Producing Graphical User Interface from Activity Diagrams

Ebitisam K. Elberkawi, Mohamed M. Elammari

***Abstract*—**Graphical User Interface (GUI) is essential to programming, as is any other characteristic or feature, due to the fact that GUI components provide the fundamental interaction between the user and the program. Thus, we must give more interest to GUI during building and development of systems. Also, we must give a greater attention to the user who is the basic corner in the dealing with the GUI. This paper introduces an approach for designing GUI from one of the models of business workflows which describe the workflow behavior of a system, specifically through Activity Diagrams (AD).

***Keywords*—**Activity Diagram, Graphical User Interface.

I. Introduction

BOTH GUI Designers and Programmers must achieve a prominent level of coordination in the design of GUI. On other side, analysis and design phases concern much of the system in several aspects and GUI must be one of these aspects. In addition, we have to activate the user factor in construction and design of the GUI.

All of these motivations incited us to find an approach to derive a GUI, based on its design in the primary phases of system development and supports the role of the user in GUI development, hence, increasing the chance of cooperation between the GUI Designer and the programmer.

Accordingly, the goals of this work are:

1. Building a GUI by deriving from first phases of the system analysis and design, which makes the GUI being built upon strong bases by one of the early phases of the system development.
2. The early design of GUI provides a greater opportunity to deal with the user by presenting the GUI in advance and to know the user views, particularly to correct errors, if any, and make the required adjustments.
3. Providing a step forward for the process of building GUI from the Activity Diagram (AD) and considering it as one of the fields of system analysis and design.
4. Building a good GUI by greater support provided from collaboration between GUI Designer and programmer.

The approach presents in this paper depends on one of models of business workflows and here we mean AD and the approach arranged in two phases: first phase **(Documentation)** produces Document Sheets (DSs) which contain all details of designing the GUI, and second phase **(GUI Structure)** produces the structure of GUI. These phases are executed in sequence where the result of first phase is the basis of the starting point of the second phase.

The rest of the paper is organized as follows: Section II gives description of some relevant literature. Section III represents to the steps of whole work which is divided to two kind of steps, preparatory steps and fundamental steps. Section IV describes our approach and its phases in more details and explanation. Practical application to show how we can execute the phases of the approach through a case study in Section V. Finally, Section VI discusses the conclusion of this work.

E. K. Elberkawi is a lecturer at the Information System department, Faculty of Information Technology, University of Benghazi, Benghazi, Libya (email: Bit2gar@gmail.com).

M. M. Elammari is a professor and the head of Software Engineering department, Faculty of Information Technology, University of Benghazi, Benghazi, Libya (email: elammari.m@gmail.com).

II. Literature Review

Elkoutbi and Keller [3] suggested an approach consists of four steps starting with the Use Case Diagram (UCD) of the system, then for each use case in the diagram, scenarios are gained in the form of Unified Modeling Language (UML) sequence diagrams.

In the next step, the UCD and sequence diagrams are transformed to Colored Petri Nets (CPNs) and then from some processes on these CPNs, a global CPN is formed where it acquires the behavior of the entire system from which the UI prototype of the system is generated in the final stage.

Pinheiro da Silva and Paton [6] discussed modelling of User Interface (UI) using UML. They presented a Library System case study and how this case study identifies some aspects that cannot be modelled using UML notation. These modelling problems determine some weaknesses of UML for modelling UIs, and on the other hand, a set of UML constructors that may be used to model UIs and determine some strengths. UML has proved to be useful for modeling user interfaces because it has a rich and integrated set of constructors for modelling UIs, they stated:

"The identification of such strengths and weaknesses can be used in the formulation of a strategy for extending UML to provide greater support for user interface design". Same authors in [7] introduced notation of the extension of UML for modeling interactive applications called Unified Modelling Language for Interactive Applications (UMLi). They introduced the UMLi user interface diagram for modeling abstract user interface presentations simplifying the modeling of the use of visual components (widgets).In addition, they presented a way for modelling the relationship between visual components of the user interface and domain objects obtained from the UMLi activity diagram notation.

Almendros-Jim´enez and Iribarne [8] presented a method for designing GUI from UML Use case model. This method is

applied in four steps. The first step, given a use case diagram of the system, involves all actors and the main use cases where actors are users or external systems interact with the system. Second step describes each Use Case by one or more activity diagrams taking into account the kind of each state of the activity diagram whether it is terminal or non-terminal state to describe the non-terminal state by another activity diagram. Third step translates the use cases and the stereotyped states (terminal state) to class diagrams and finally, from class diagrams, GUI component prototypes are produced.

