Design Patterns for Dialog Boxes in User interface Mobile Applications

Abstract

The aim of this study is to investigate the emerging challenges accompanying the ongoing development of information technology nowadays, especially in the field of smart phone applications. This makes IT experts, designers, manufactures and researchers in need of finding better and more effective solutions to overcome these challenges and obstacles. One of these challenges is presented when the SW keyboard is shown and hidden in UI applications of PDA , PC or any other mobile devices. This keyboard is shown when the user wants to enter a text, which leads to the occupation of the application area by this SW keyboard. This eventually means that the application will have less room for its " normal interaction ".The main aim of this research is to use a Model view controller (MVC) designpattern to solve this problem associated with SW keyboard .It is intended to make the interaction of dialog boxes when entering a text on mobile UI easier ,more effective and more practical.

Keywords: user interface (UI),Personal Digital Assistant (PDA),Software(SW),Personal Computer (PC),Information Technology(IT), Global Positioning Systems(GPS),Model-view-Controller (MVC).

1. Introduction

Smart phones have been developed to include many different functions like game applications wireless and Wi-Fi network connections and many more. For this , user interface for smart phones has become a big concern worldwide. Due to the wide variety of inputs, small screens and sensitivity to interrupt tasks, mobile user interfaces present big challenges that are not presented on desktop which are derived from the same variety of user input methods, such as touch screens, accelerometers, and GPS capabilities. Since designing mobile interfaces is a relatively new practice, and the field is under fast progress of technology nowadays. For this very important

reason , there is a need for effective and flexible

means to tackle those obstacles when designing user interface for modern mobile phones which have high in-built sensitivity and a wide variety of options and functions, One of the problems designers encounter in the mobile domain is the range of variation of the capabilities from a device to device and platform to platform. Biel stated that one phone can have a touch screen and no buttons, you can use a full keyboard, or one phone can have a number pad and some arrows. These variances are equally different when it comes to computing power, screen capabilities, and more. To accommodate these differences likely requires building multiple designs for each platform the interface needs to run on [1].In this research work, we are using MVC architecture pattern for building a user interface of mobile, which is fast and efficient way of creating different UI for mobile,This pattern is used as a methodology for designing the mobile user interface [2].Generally, design patterns can help solve complex design problems if they are properly used, however the main advantage of using the MVC pattern is decoupling the business and the presentation layers [3].MVC is defined as a common design pattern to integrate a user interface with the application domain logic. MVC separates the representation of the application domain (Model) from the display of the application's state (View) and user interaction control (Controller).

The MVC design pattern is comprised of three major components[23] the Model (The Data Layer), the View (The User Interface Layer) and The Controller (The Business Logic Layer).The main problem in this research is how to resize dialog boxes to avoid some parts of these text dialog boxes become invisible. In this context, what is meant by TEXT BOX or DIALOG BOX is any message created or looking up a contact on the CONTACT LOG saved in your phone. Resizing mainly prevents hiding some parts of the dialogs which eventually become invisible once a long text is entered. Also, the severity of this problem largely depends on the type and style of user interface being used. The fact that smart phones have small screens which do not allow for much space and, therefore less interaction for users. So the smaller screens makes this issue inevitable, challenging and totally inconvenient among smart phone users.

2.Overview of Design Pattern

A user interface pattern was given center stage with the PEICS conference, which focused explicitly on issues surrounding designing and engineering that is using design patterns. Amongst the things that have been Design patterns represent a highly effective way to improve the quality of software engineering. Due to its ability to a capture the best practices and design knowledge based on real experience of software design, making it available to all software engineers [4]. It presents a generic proven solution to a common recurring design problem. A design pattern in software engineering is a general repeatable solution to a commonly occurring problem in software design A design pattern isn't a finished design that can be transformed directly into code. It is a description or template for how to solve a problem that can be used in many different situations. Object-oriented design patterns typically show relationsand interactions between classes or objects ,without specifying the final application classes or objects that are involved[4].

2.1 Benefits of Design Patterns

As we know that design patterns have many benefits. The following states the main benefits of design patterns[5]. They can speed up the development process by providing tested, proven development paradigms.

1-Effective software design requires considering issues that may not become visible until later in the implementation**.**
2-Reusing design patterns helps prevent subtle issues that can cause major problems. Therefore, design patterns improve code readability for coders and architects familiar with such patterns.
3-Common design patterns can be improved over time, making them more robust than ad-hoc designs[5]. The next section provides types of MVC.

