Using IFML for user interface modeling: an empirical study

Randerson Queiroz, Tayana Conte
Computer Institute - IComp
Federal University of Amazonas - UFAM
Manaus, AM
(rsq, tayana)@icomp.ufam.edu.br

Anna Beatriz Marques
Federal University of Ceará - UFC
Russas, CE
beatriz.marques@ufc.br

***Abstract* — Front-end interface and user-system interaction are factors that must be carefully considered in software development due to their influence in quality of use. On some occasions, it is the first concern addressed by developers, as it comes naturally from the requirements analysis performed with stakeholders. IFML is a standard language of OMG that supports the abstract description of these front-end interfaces, for software applications on different devices. IFML has been used in the context of Model-Driven Development (MDD) and Model-Driven Architecture (MDA) to describe the elements and behavior of interfaces, aiming to generate code for those interfaces. However, it is necessary to investigate the use of IFML in traditional software development, in order to better understand how it is used for modeling front-end interfaces. This article presents an empirical study that aimed to verify the quality of IFML models created based on a subset of requirements of software two web applications. The quality was defined in terms of models' correctness and completeness. The results showed that the correctness of the models was low, varying from 51% to 55%, while the completeness varied from 66% to 69%. In order to better understand the results, we analyzed syntactic and semantic defects found.**

Keywords-component: IFML, User Interface, Software development, Empirical Study.

I. Introduction

The Unified Modeling Language (UML) is widely used to model the system in different stages of traditional software development [6]. However, UML does not present a specific model to describe specifications of front-end interface and user interaction through this interface [9]. To cover this gap, the OMG (Object Management Group) proposed the adoption of Interaction Flow Modeling Language (IFML) [11]. IFML supports the abstract description of front-ends for devices such as computers, laptops, mobile phones and tablets. The objective of IFML is to express the content of these front-end interfaces and the data flows between the front-end components of the application [3].

The IFML uses a single diagram, in which developers can specify the user interface organization, the content displayed for the users and the effect of interface events produced by user interaction or by system notifications [1]. Since IFML is an extension of UML, the artifacts generated by UML notation are usually used as the basis for modeling with IFML [1][11][5]. IFML has often been used in the context of Model Driven Development (MDD) Model Driven Architecture (MDA) [10] to describe the elements and behavior of front-end interfaces, aiming to generate codes of these interfaces [1] [2][8]. However, the concern with the quality of the user interface is not present only in MDD and MDA development contexts. Can the user interface be modeled in a complete way using IFML in traditional software development? What is the correctness of the IFML diagrams created to represent the user interface?

To answer these questions, we conducted an empirical study in which graduate students (with experience in software industry) modeled the front-end interface using IFML, based on requirements of two web applications. The study aimed to verify whether the subjects can model using IFML correctly and completely in the traditional development context (not MDD or MDA). In order to better understand the results, we analyzed the syntactic and semantic defects of the models.

In order to analyze the completeness of a IFML model, we verified whether the elements used by the subjects were sufficient to represent the system requirements. In the analysis of the correctness of a IFML model, we verified whether the elements used by the subjects to represent the requirements were used correctly according to IFML syntax. It is important to conduct empirical studies in order to investigate the models and languages suitable for supporting software development teams in UI design.

The remainder of this paper is organized as follows: Section II presents the basic elements of IFML. Section III shows how we planned and executed the empirical study. Section IV shows the analysis of the results. Finally, Section V presents a discussion and final considerations.

II. Interaction Flow Modeling Language (IFML)

In this section we present a more detailed view of IFML and its elements. For a better understanding, we present a simple example of an IFML diagram.

The Interaction Flow Modeling Language (IFML) is a platform-independent model (PIM) used to express interaction design decisions regardless of the deployment platform [8]. Brambilla et al. [4] claim that IFML is designed to express the content, the user interaction and the control behavior of front-end software applications. Figure 1 shows the basic elements of IFML.

DOI reference number:10.18293/SEKE2018-103

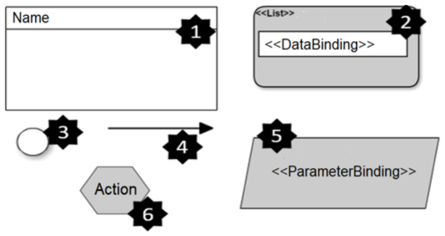

Figure 1. Basic Elements of IFML.