Through the previous works we can find that most of them are specifically concern with achievement of a design of GUI from UCD as it the case in works [3], [8], where in these works, a design of GUI from Use Cases has achieved. Here, it is worth to note the role of Use Cases and what they present of details about the system. As we know, the Use Cases describe the system functions from an external actors view point and on the other hand, precisely in work [8] focus has been made on the relation <include> without knowing the role of the relation <extends> because each of the two relations has an important role in Use Case Model.

Concerning work in [6], it has been focused on UI Modeling by means of UML. With this work, we have seen the strong and weak aspects in UML and the role of these aspects in UI Modeling.

Work in [7] concerns with the aspects of GUI that are not covered by UML notation and therefore producing additional notation to cover these aspects and in this work it has been specially focused on Interactive Applications.

We find in these works that each has its own advantages and disadvantages and in general, most of them depend on Use Cases. Therefore, this work has to design a general approach through which GUI is structured.

This approach is dependent, in its work, on AD. We have chosen these ADs for their important advantages. The main advantage is the description of workflow behavior within the system and to model the flow of operation.

On the other side, ADs are important in both the modelling of the dynamic aspects of a system and building executable systems anywhere with forward and reverse engineering.

The other advantage related to the control flow is the several kinds of flows provided by the ADs, which include sequential, concurrent or branched flows. These are represented by using elements like join, fork, merge, etc. ADs are used to imagine, build, and document the dynamics of objects [1], [2].

III. Work Steps

The whole work is represented in three preparatory and two fundamental steps. The preparatory are those where the basic work diagrams, patterns and their definitions in details are prepared. In the following sections, the two types of steps are discussed.

A. Preparatory Steps

1. **Providing AD:** Providing the AD of the system, where it is designed by the analyst or person responsible for it.
2. **Formulation of Rules:** Formulate a number of general comprehensive rules.
3. **Preparing DSs Structure:** Specifying the general view of DS and structure of data it contains.

B. Fundamental Steps

1. **Applying the Rules to AD:** Applying the rules to AD, we shall have the contents of DSs.
2. **Translating the DSs to GUI:** When we translate the contents of DSs to their equivalents of GUI elements, we shall have a complete GUI with all of its elements.

IV. The Approach

The approach suggested in this paper consists of two phases. In the first phase, the formulated rules are applied to ADs, where we will have contents of DSs. In the second phase, the contents of the DSs will be translated to equivalent GUI elements, and the structure of GUI is produced. Fig. 1 describes the phases of the approach, the requirements, and results of each phase.

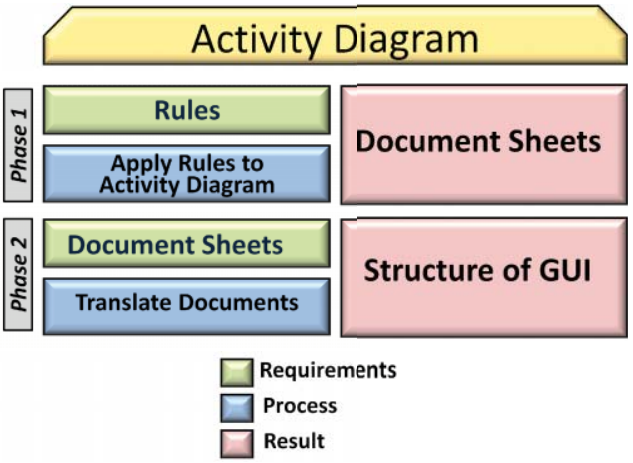

Fig. 1 Phases of the approach

A. First Phase: Documentation

The following is a list of suggested rules based on the importance of input, processing, output, and cooperation with the user for each activity in the AD. An activity is the procedure being constructed; it is also used to express a set of actions [4].

In this relation, the activity directs what is to be performed. This partitioning of an activity was the foundation of the construction of the rules presented below, with an aim to obtain the desired outcome through application.