2.2Model View Controller (MVC) Design Pattern:

The term MVCis made from Smalltalk, which is a programming language that was particularly designed to support the concepts of object-oriented programming.Model View Controller design pattern is also an architectural design that helps in making a user interface mobile applications modify-able with future requirements by splitting the whole application into three components, model, view and controller [7]. Each of these components handles discrete set of tasks.

-Model: is the core of the application. This maintains the state and data that the application represents. When significant changes occur in the model, it updates all of its views

-Controller: basically takes the role of a vocal point between the model and the view.

-View: The user interface which displays information about the **Model** to the user. Any object that needs information about the **Model** needs to be a registered **View** with the **Model**.Initially, MVC was used for designing and building desktop applications with rich graphical user interfaces. Over time, the original MVC pattern evolved and variants emerged driven by technological evolutions and new needs. Nowadays, MVC is used for integrating interface logic with domain logic in development of various domains, such as Web applications and Mobile systems.

2.2.1 MVC Interaction Cycle

This section illustrates the MVC Interaction Cyclein four steps [8]**:**

-The first step: The user interacts with the view through a user input such as clicking a button or a link on a user interface. The view sends the user input event to the controller. The controller handles this request.

-**The second step**: The controller sends calls to the model to modify its state according to the request

-The third step: The controller sends calls to the view to modify its state. In fact, when the controller receives a request from the view, it may need to modify the view state; for example, the controller could enable or disable certain buttons or menu items in the user interface.

-The fourth step: The model updates the view representation when its state is changed. Actually, something changes in the model. This change is based on some requests by user input, such as clicking a button, or some other internal changes. The model updates the view that makes its display and eventually the user interface changes. This means that the view updates its state directly from the model**.**

2.2.2 Advantage of MVC DesignPattern

The biggest advantage of the MVC design pattern is thatit separates the model from the view. As stated earlier that the model represents the data and the business rules and the view represents elements of the user interface such as texts, images, and form inputs.This separation allows for easy changes for each object without affecting each other. It also leads to easier maintenance and modification of the UI[9].Separates the three objects that lead designers to work on the UI of mobile devices without worrying about the underlying data .It also helps developers focus on the data instead of being too concerned about data presentation and avoid code repetition. MVC has the ability to reduce designing time because programmers who focus on the controller object can work independently while designers are responsible only for the view object or model object. MVC has the ability to bring about changes in the view object without recompiling the code of the model objects or the object of the controller[9].

2.2.3 MVC Architectural Pattern of Mobile Web Application

It was already mentioned in the introduction, that mobile technologies is one of the swiftly evolving areas in information technology. Mobile technologies are a perspective and a well suited investment for many reasons. Most electronic devices are becoming smaller, requiring less energy and a lower data transfer rate. Nowadays, more and more people start to use mobile devices because they are simply very useful tools in for a wide range of purposes and fields. As we all know that there are so many types and brands of mobile devices and different ways to present website content to them. Also, there was a

Wireless Application Protocol (WAP) Forum established, WAP 1.0 standard was introduced , which described complete software stack for mobile internet access [10].Since2004, WAP disappeared from handsets as there is now support for full HTML even in low-end market phones. The market of mobile devices will be increasing in the next few years. So using an effective method to easily transfer existing applications into the new market will be very valuable as mobile web application using MVC architectural pattern ,which is a very fast and efficient way to build different end-user sites without the need of redeveloping the core application[10].

2.3 ICONIX

ICONIX is an object oriented software development methodology, consists of dynamic and static workflows [12], and it uses UML diagrams in a four-step process that transfers from use case to code.ICONX process is very suitable for MVC and focuses on the area that lies in between use cases and code. It also describes the core logical analysis and design process.This essential logical analysis is designed to move the user from requirement analysis to implementation in a quick and efficient manner.A very essential element of the ICONIX Process is the use of **Jacobson's Robustness Analysis** technique to bridge the gap between requirements analysis and detailed design. In fact, this analysis approach is the most convenient for MVC.

2.3.1 ICONX Process

The ICONIX process is divided into four milestones or founding steps .At every stage; all steps are carefully reviewed and updated.