The basic elements of IFML are described below: 1) *ViewContainer* is an interface element that comprises elements displaying content and supporting the interaction and/or other *ViewContainers*; 2) *ViewComponent* is an interface element that displays content, i.e., content and data entry elements contained in *ViewContainers*; 3) *Event* is an occurrence that affects the state of the application. Events can be produced by user interaction, by the application or by an external system; 4) *Navigation Flow* is an update of the interface elements in view or triggering of an action caused by the occurrence of an event. Data may be associated with the flow through *parameter bindings*; 5) *Parameter Binding* is a specification in which an input parameter of a source is associated with an output parameter of a target; 6) *Action* is a piece of business logic triggered by an *event*; it can be server-side (default) or client-side, denoted as (Client).

Figure 2 shows an example of an IFML diagram, describing a user interface where the user can search for a product by entering some search criteria in the Product Search form. The model consists of a Product *view container* (depicting a screen or Web page) that contains two *view components* (visual widgets placed on the screen), i.e., the Product Search form, where the user can enter the search criteria, and the Search Result list, which displays the search results. In addition, a product exclusion *action* can be triggered when the user selects the Exclusion Event associated with the Search Result.

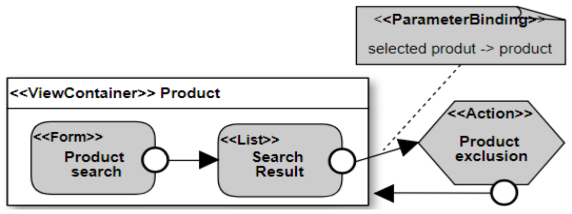

Figure 2. Example of IFML diagram.

III. EMPIRICAL STUDY

In this section we present the empirical study, the details of the study planning and the execution based on [13]. The artifacts used in the empirical study are available in a technical report [12].

A. Study Planning

The empirical study aimed to analyze the use of IFML in the modeling of interfaces in order to analyze the quality of the models in terms of correctness and completeness. Based on the we defined the following research questions: Can the user interface be modeled in a complete way using IFML in traditional software development? What is the correctness of the IFML diagrams created to represent the user interface?

We defined the necessary resources for its execution during the planning of the study, as detailed below:

1) Context: we carried out the study in academic context with graduate students.

2) Subjects: 16 graduate students participated in the study. All subjects had experience in software industry. However, they had not previously used IFML. We divided the subjects into two groups for modeling different scenarios.

3) Artifacts used: we prepared a consent form, in which the subjects could agree or not agree to make their data available for analysis in this research. In order to assist the subjects during modeling, we developed a guide of IFML elements.

4) Scenarios: the subjects used functional requirements of a system as the basis for modeling the interface. The requirements were described as scenarios. The scenario that group A received described a system of an airline company, which could be used to track flights to its destination. The scenario that group B received described a website that provided tips of restaurants by area of the city. Both scenarios had the same number of requirements (four requirements).

a) Group A Scenario: the scenario of group A contained the following requirements: 1) to access the system with login and password; 2) to track previously registered flights; 3) to register flights to be tracked; and 4) to configure notifications with flight route updates.

b) Group B Scenario: this scenario contained the following requirements: 1) to search for restaurant per area; 2) to bookmark a chosen restaurant as favorite; 3) to view tips about the park nest to a favorite restaurant; and 4) to confirm a restaurant booking.

B. Study Execution

The study was conducted in a single day lasting 2 hours and 30 minutes. We divided the activity into training, preparation of the activity with receipt of the scenarios and modeling with IFML. The study started with the training. In the training, the subjects received training on all the elements of IFML. The training contained examples and two practical exercises and lasted about 1 hour and 30 minutes. After the training, the subjects received and signed the consent form. The subjects were randomly organized in two groups (A and B). Each subject received a requirements scenario, according to the group he was assigned to.