1. **Rule Numbering Guide:** The elements of GUI which are covered in our work are: Form, Button, Textbox, Label, and List, hence we formulated the numbering of rules based on this matter. Table I provides an explanation for each type of these numbering and its details.
2. **Formulated Rules:** We have 13 formulated rules separated to three groups, first group related to F(K) kind and consists of 4 rules, the second group related to W(L) kind and consists of 4 rules, the last to FW(M) and

consists of 5 rules. So, (K=1,2,3,4), (L=1,2,3,4) and (M=1,2,3,4,5) in the following details of all these rules:

TABLE I
RULE NUMBERING GUIDE

Figure	Description
F(K)	F _ represents rules pertaining to form and points about the forms.
W(L)	W _ represents rules pertaining to other elements of GUI (Button, Textbox, Label, and List) and its related points.
FW(M)	FW _ represents rules containing a combination of the above F & W.

F(K) Rules:

F(1) Each activity in the AD is a form, where this activity is one of these mentioned in Rule **W(1)**.

F(2) GUI elements of activities will compose in the same form when these activities describe the flow of the same operation

F(3) The succession of forms are based on the sequence of the activities in the AD.

F(4) Based upon the forms retrieved from the F(K) and FW(M) mentioned rules, the main form of the system is initiated by using the (Bottom_Up) approach.

W(L) Rules:

W(1) Activities will be considered if it demands GUI elements for:

- Acquiring vital input to implement its complete operation
- Execution of its tasks
- Displaying the output resulting from implementation of its tasks
- Convey a message to the user

W(2) The activities that demand GUI elements for its input are obliges to contain:

- (X) Text boxes
- (X) labels
- (2) Buttons ('Ok' and 'Cancel' button)

where X is a number dependent on the number of data inputs.

W(3) The activity that demands GUI elements for executing its work obliges to contain:

- button (Execute Button)

W(4) The activity that demands GUI elements to display its output obliges to contain:

- (Y) lists
- (Y) labels

where Y is a number dependent on the number of resulting outputs.

FW(M) Rules:

FW(1) The activity that demands GUI elements to interact with its user is an isolated form enclosing a message including a button whose objective is to terminate the display form.

FW(2) If an activity demands a sub-input based on essential input (Multi-input), therefore an independent form is created to acquire any further input applying the rules illustrated in Rule **W(2)**.

FW(3) If an activity demands a sub-output based on essential output (Multi-output), therefore an independent form is created to display any further output applying the rules illustrated in Rule **W(4)**.

FW(4) There is an 'Exit' button for every main form, and a 'Return' button for every sub-form.

FW(5) If GUI designed for the applications which called "application specific GUI" [10], suchlike ATM system, do not require 'Exit' and 'Return' buttons. This type of system has no need to return to select a particular form. Therefore, a 'Welcome' form is revealed, followed by the remaining forms.

B. Second Phase: GUI Structure

As a result of the previous phase, i.e. applying rules to AD, all necessary information about required GUI is obtained and is sorted into sheets called DSs. All details about the elements such as element GUI name, type, number, etc. and other information related with forms and messages show to user are explained into these sheets. Fig. 2 explains the General Design of DSs.

C. The Relation between GUI Designer and Programmer

As already mentioned, one of the goals of this paper is to build a good GUI by greater support provided by collaboration between GUI Designer and the programmer.

This collaboration helps to design a GUI which is clear to the end user and that is one of the principles of good GUI design [5].

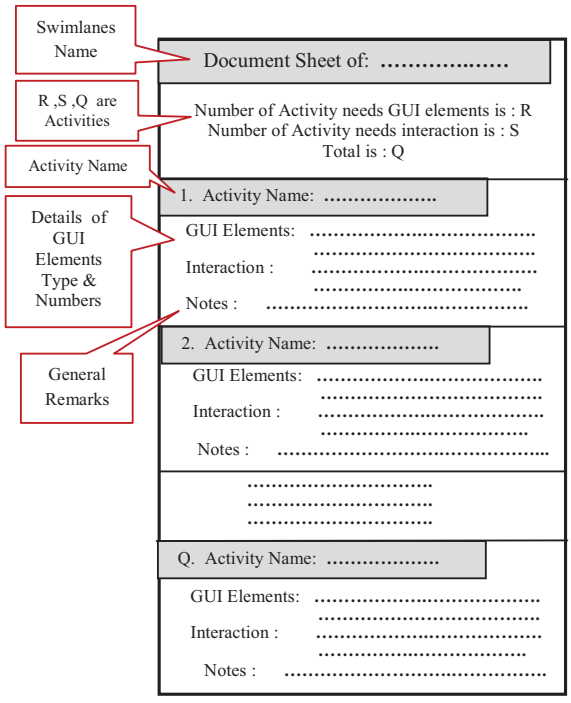

Fig. 2 General Design of a Document Sheet

So, we must know the relationship between GUI Designer and the programmer and where the meeting point is between them. As illustrated in the Fig. 3, we find that the DSs are the meeting point between them and that is shown in the result of phase one **(Documentation)**. The GUI Designer implements

the first phase starting from applying rules to the AD and producing DSs. The GUI Designer delivers these document sheets to the programmer, where the later converts them to the structure of GUI based on the contents of detailed sheets.

