of Visitors.

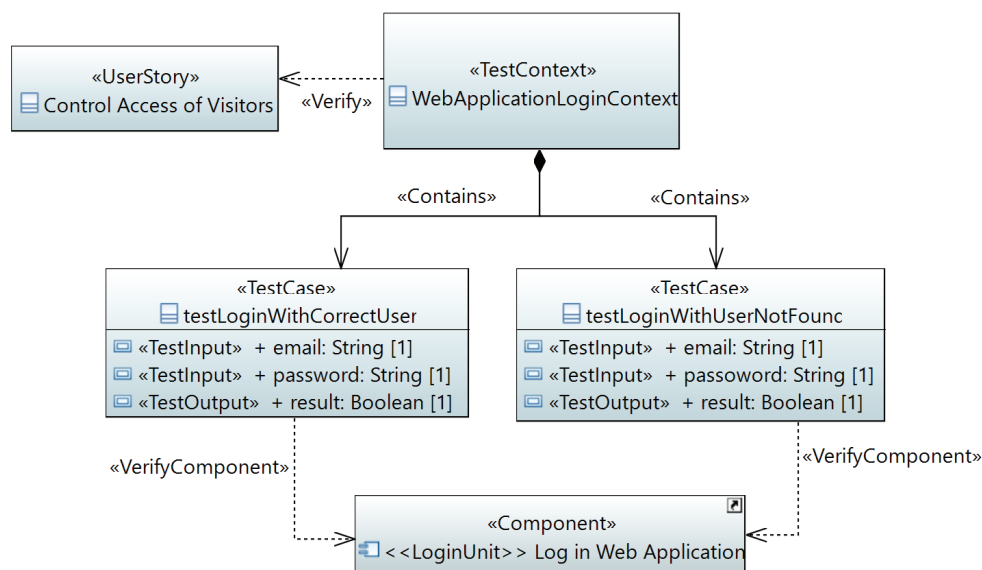**Figure 6.** The agile models of DAPs' Management.

Figure 7. The test cases for the Control Access of Visitors (Sprint 01).

However, as presented in the previous section, agile models proposed in the AC-MDD Framework always add value, producing both web applications' and testing's source-codes. Steps 01 to 05 of the WebAC-MDD Method are the only steps that propose documentation without impacting the generation of source-codes.

Steps 03 and 05 define planning issues for Web applications' development, proposing the definition of development's Sprints and their User Stories. Step 04 is a fundamental step, because it details use cases' scenarios that will be developed, without focusing on extensive documentations (adoption of AM's practices).

Therefore, Steps 01 and 02 are the only ones proposed to produce complementary documentations of Web applications. For this reason, if it is the interest of the development team, the execution of these two steps proposed in the WebAC-MDD Method can be optional, reducing documentations' efforts.

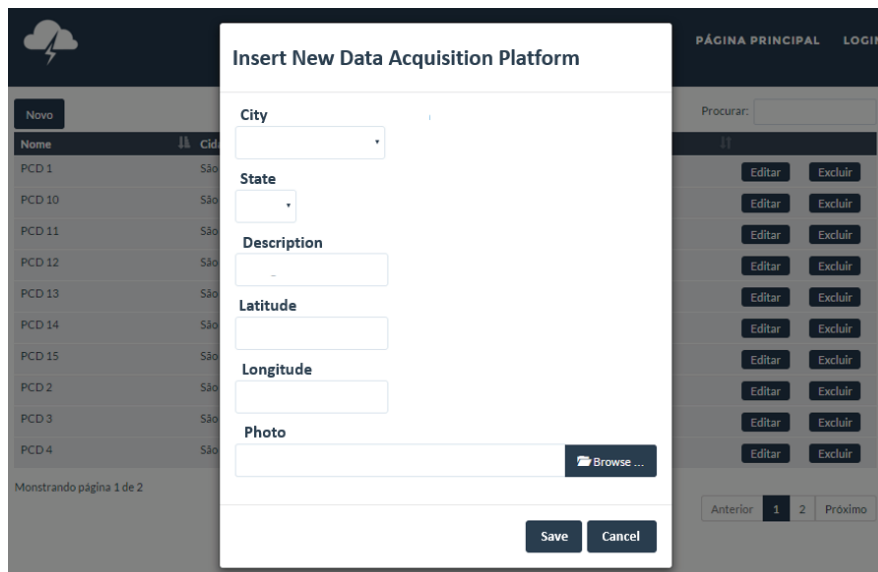**Figure 8.** The CHI to list the DAPs stored.