After receiving the scenarios, the subjects started the modeling step using IFML. After concluding the modeling task, we carried out a discussion with the subjects about the activity they developed. In the discussion, each subject talked a little about their perceptions of the IFML. The discussion was recorded via audio and video. Thus, the audio was transcribed and analyzed.

IV. DATA ANALYSIS

This section presents the analysis of the results, through which we aimed to identify the completeness and correctness of the modeling. We also performed an analysis in order to identify the syntactic and semantic defects of each model. The subjects' perception about IFML was also analyzed. To perform this analysis, a researcher inspected the models developed by both groups and a second researcher validated the identified defects. Data from subject S1 were excluded from the analysis because he did not participate in all activity.

A. *Completeness and Correctness*

To achieve the completeness and correctness of each model, we elaborated oracles in order to support the analysis. The oracle corresponds to a possible solution for the scenarios modeling and defines a set of IFML elements that can be used in the solution. In the analysis, we used the oracles as basis for analyzing the elements used by the subjects, the elements not used and the elements that they could use in the model. Each oracle has specific requirements for each scenario. For each requirement, we listed which elements were necessary to model the front-end related to the requirement described. Table I shows part of the oracle, with the required elements for the front-end related to the modeling of the *Access System* requirement.

TABLE I. ORACLE GROUP A

Group A – Scenario A
Access System
ViewContainer
ViewComponent Form
Submit event
Type of Data
Action
Parameter Binding

In order to define the completeness of each model, we proceeded with the sum of the number of elements used in the requirements divided by the number of elements necessary to represent the requirements according to the oracle. To obtain the correctness of each model, we also performed a calculation of the number of elements correctly used in the requirements divided by the number of elements defined in the oracle. Table II shows the mean of completeness and correctness of each subject and its respective group.

TABLE II. COMPLETENESS AND CORRECTNESS RESULTS.

Group A									
	S3	**S5**	**S7**	**S8**	**S10**	**S11**	**S13**	**S15**	Mean
Comple.	81%	67%	46%	86%	31%	82%	70%	89%	**69%**
Correct.	57%	52%	26%	74%	30%	70%	52%	83%	**55%**
Group B									
	S2	**S4**	**S6**	**S9**	**S12**	**S14**	**S16**	**S17**	Mean
Comple.	50%	55%	86%	90%	57%	72%	50%	70%	**66%**
Correct.	41%	23%	64%	73%	27%	68%	45%	68%	**51%**

The mean for completeness of the diagrams created by group A and group B were close to 69% and 66%, respectively. This shows that the subjects had difficulty to completely model the requirements. The *ViewComponen*t and *Event* elements were not used. In addition, the mean for correctness of the diagrams created by groups A and B were 55% and 51%, respectively. This result shows that even though the elements have been used, they were used incorrectly, thus decreasing the quality of the diagram created.

Since the subjects from group A and B used different scenarios, the results could have been influenced by the difference between these scenarios. The level of difficulty for modeling a requirement of one scenario could be greater than the requirement of the other scenario. In order to verify whether the scenarios had influenced the results, we applied a statistical hypothesis test. Table III shows the null and alternative hypotheses. The null hypotheses states that: “H_{01} - There is no difference in terms of completeness in modeling with IFML based on scenario A or B”, and “H_{02} - There is no difference in terms of correctness in modeling with IFML based on scenario A or B”.

TABLE III. NULL AND ALTERNATIVE HYPOTHESES

Null Hypotheses
H_{01} – There is no difference in terms of completeness in modeling with IFML based on scenario A or B
H_{02} - There is no difference in terms of correctness in modeling with IFML based on scenario A or B
Alternative Hypotheses
H_{A1} – There is a difference in terms of completeness in modeling based on Scenario A in relation to Scenario B
H_{A2} – There is a difference in terms of correctness in the modeling based on Scenario A in relation to Scenario B

We used the statistical Mann-Whitney non-parametric method. We used $\alpha = 0.05$ due to the sample size [5]. To perform the tests, we used the SPSS tool v20.0.0. The obtained results support the null hypotheses H_{01} and H_{02}, indicating that there is no significant difference in the completeness indicator ($p = 0.878$) nor in the correctness indicator ($p = 0.574$) when modeling using IFML with scenario A or B. Therefore, the fact that a group modeled using scenario A or B did not influence in the way the subjects modeled. It means that the scenarios did not influence the results of completeness and correctness, not affecting the results reliability.