Milestone 1: Requirements Review
This step is considered as requirements analysis, which is performed by identifying a problem statement and real-world domain objects in a domain model. It also includes identified functions requirement by Use Case diagram, which generates some prototypes for each use case. From this analysis, use cases can be identified, a domain model is produced and some prototype GUIs are made.

Milestone 2: Preliminary Design Review
Another very important milestone is Robustness Analysis. It is considered as a middle ground between analysis and design as it discovers objects for each use case and updates the domain model according to the objects discovered.Once use cases are identified, texts can be entered to see how users and the system will interact. Then, robustness analysis is done to find any potential errors in the use case text which means the domain model is updated accordingly. The use case text is important to observe how users will interact with the proposed system.

Milestone 3: Detailed Design Review
This step is mainly concerned with the design. We use objects which are discovered from the robustness analysis to make sequence diagrams, and we use the domain model as explained in the previous step to design the class diagrams. During this stage of the ICONIX process, the domain model and use case text from milestone 2 are used to design the system . In this step ,class diagrams are produced from the domain model and the use case texts are used to make sequence diagrams.

Milestone 4: Deployment
The final step is the execution of all the desired system .Unit tests are written to verify if the system matches to the use case text and sequence

diagrams. Finally, the code is written using the class and sequence diagrams as a guide.

2.3.2 Robustness Analysis

Robustness analysis intends to fill the gap between analysis (the what) and design (the how). Actually, robustness analysis is considered as a preliminary design when designers makeassumptions on the design structure and start thinking of any possible technical solutions.For supporting robustness analysis, they use robustness diagrams. it uses UML concepts. and also It is a specialized communication diagram that uses stereotyped objects. It was Introduced by Jacobson and it basically analyzes use cases and estimates the first set of objects that participate with those use cases. It also classifies objects according to the roles that use cases play. Robustness analysis helps discover objects and identify the main domain classes before design or implementation[13]. Robustness Analysis consists three of elements

-Entity objects: describe objects dealing with persisting states.

-Boundary objects: describe links between the system and environment.

-Controller objects: describe use-case specific behavior

2.4.2.1 Robustness Analysis in MVC Design Pattern

MVC objects are related to EBC objects in one-to-one mapping. Thus, entity object maps onto model object, boundary object maps onto view The MVC and EBC are techniques that separate responsibilities in software to avoid potential coupling [13].ICONIX is a methodology approach that uses entity, boundary, and controller objects that presents a fundamental approach for modeling software systems , and also it is the most convenient for GUI-based Object-Oriented (OO) applications.When we start extracting objects, analyze use cases and attempt to ignite interaction among diagrams through these objects, there are four primary rules that must be followed:

-Actors are allowed only to interact with the boundary objects.

-Boundary objects are allowed only to deal with controllers and actors.

-Entity objects controllers are allowed to engage in the same interaction.

▪**-Controllers** basically interact with boundary objects and entity objects, and to other controllers, but not with actors.Generally, there is one basic map between the actor and the boundary objects. In this context, The controller objects can merely interact with all the objects, but not allowed to access the actor. Also, the entity objects can interact with each other through the controller object. Thus, the controller object is considered the route of communication between objects [13].

4. Methodology:

This section include both designing and implementation.

4.1 Designing a Mobile User Interface Using MVC Design Pattern

The methodology used for solving this problem is ICONX. It involves many steps including the selection of a proper design pattern. The ICONIX process is a streamlined approach to software development. One of the main advantages of this process is that helps extract code from use cases quickly and efficiently. This process is done by using a concentrated subset of the UML and related tools and techniques. Thus, MVC proves to be the most effective and convenient design pattern to fix the stated problem.

3.1.1 FrameworkofMVC Design Pattern

MVC pattern is usually implemented by ASP MVC.NET and J2EE, but in this research use ASP. ASP.NET is a development framework for building web pages and web sites with HTML, CSS, JavaScript and server scripting. This application framework was developed by Microsoft for building desktop applications. It is based on Common Language Runtime (CLR) which gives developers the freedom to develop applications in multiple languages like Visual C#, VB.NET, Visual J#, Visual C++ and other languages that are supported by the .NET framework .ASP.NET supports three different development models: Web Pages, MVC , and Web Forms.