Figure 9. The CHI to insert a new DAP.

On the other hand, as the AC-MDD Framework proposes activities more related to agile modeling of Web applications, consequently, the concern about the lack of documentation in the adoption of agile methodologies, highlighted in the research conducted in 2017 by VersionOne (2017), is considered in the proposed framework.

Additionally, as agile models proposed in this research add value to source-codes generation, there is a considerable gain in productivity of Web applications' development.

For the developed case study in this paper, Figure 10 highlights a comparison between lines of code generated from the agile models produced in all Sprints of the 'IF-Enchentes-SBV' project, in relation to the lines of code that had a manually work of the developers, mainly for improvements regarding to 'look & feel' aspects.

As can be seen in Figure 10, the lines of code produced with AC-MDD Framework's adoption is larger than the lines of code improved by the developers, considering all Sprints proposed to the 'IF-Enchentes-SBV' project.

In order to develop the adopted case study with a total of 27,197 lines of code, it was necessary to define several agile models. Table 5 shows a summary of some elements modeled in this project using the AC-MDD Framework.

Table 4 makes clear that the 'IF-Enchentes-SBV' developed as a case study is a project with considerable complexity and effort to apply the AC-MDD Framework. In addition, the entire project was developed in a total of 8,640 hours by students of the IFSP, with part-time of work.

As detailed in Section II, as there are other agile processes, this section also presents a comparison between the AC-MDD Framework and these existing AMDD processes.

It is worth emphasizing that AC-MDD Framework was proposed to increase productivity in the development of Web Applications, generating source-codes from agile models. Its purpose was to reduce waste of resources on modeling and documentation stages of web applications, involving collaborative work.

Table 5 shows a summary of the existing AMDD Processes, comparing them with AC-MDD Framework. The existing AMDD processes are summarized and classified according to their method engineering approach in this table.

As proposed by Matinnejad (2011), MDD-based AMDD processes are originated from a model-driven method and agile principles are incorporated into them, while agile-based AMDD processes are derived from introducing MDD into an agile method as the skeleton process. Thus, as the AC-MDD Framework was proposed based on AMDD High-Level Lifecycle, its method engineering approach is agile-based.

Table 5 also presents the main purpose of each AMDD process. As it can be noticed, the AC-MDD Framework is the only one that focuses on Web Application development. Thus, the framework addresses the development of a new scope of applications that can adopt an AMDD process, considering specific characteristics of these applications.

Finally, the year of introduction of each process is listed in the last column of the table with its respective authors. Evidently, the concept of AMDD is still recent nd demands greater attention in research and practice.

To complement the comparison of existing AMDD processes, Table 6 was designed to highlight a comparison of the following criteria among these processes: Agility, MDD, and AMDD.

	Sprint 01	Sprint 02	Sprint 03	Sprint 04	Sprint 05	Total
Improved Lines of Code (Look & Feel)	1016	1581	1016	2669	2115	8397
Generated Lines of Code	5533	4750	2150	3499	2868	18800

Figure 10. Lines of code to the IF-Enchentes-SBV: Generated vs. Improved by Developers.

Table 4. Summary of some elements produced in the 'IF-Enchentes-SBV' project using AC-MDD Framework.

	Sprint #01	Sprint #02	Sprint #03	Sprint #04	Sprint #05	Total
Requirements	12	07	07	08	10	44
Use Cases	21	21	16	12	29	99
User Stories	31	33	26	24	36	150
Domain Classes	14	06	20	11	11	62
Test Cases	108	139	73	84	218	622
Produced CHIs	18	13	12	8	4	55

Table 5. Summary of Existing AMDD Processes.