B. *Syntactic and semantic defects*

We decided to classify the defects in syntactic or semantic for a better understanding about the defects and their impact. We also explored the possible difficulties in using the elements to model in a correct and understandable way. The concepts of syntactic and semantic properties have been adapted to the context of this study [8]. The definitions we used are: *Syntactic*, defect characterized by the incorrect use of IFML elements; *Semantic*, defect characterized by the incorrect modeling of the problem domain. Figure 3 shows the total number of syntactic and semantic defects, distributed into each group.

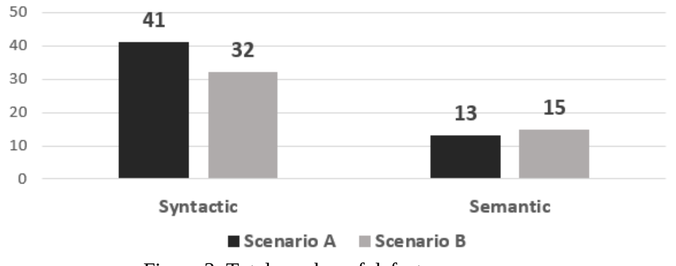

Figure 3. Total number of defect occurrences.

We identified 101 occurrences of defects, being 73 syntactic and 28 semantic, considering all the diagrams created by the subjects. When we did not consider repeated defects, we obtained a list of 24 unique defects. Figure 3 shows that we identified 41 occurrences of syntactic defects and 13 occurrences of semantic defects in diagrams created based on "scenario A". On the other hand, we identified 32 occurrences of syntactic defects and 15 of semantic defects in diagrams modeled based on "scenario B". For a better view, we listed the most common defects in Table IV, mentioning the type and number of occurrences for each defect. The list shows only the defects that have been repeated more than once. The other defects that occurred only once are not presented in the list.

TABLE IV. MOST COMMON DEFECTS

Defect	**Type of defect**	**Number of Occurrences**
Does not specify the data type	Syntactic	27
Uses the wrong *event*	Syntactic	15
Does not inform the data that is being passed	Syntactic	14
Uses the wrong *ViewComponent*	Syntactic	6
Uses Default *ViewContainer* outside of *XOR ViewContainer*	Syntactic	5
Did not model the interaction of notification settings	Semantic	5
Did not model the actions cancel and confirm	Semantic	4
Did not model the requirement confirmation View	Semantic	4
The *action* to choose the View Booking feature was not modeled	Semantic	3

We also listed the number of defects per element, but this was only possible for syntactic defects. Making up a list for semantic defects was not possible because 16 out of the total 28 semantic defects are related to the complete omission of a requirement. These semantic defects are related to the omission of all the elements that subjects could apply in the modeling, so it is not possible to make the exact count of the elements involved in each defect. Figure 4 shows all the syntactic defects found in both scenarios, considering each possible element involved in the defect.

1) *Syntactic defects*

Figure 4 shows the number of occurrences of defects in each element of the IFML. The major number of defects is related to the *Parameter Binding* component, with 16 defects. Although the subjects used this element correctly in order to inform the data related to the interactions modeled, they did not correctly apply the standard specified by the language. There were also 14 occurrences of defects involving the *select event*. The subjects preferred to only indicate that there was an event, without specifying the element type. There was a large number of defect involving the *ViewComponent List* and *ViewComponent Details*. We identified 34 defects in total. Some of these occurrences are related to the misuse of the *ViewComponent* type used to model the requirement. This may indicate that the subjects did not understand the difference among the types of view components.

As shown in Table IV, the most frequently defect "Does not specify the data type" is related to the *ViewComponent*, in which the subjects did not demonstrate the data of the components by following the standard proposed by the language. The definition of the type of data is typically represented in UML diagrams, such as the class diagram. However, since the only artifact elaborated in this study was the IFML diagram, the omission of this type of information may reduce the understanding of the content of the interface. The "Uses the wrong *event*" defect is directly related to changes in the state of the modeled system. These changes are initiated through the *events* and the subjects did not use the correct *events* for each requirement. In some cases, the subjects did not specify the type of *event*. This shows that they did not understand the difference between *event* types.