3.1.2 ICONX Methodology

It mainly consists of four stages:

1. Stage One: Requirements Review

Once we think about implementing the ICONIX process, we must bear in mind some requirements for analysis. This analysis enables use cases to be identified, a domain model can be produced and some prototype GUIs are created as a result. This step includes identifying domain model, use cases and generating GUI prototyping for each use case.

3.1.2.1 Identifying Real-World Domain Objects

It simply focuses on the real world and describes the problem of user interface.

3.1.2.2 Allocate Functional Requirements on Use Case Diagrams

The use case diagrams are usually defined during the requirements activities to capture the requirements of the functionalities of the system. The scenario of the use case is to describe the interaction of the user with the system.The use case illustrates all actors (actor is any person who has interaction with the user interface) .Another function of the use case is to describe how the user interface responds to those actors. The representation is used to extract the functional requirements of the system.

3.1.2.3 Generate GUI Prototyping for each Use Case

At this stage, the final prototypes of the main components are placed. The user interface is the main component and user interface prototypes will be generated for each use case of the system being developed to illustrate their actions. For examples, the elements of the proposed GUI include:

- **Button**: a control that can be clicked to perform an action
- **A text box**: allows the user to enter text information which is to be used by the program.
- **A list box**: a type of box within a collection of graphical user interface widgets that can be grouped.

3.1.2.4 Stage Two :Preliminary Design Review

This step is mainly concerned with using a robustness analysis, which is the best for visually describing the MVC.This analytical procedure has two sub-steps: Perform Robustness Analysis and Update Domain Model.

3.1.2.5 Perform Robustness Analysis for each Use Case

The *Robustness diagram* includes multiple elements. It actually consists of a class diagram and an activity diagram. It visually represents behavior of use case , showing both participating classes and software behavior. A robustness diagram is probably easier to read than an activity diagram since objects speak to each other.

-**Boundary**: is the interface between the system and the outside world. Boundary objects are typically screens or web pages

-**Entity**: Entities are usually objects from the domain model.

-Control: Control objects are the "glue" between boundary and entity objects.

3.1.2.6 Update Domain Model

This sub-task is performed through extracting entity classes from the domain model, then adding any missing entities discovered during the robustness analysis.

3.2.1 Stage Three: Detailed Design Review

During the stage of ICONIX process, the domain model and use case text from stage 2 are used to design the system being built. A class diagram is produced from the domain model and the use case text is used to make sequence diagram. This step is categorized into: Sequence diagram, Design patterns ,and Class diagram.

3.2.2.1 Generate Sequence Diagram from EBC on the Robustness Diagram.

This step explains the sequence model, which is one of the UML models .We must show interaction between the set of objects,messages being sent and received by those objects and demonstrate the behavior of objects. In fact ,the boundary and entity classes in a robustness diagram will generally become object instances in a sequence diagram, while controllers will become messages.

3.2.2.2 Select Suitable Design Pattern

As we know, there are 23 different types of design pattern. However, we still need to define what we actually mean by *MVC design pattern*. MVC is basically a set of classes to build a user interface. Those classes that define the main MVC relationship are Observer and Strategy .The diagram below illustrates the three essential types of objects in the Observer: the model is the application data, the view is the screen and the controller defines the way the View reacts to user input. As shown, this comprehensive Observer process allows us to attach multiple Views to the same model. This Observer pattern aims at defining the one-to- many relationships between the subject and the observers. This means that if the Subject is altered, then all Observers are updated .The Subject here keeps the list of the Observers and can attach and detach objects to the list. Another component of MVC is the View –Controller relationship.The Controller is used by the View to implement a certain type of response .It also allows the View to respond differently to user input .This View-Controller connection is an example of the strategy design pattern. Now we can state that the View takes on the role of the Observer object and the Model acts as a Subject from the Observer pattern.

3.3.1.1 Update the Domain Model into Class Diagrams as needed.

Class diagrams indicate the set of classes and relationship between them. Every class contains three elements: Class name, Attributes and Method or Operation.

3.2Stage Four: Implementation

This is the final stage of the ICONEX methodology. It verifies the system which will match up with the Use case, Text and Sequence diagrams. Finally, Code is written by using the Class and Sequence diagrams .In this research we used the ASP.NET MVC framework for designing user interface of mobile.

4.1 Implementation

In this section we explain the actual implementation of the proposed solution .The whole process is based on the selected methodology and the programming language for user interface of mobile devices.