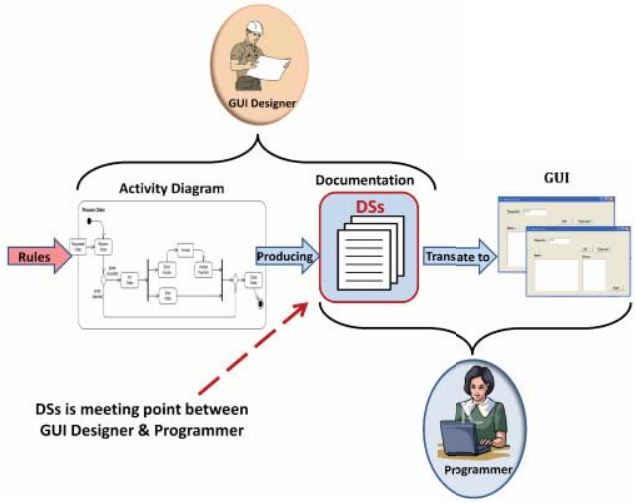

Fig. 3 Relation between GUI Designer and Programmer

V. Case Study

The case study (Home Security System) is presented here to explain how to implement our approach. In this context, Home Security System added to any residents (house, palace, or apartment….etc). Using one of these systems can make you secure and safe. There are many reasons to consider getting a home security system; the most important ever is personal safety. [9]

To understand the system we explain scenario of the work of the system:

1. The idea of the work of the system is based on sending signals to provide security to the place. When this signal is broken, a warning will be issued to alert the person in the place to take the necessary action.
2. The home owner will adjust the system to work through the control panel connected to the system.
3. To guarantee security, no user will use the control panel without entering the password to prove his/her validity as an owner of the home or the place where the system exists, and only three times are allowed to enter the password otherwise, the system will be locked.
4. Through control panel, starting and stopping the system will be made by the home owner in order to make the system active or deactivate.
5. This panel is connected to a monitor to display the current state of system.
6. When starting the system, the test of the state of sensor will be finally made. This sensor is a sensitive device and it changes when anything breakthrough the field of emission of the signals in the circle of the signals. As soon as a breakthrough is made, the following will happen:

- ✓ The system will display a breakthrough on a monitor connected to the system.
- ✓ An order will be given to alerting sound to the homeowner.
- ✓ Send a signal to a phone – must be linked to system which automatically will call through the system, the emergency, the police or the mobile of the homeowner, according to the phone number previously set in the preparation of the system (Fig. 4).

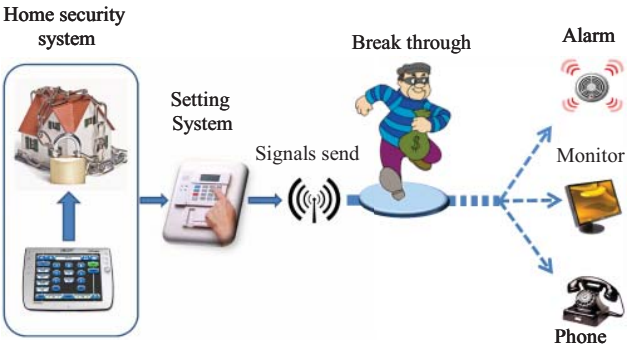

Fig. 4 Home Security System

A. Applying First Phase (Documentation)

In the first phase, an AD for the system must be provided to applying the rules and obtained details of document sheets. Fig. 5 shows an AD of Home Security System.