The defect "Does not inform the data that is being passed" is related to non-compliance with the standard language in the use of the *Parameter Binding* element. In the particular context of this study, this defect did not impair the comprehension of these data. On the other hand, this defect may be harmful in systems where the data stream is essential for the full operation of the system itself. The defect "Uses Default *ViewContainer* outside of *XOR ViewContainer*" refers to the non-organization of the *containers* in the models. This shows that the subjects who modeled with this defect had difficulty in understanding the organization rule of the *containers*. Considering systems with a large number of tabs and navigations among windows, this defect would be potentially harmful.

2) *Semantic defects*

Among the 28 semantic defects we identified in the diagrams created, 16 of them are complete omissions of a requirement or part of a requirement. We noted that these defects are related to the omission of the elements necessary to adequately model the requirement. The reasons for this phenomenon of omission may be the misunderstanding of the elements involved or the tiresome that the subjects may have felt during the final part of the modeling. However, the semantic defects of omission are related to the requirements that can be considered as the most difficult ones for modeling. For example, in Group A, the semantic defects of omission are related to the requirement "to configure notifications with flight route updates". This

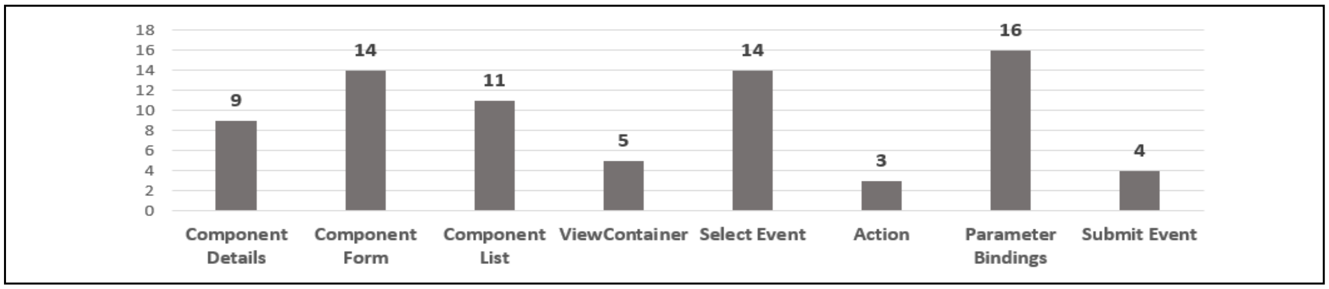

Figure 4. Number of syntactic defects per element

requirement requires a wider combination of elements to model the interaction that the user needed to complete their goal. In Group B, the requirement "to confirm a restaurant booking" requires that the subject models the system feedback for the user. For this requirement, we identified the major number of semantic defects.

The *action* element was related to all semantic defects, so there is a possibility that the major cause of this defect is the misunderstanding about the way that elements should be used. The quantitative data related to semantic defects indicates that the subjects found to be more difficult to use correctly the action element. Other semantic defects were repeated only once, e.g. the defect "Wrong Organization of *Containers*" and "Wrong Flow Sequence between *containers*". Both of them are semantic defects and impair the understanding of the model as a whole.

3) *Perception of the subjects*

The results of the correctness indicated a high number of occurrences of syntactic defects, pointed out a subjects' difficulty in correctly using some of the elements of IFML. Some subjects spontaneously commented something about this difficulty, further reinforcing the results of the study.

Subject S4, for example, reported that "*it is very challenging in the form of representing the screens, even getting tiring because of its many containers and types*". Subject S15 also reported the same difficulty "*it is difficult to use [the IFML] due to the numerous containers, it ends up leaving the diagram a little messy, making it difficult to visualize and have an idea of the requirements which are being put there*".

We observed that some subjects considered difficult the IFML elements with similar features. For example, elements like *ViewComponen*t and E*ven*t have several types to be used in different situations in modeling. That difficulty was also reported by subject S3 "*it is difficult to use because it may have many similar components, making it a little confusing when it comes to choosing. There are too many elements, it's very confusing when it comes to doing it*".