4.2 Steps of the ICONIX Methodology

This methodology is very efficient at solving such mobile related applications. It consists of four main

steps and each step has its own secondary parts. All of these steps will be executed to solve the problem according to the proposed methodology.

4.2.1 Step One :Requirement Analysis

This step is divided into three subtasks, these subtasks are:

-Domain model: is the Class in its primitive status. In this step there are three classes: Storage Class , MobileController Class andUser Class.

-Use Case Diagram

This step illustrates the roles of the proposed actors who interact with the application. This step also shows the most interactive user/s with the general functions of the system.

4.2.2 Step two:Preliminary Design Review

This step consists of two subtasks: Perform robustness analysis for each use case and update domain model.

-Perform Robustness Analysis include:

1.Robustness Analysis for Star.
2. Robustness Analysis for contact setting
3. Robustness Analysis for Create Message.
4. Robustness Analysis for Search.
5. Robustness Analysis for DELETE Contact

The main operations that MVC contains :

1-Adding a Model

Folders of Models contains the classes that represent the applicationModel.

2- Adding a Controller.

3-Adding Views for Displaying the Application.

5.1 Conclusions

One of the most important findings in this research is that Design Patterns can solve specific design problems and make object oriented designs more flexible and reusable. They also help designers reuse several designs, including design alternatives to avoid other alternatives that might compromise reusability.It was also concluded that the main objective of design patterns is to reuse good practice in the design of newly developed applications. Another important objective of using design patterns is to develop common applications and better understanding of the overall designing process which is performed by reusing the same generic names for implemented solutions.

This thesis has concluded that MVC patterns are considered as pioneering patterns for synchronizing user interfaces with domain data. It is actually an excellent choice for Web-based applications.In fact,Web structures naturally support the division of responsibilities of the components of MVC patterns . However, these patterns suffer from poor handling of view state logic, and assume decoupled View and Controller which does not match with many state of the frameworks in project. Robustness Analysis helps discover objects for each use case and identify the main classes before designing or implementation , also it is the best way to analyze MVC because it represents three objects :Entity objects present classes, Boundary objects present links between the system and the external environment and Controller objects present logical software functions. Finally, we used ASP MVC .NET framework to solve the problem. After applying ICONEX and MVC.

5.2 Future Work

The researcher will attempt to implement all of the proposed work on "actual mobile phones connected to Internet which can allow for more options for users such adding and editing photos, looking up contacts via emailing. Another future goal is to research for more up-to-date design patterns and find what other UI related problems can be solved by using design patterns.

References

[1] Bettina Biel, Thomas Grill, Volker Gruhn, "Exploring the benefits of the combination of a software architecture analysis and a usability evaluation of a mobile application", *Journal of Systems and Software (JSS)* Vol 83(11), pp 2031-2044, 2010.

[2]S. Alpaev ." Applied MVC Patterns. A pattern language", presented at the Viking PLoP conference,2005.

[3][23] JoydipKanjilal ,"Implementing the MVC Design Pattern in ASP.NET",article,31 Jan 2008.

[4] DragosManolescu, Markus Voelter, and James Noble, "Pattern Languages of Program Design ", 1st ed.: Addison Wesley, 2006, vol. 5.

[5].www.javagyan.com/blogs/design-patterns Acceded in [March_2013].

[6] P. Argall1, R. J. Sica1." Development of a new Lidar Data Analysis Program". The University of Western Ontario London ,2013.

[7]A. Kolu." MVC FRAMEWORKS IN WEB DEVELOPMENT". Master thesis, University of JYVSKYLN, 2012.

[8]L. D Í E Z." Secure, scalable and component based Web shop using Struts and Hibernate". Master of Science Thesis Stockholm, Sweden 2006.

[9]TomášChlouba "Design Patterns in Mobile Architectures", Topic paper , University of Hradec Kralove, Rokitanskeho 62, Hradec Kralove, 500 03 Czech Republic, October 31, 2010.

[10]ICONIX Process ".available : http://iconixprocess.com/iconix-process/.

[11]F. Cover. D. Rosenberg, M Stephens ."*Use Case Driven Object Modeling* with *UML*: Theory and Practice" .ApressBerkely, CA, USA ©2007, Jul 31, - 472 pages.