AMDD Process	Approach	Main Objective	Proposed By / Year
AMDD High-Level Lifecycle	Agile-Based	Scaling agile development. Putting forward the notion of AMDD approach.	(Ambler, 2007)
Sage	MDD-Based	Applying agile approach to high assurance software. Supporting executable delivery from partial conflicting models.	(Kirby Jr, 2006)
MDD-SLAP	Agile-Based	Benefiting from both agile and MDD advantages in developing real-time telecommunication systems. Establishing a simple, yet fundamental correspondence between MDD activities and agile practices.	(Zhang & Patel, 2011)
AC-MDD Framework	Agile-Based	A framework for Web Applications to increase productivity by generating source-codes from agile models. It tackles waste reduction of resources on the modeling and documentation stages of web applications, involving collaborative work.	[This paper] (2017)

Table 6. Criterions' Evaluation of existing AMDD Processes.

Criterion / AMDD Process		AMDD High-Level Lifecycle	Sage	MDD-SLAP	AC-MDD Framework
Agility	Flexibility	✓	✓	✓	✓
	Rapidity	✗	✓	✓	✓
	Learning	✗	✗	✓	⚠
	Responsiveness	✗	✓	✓	✓
MDD	Cost Effectiveness	✓	✗	✗	✓
	Metadata Management	✓	✗	✗	✓
	Automatic Test	✓	✗	✓	✓
	Tool Support	✗	✓	✓	✓
	Problem Domain Analysis	✗	✓	✓	✓
	Verification / Validation	✗	✗	✗	✓
AMDD	Smoothness	✓	✗	✓	✓
	Coherency	✓	✗	✓	✓
	Simplicity	✓	✓	✓	✓
	Generality	✓	✗	✗	✗

✗ Not Supported: Criterion is not supported by the AMDD Process. ⚠ Partially Supported: Criterion is partially supported by the AMDD Process. ✓ Fully Supported: Criterion is completely supported by the AMDD Process.

For the preparation of Table 6, the research conducted by Matinnejad (2011) was used as the basis. In this research, the author defined each of the selected criteria and classified them in these three categories.

It was decided in this paper to define whether the selected criterion is included by the existing AMDD process, using three different ways: Not Supported, Partially Supported; or Fully Supported.

Notice, in Table 6, that the only criteria not completely supported by the AC-MDD Framework are: Learning and Generality. The Learning criterion is considered partially supported, because although the WebAC-MDD Method is proposed to apply the AC-MDD Framework, new agile models are defined in the Web-AML Profile. Thus, there is a learning curve with slow growth at the beginning so that modeling can be more appropriately performed.

However, the Generality criterion is defined as not supported, because the focus of the AC-MDD Framework is only Web applications. For this reason, it was not considering characteristics of applications of other contexts in its elaboration.

Conclusion

This paper presented the AC-MDD Framework aiming to increase productivity by generating source-codes from agile models. It tackles waste reduction of resources on the modeling and documentation stages of web applications, involving collaborative work among stakeholders.

To reach this goal, some visual constructs from a novel UML profile named Web-AML was conceptualized and implemented. To apply the AC-MDD Framework using these new visual constructions, a WebAC-MDD Method was defined providing all steps to transform agile models into Web applications source-codes.

A case study highlighted a step-by-step transformation process of agile models into source-codes to provide verification and validation of the AC-MDD Framework, the WebAC-MDD Method, and the Web-AML Profile. This was performed in a real project called IF-Enchentes-SBV. This project aimed to offer a collaborative environment to control and alert about floods and meteorological conditions in São João da Boa Vista, a city located in the State of São Paulo, Brazil.

As it was presented with the adoption of the AC-MDD Framework in the case study, a significant gain in productivity was achieved with the generation of 69% of lines of code in relation to project's total lines of code. In addition, the proposed framework is the only one that focuses on Web Application development, addressing the development of a new scope of applications, considering specific characteristics of these applications.

However, a negative aspect is that as new agile models were proposed in the Web-AML Profile to apply the AC-MDD Framework, there is a learning curve to make the best use of these models and represent the proposed solutions in a more appropriate way.

Considering a general analysis of Web Engineering approaches, it can be concluded that although the Web-AML Profile still has some limitations for modeling elements of Web applications, the profile presents itself as a good solution to deal with the development of these applications, offering a complete proposal of its documentations.

Since the Web-AML is a new UML profile, it can be used with any CASE tool, in order to allow the interoperability of models created with the WebAC-MDD Method. In this way, the data exchange between tools is facilitated. Moreover, both the WebAC-MDD Method and the Web-AML Profile can be used in a stand-alone manner.