V. DISCUSSION AND FINAL CONSIDERATIONS

This study aimed to verify how graduate students model the user interface using IFML. The overall results of this study showed that the subjects had difficulties in modeling correctly the front-end based on a scenario describing a set of requirements. Furthermore, the elements of IFML were misused. Syntactic defects showed that the subjects had major difficulties in using the *events* and *view component* elements correctly. Regarding semantic defects, 16 out of the 28 defects were of total omissions of the requirements, which indicate models that do not specify all the requirements described in the scenarios.

In the context of this study, the subjects were able to model the requirements contained in the scenario with a completeness of 69% and 66% (Groups A and B respectively). The correctness of the models was even lower, with a mean of 55% and 51% respectively. The low number of the correctness can be related to the difficulties that the subjects have faced in using the elements of IFML.

With the results of this study, we expect that this research provides a better direction for professionals interested in using IFML in the user interface design. The results explore how the IFML diagram can be used in the interface design and possible difficulties the professionals can face when using some IFML elements. It is necessary to investigate the use of IFML in different contexts, with subjects from different levels of experience. The results of this research show that it is possible to comprehensively model the front-end interface of a web application using the IFML language. However, some difficulties regarding the elements of the IFML language can affects the correctness of the front-end interfaces.

Finally, as the study was applied in a small sample in an academic environment, it should be replicated with a more representative and heterogeneous sample.

ACKNOWLEDGMENTS

We would like to thank the financial support granted by UFAM, CNPq through processes numbers 423149/2016-4 and 311494/2017-0, and CAPES through process number 175956/2013.

REFERENCES

[1] C. Bernaschina, S. Comai and P. Fraternali, "Formal semantics of OMG's Interaction Flow Modeling Language (IFML) for mobile and rich-client application model driven development", Journal of Systems and Software, 137, 239-260, 2018.

[2] C. Bernaschina, S. Comai and P. Fraternali, "IFMLEdit. org: model driven rapid prototyping of mobile apps", In Proceedings of the 4th International Conference on Mobile Software Engineering and Systems p. 207-208, IEEE Press, 2017.

[3] M. Brambilla and P. Fraternali, "Interaction flow modeling language: Model-driven UI engineering of web and mobile apps with IFML", Morgan Kaufmann, 2014.

[4] M. Brambilla, E. Umuhoza and R. Acerbis, "Model-driven development of user interfaces for IoT systems via domain-specific components and patterns", Journal of Internet Services and Applications, 8(1), 14, 2017.

[5] T. Dyba, V. Kampenes and D. Sjoberg, "A systematic review of statistical power in software engineering experiments", Information and Software Technology, Volume 48, Issue 8, 2006.

[6] K. Frajták, M. Bureš and I. Jelínek, "Transformation of IFML schemas to automated tests", In Proceedings of the 2015 Conference on research in adaptive and converggent systems p. 509-511, ACM. 2015.

[7] P. Kamthan, "A framework for understanding and addressing the semiotic quality of use case models" Model-driven software development: Integrating quality assurance. Hershey, PA: IGI Global, 2008

[8] 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 p. 83-88, IEEE, 2016.

[9] N. Moreno, P. Fraternali and A. Vallecillo, "WebML modelling in UML", IET software, 1(3), p. 67-80,2007

[10] J. R. P. Moreira and R. S. P. Maciel, "Towards a Models Traceability and Synchronization Approach of an Enterprise Architecture" In The 29th International Conference on Software Engineering & Knowledge Engineering (SEKE), 2017.

[11] OMG, 2015. Interaction flow modeling language (IFML), version 1.0. http://www. omg.org/spec/IFML/1.0/ .

[12] R. Queiroz, A. Marques and T. Conte, "USES Technical Report TR-USES-2018-005. Using IFML for user interface modeling". Technical Report of Usability and Software Engineering Group (USES), 2018. Available in http://uses.icomp.ufam.edu.br/relatorios-tecnicos/

[13] C. Wohlin, P. Runeson, and M. Höst, "Experimentation in Software Engineering: An Introduction, Kluwer International Series in Software Engineering, 2000.