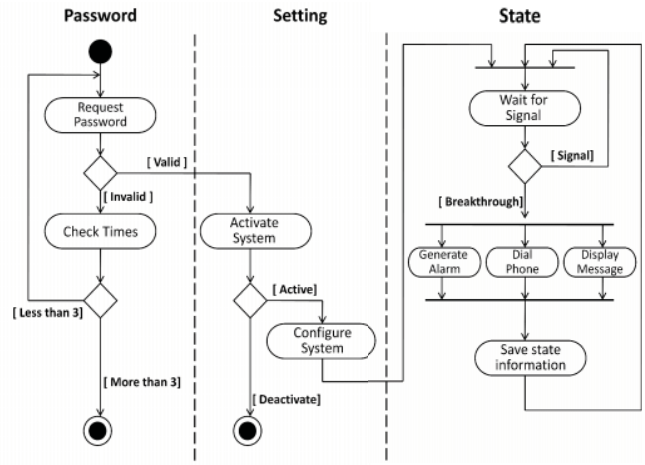

Fig. 5 AD of Home Security System

After applying rules on AD we find four activities need GUI elements to implement its works, Fig. 6 shows these activities which bordered by red circles and labeled A,B,C and D, and Fig. 7 shows DS of each activity of the four activities.

Activities in Fig. 6 are described as follows:

1. Activity labeled A: Demands GUI elements for its input (Rule W(2)).
2. Activity labeled B: Demands GUI elements to interact with its user (Rule FW(1)).

3. Activity labeled C: Demands GUI elements for its input (Rule W(2)), and it demands a sub-input based on essential input (Multi-input) (Rule FW(2)).
4. Activity labeled D: Demands GUI elements to interact with its user (Rule FW(1)).

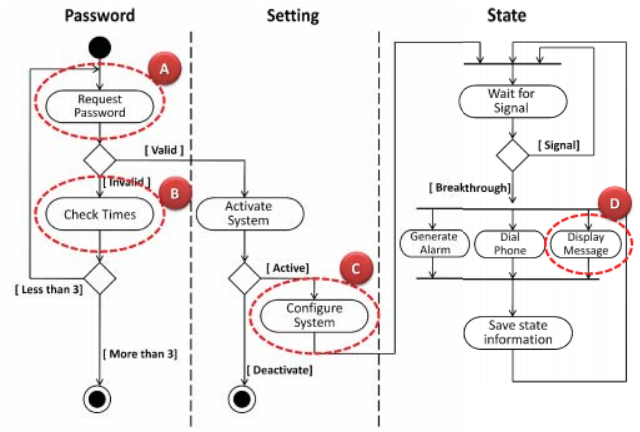

Fig. 6 AD of Home Security System after applying rules

B. Applying Second Phase (GUI Structure)

In this phase, we can easily translate the information in DSs to equivalent GUI elements to obtain the GUI structure. The structure is shown in Fig. 8.

VI. Conclusions

Based on the fact that GUI is an important joining point between user and applications or systems, and relying on the fact that the success of the applications are based mainly on the success of its GUI, the GUI development process has been presented by our approach to develop GUI using ADs.

The approach consists of two consecutive phases— Documentation phase and GUI Structure phase.

The input of the first phase is AD where as the output is DSs, as for the second phase, the input is DSs along with GUI elements and the structure of GUI as the outputs.

So, AD is the backbone where as the DSs are the joining point of the phases of the approach and the structure of GUI is the result— taking into consideration the formulated rules.

The structure of GUI was satisfactory and comfortable obtaining to the system of case study— Home Security System—and by comparing the results of GUI with both the scenario of the system and its work flows, we found that it's a great fit, and gives the required consent.

Document Sheet of: Password

Number of Activity needs GUI elements is : 1
Number of Activity needs interaction is : 1
Total is : 2

1. Activity Name: **Request Password**

GUI Elements: (1) Label [Enter Password]
(2) Button [Ok , Cancel]
(1) Text
Interaction : ---
Notes : ---

2. Activity Name: **Check Times**

GUI Elements: ---
Interaction : [If incorrect password interred for three times]
Notes : ---

Document Sheet of: Setting

Number of Activity needs GUI elements is : 1
Number of Activity needs interaction is : 0
Total is : 1

1. Activity Name: **Configure System**

GUI Elements: (2) Label [Start Date , Breakthrough Time]
(2) Button [Ok , Cancel]
(2) Text
Interaction : _ _

Notes : Activity needs multi-input and that's mean derivation one of two new forms, depends on breakthrough time

If 6<breakthrough time <23 then form contains:
(2) Label [Police Station Phone Number , Cell Phone Number]
(2) Button [Ok , Cancel]
(2) Text

If 6>breakthrough time >23 then form contains:
(1) Label [Police Station Phone Number]
(2) Button [Ok , Cancel]
(1) Text

Document Sheet of: State

Number of Activity needs GUI elements is : 0
Number of Activity needs interaction is : 1
Total is : 1

1. Activity Name: **Display Message**

GUI Elements: ---
Interaction : [When a breakthrough of signal occur]
Notes : ---