It is worth mentioning that the transformations proposed in this paper to create the CHIs do not include look & feel features. We suggest using a style sheet to define the presentation of the CHIs and apply the effects defined in the modeled elements.

As future work, we aim to use the AC-MDD Framework in other real projects to receive feedback from the people involved in such projects.

Finally, it is important to notice that this paper shows that using the AC-MDD Framework is a viable alternative to increase the web application development productivity and decrease efforts put into modeling and documentation.

References

Ambler, S. W. (2007). *Agile model driven development (AMDD). Xootic Magazine.* Retrieved from http://agilemodeling.com/essays/amdd.htm

Ambler, S. W. (2014). *Agile modeling.* Retrieved from http://www.agilemodeling.com/

Ambler, S.W. (2008). Agile software development at scale. In B. Meyer, J. R. Nawrocki, & B. Walter (Eds.)*, Balancing Agility and Formalism in Software Engineering* (p. 1-12). Berlin, DE: Springer.

Frajtak, K., Bures, M., & Jelinek, I. (2015). Using the interaction flow modelling language for generation of automated front-end tests. *Federated Conference on Computer Science and Information Systems, 6*, 117-122. doi: 10.15439/2015F392

Huang, Y. C., & Chu, C. P. (2014). Developing web applications based on model driven architecture. *International Journal of Software Engineering and Knowledge Engineering, 24*(2), 163-182. doi: 10.1142/S0218194014500077

Kirby Jr, J. (2006). Model-driven agile development of reactive multi-agent systems. *Annual International Computer Software and Applications Conference (COMPSAC'06), 2*, 297-302. doi: 10.1109/COMPSAC.2006.144

Kroiss, C., & Koch, N. (2008). *UWE metamodel and profile: user guide and reference.* München, DE: Technical Report.

Liang, X., Marmaridis, I., & Ginige, A. (2007). Facilitating agile model driven development and end-user development for evolvingweb-basedworkflow applications. *IEEE International Conference on e-Business Engineering, 2007*, 231-238. doi: 10.1109/ICEBE.2007.37

Matinnejad, R. (2011). Agile model driven development: An intelligent compromise. *IEEE International Conference on Software Engineering Research, Management and Applications, 2011*, 197-202. doi: 10.1109/SERA.2011.17

Moreno, N., Fraternali, P., & Vallecillo, A. (2007). Webml modelling in uml. *IET Software, 1*(3), 67-80. doi: 10.1049/iet-sen:20060067

Pressman, R. S., & Maxim, B. (2014). *Software engineering: A practitioner's approach* (8th ed.). New York, NY: McGraw-Hill Science.

Qumer, A., & Henderson-Sellers, B. (2008). An evaluation of the degree of agility in six agile methods and its applicability for method engineering. *Information and Software Technology, 50*(4), 280-295. doi: 10.1016/j.infsof.2007.02.002

Romano, B. L., Silva, G. B. E., Cunha, A. M., & Mourão, W. I. (2010). Applying MDA development approach to a hydrological project. *IEEE Information Technology: New Generations (ITNG), 2010,* 1127-1132. doi: 10.1109/ITNG.2010.121

Rumpe, B. (2014). Agile test-based modeling. *International Conference on Software Engineering Research and Practice, 1*, 10-15.

Schauerhuber, A., Wimmer, M., Schwinger, W., Kapsammer, E., & Retschitzegger, W. (2007). Aspect-oriented modeling of ubiquitous web applications: The aspectwebml approach. *IEEE International Conference and Workshops on the IEEE Engineering of Computer-Based Systems, 2007*, 569-576. doi: 10.1109/ECBS.2007.20

VersionOne. (2017). *Inc. 11th annual State of Agile Report*. Retrieved from http://stateofagile.versionone.com

Wakil, K., Jawawi, D. N., & Safi, A. (2015). A comparison of navigation model between uwe and webml: Homepage development case study. *International Journal of Information and Education Technology, 5*(9), 650-655. doi: 10.7763/IJIET.2015.V5.585

Zhang, Y., & Patel, S. (2011). Agile model-driven development in practice. *IEEE software, IEEE Computer Society, 28*(2), 84-91. doi: 10.1109/MS.2010.85