Fig. 7 Document Sheets of online shopping system

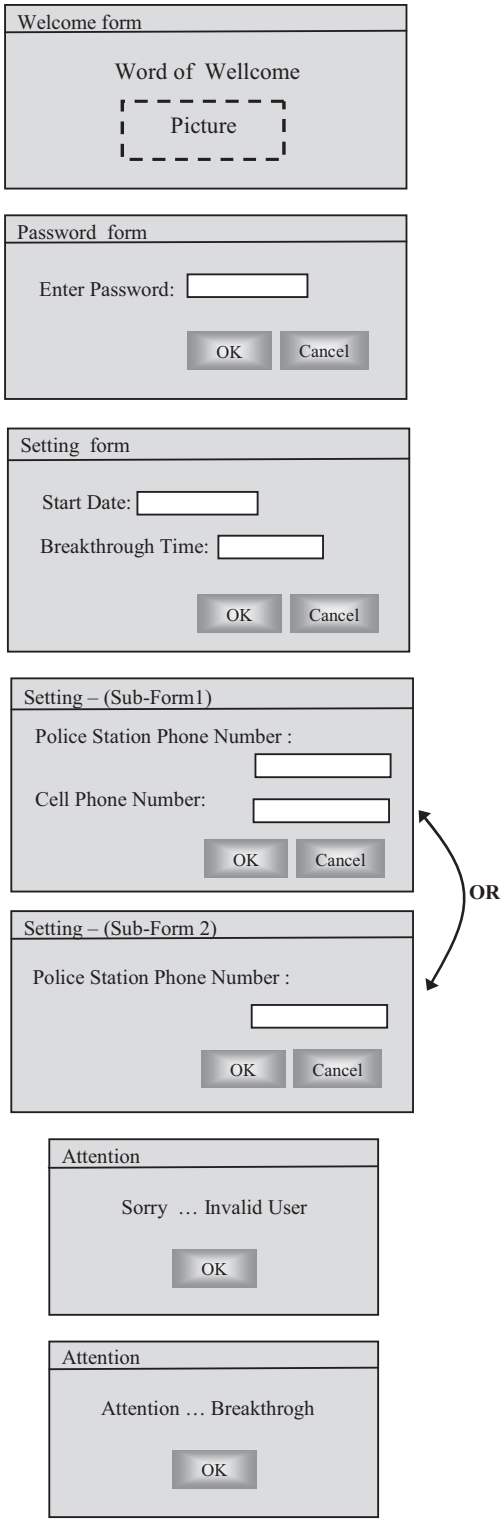

Fig. 8 Structure of GUI of Home Security system

REFERENCES

[1] B. Agarwal, "Some Rules to Transform Activity Diagrams into Colored Petri Nets", In International Journal of Recent Technology and Engineering (IJRTE), ISSN: 2277-3878, Vol.1, Issue-5, 2012

[2] G. Booch, J. Rumbaugh and I Jacobson, *The Unified Modeling Language User Guide*, Publisher: Addison Wesley First Edition October 20, 1998 ISBN: 0-201-57168-4, 512 pages

[3] M. Elkoutbi and R. Keller, "User Interface Prototyping Based on UML Scenarios and High-Level Petri Nets", In 21st International Conference on Application and Theory of Petri Nets (ICATPN '00), pages 166-186, 2000.

[4] M. Felici, "Activity Diagrams", University of Edinburgh, School of Informatics, 2004

[5] J. Hobart, **"**Principles of good GUI Design", Unix Review, vol. 13, no. 10, pp. 37-46, Sep. 1995.

[6] P, da Silva P. and N.W. Paton, "User Interface Modeling with UML", in 10th European-Japanese Conference on Information Modelling and Knowledge Representation, 2001.

[7] P. da Silva and N. W. Paton, "UML*i:* The Unified Modeling Language for Interactive Applications," In Proceedings «UML» 2000, Evans, A., Kent, S. (Eds), LNCS, Vol. 1939, 117-132, Springer, 2000.

[8] J. M. Almendros-Jim´enez and L. Iribarne, "Designing GUI Components from UML Use Cases," In 12th IEEE International Conference and Workshops on the Engineering of Computer-Based Systems (ECBS'05), pages 210–217, IEEE Computer Society Press, 2005.

[9] Http://www.ehow.com/facts_5011416_why-home-security-systems-important-html (2009).

[10] Http://www.reference.com/browse/graphical+user (2015).