**A study of possible alternatives for replacement of Director MX software in the Ozlab setup**

By Abhishek Gupta & Bharadwaj Radhakrishna

Under

Prof. John Sören Pettersson

May 2011

**INDEX**

1. Background for Ozlab

1.1 Definitions

1.2 Theory

1.3 Ozlab Behaviors

**2. Example of Ozlab testing**

**2.1 Processes involved**

**2.2 Developing the graphics**

**2.3 User Testing**

**2.4 Key points gathered during first user testing**

**3. Flaws in the existing method/setup of testing used**

**4. Recommended solutions**

**4.1. Building a new dedicated platform from scratch: Possible recommendations**

**4.1.1 Variables**

**4.2. Using other existing prototyping tools:**

**4.2.1What is prototyping?**

**4.2.2 Why is prototyping needed?**

**4.2.2.1 Features and Variables with respect to Ozlab**

**4.2.3 Tools for prototyping**

**4.2.4 Low fidelity prototypes**

**4.2.5 Medium fidelity prototypes**

**4.2.6 High fidelity prototypes**

**4.2.7 Mind map**

**5. Morphological chart**

**6. Conclusion**

**References**

**1. Background for Ozlab**

Ozlab encompass two aspects of testing.

Firstly, the facilities are using the Wizard-of-Oz technique. The Wizard-of-Oz technique basically presumes a test environment that disguises the test moderator (TM) from the test person (TP) as in Fig. 1. This is usually done by having 2 adjoining rooms for the tests connected through a one way mirror. Usually the test is conducted with video and audio recording to save the test sessions. The sessions are controlled by TM's behind the one way mirror; they have access to all audio and video recording as the test session progresses helping the TP through the test scenarios.

Secondly, the other aspect of the program is a matter which is vital to Ozlab-based testing. Ozlab is a program that enables the TP to see what the TM shows. All the prototypes that is viewed and "controlled" by the TP is in fact all controlled by the TM, every click and every change in screen image are operated by the test moderator. This is so that the prototypes won't have to be programmed, allowing more focus to be on the test and the test results.

*"The benefit for designers, whether they are professional or inexperienced, of avoiding programming is the ease with which alternative proposals can be produced for discussions with peers and for testing with prospective end-users."* (Pettersson 2002)

Currently the system is based upon Macromedia Director MX and its programming language LINGO. This is not an ideal program to use since it was changed quite drastically since Adobe acquired Macromedia in 2005. The Current Director version from adobe is 11.5 and it cannot be used by the Ozlab converter.



Figure 1. Example of how an Ozlab can be set up. Illustration by Anders Karlsson. © Karlstad University

**1.1 Definitions**

In two student reports (Brundin 2011, Lamberg 2011) we found the following definitions:

**Test moderator (TM)** - A moderator during test sessions.

**Test leader (TL)** - Controls the responses during Ozlab test sessions.

**Test person (TP)** - An individual a test is done to.

**Low-fidelity prototypes (lo-fi) -** Primitive prototype to cover the basics to be tested.

**High-fidelity prototypes (hi-fi) -** Advanced prototype to test a near-finished product.

**Ozlab** - Test facilities for conducting Wizard-of-Oz tests.

**Test runner -** The program that the prototypes are tested in**.**

**Interaction shell -** The prototype that is used in Ozlab is called interaction shell.

**Shell-builder -** The part of Ozlab where the shells for the test runner gets built.

**Wizard-of-Oz (WOZ)** - Test technique where one person (the Wizard, often hidden)

executes the responses seen or heard at the test person’s computer.

**1.2 Theory**

**User Testing**

User testing is essential when assessing how well a product or program will work when it released on the market, or at least in minimizing the flaws of the product. Ruben and Chisnell (2008) discuss usability tests as a research tool with a lot of uses. It can be used with a large group of participants with a complex design, or it can be small tests with individuals. Both can be used to gather either quantitative or qualitative data, both formal and informal. It all depends on what objectives the user tests are determined to meet.

The main reasons for user tests are to establish the design, remove design flaws, eliminate user frustration, and increase profit. There are several benefits of fulfilling each of these reasons; eliminating user frustration for instance will give the consumers a motive to continue to buy the product released by the company. It can also be truly valid since it will give the customers a reason to think that the developer company truly thinks of the customer’s priorities (Rubin & Chisnell 2008).

**Prototypes**

When user tests are conducted it is common that prototypes are involved. There are several kinds of prototypes that differ both in the way they are constructed and in content. Firstly we have low-fidelity (lo-fi) prototypes which are easy and fast to make. An example of lo-fi prototypes is the paper prototype which is a very basic sketch that shows a fast overview of what a program or layout can do. It becomes easy to test this on test persons by just having several sketches and switch them depending on what the TP chooses in menus. It is also usable for just showing a rough draft of a design.

*"The value of the paper prototype or paper-and-pencil evaluation is that critical information can be collected quickly and inexpensively"*

(Rubin & Chisnell 2008)

The opposite of the lo-fi prototype is the high-fidelity(hi-fi) prototype. The hi-fi prototypes often show a lot more of the product than the lo-fi prototypes. A hi-fi prototype can be made by having depth in the content of the prototype or by having an almost complete design. It is often used to test whether the graphical design or navigational structure of the product is satisfactory (Cagan 2008).

Ozlab is a mixture of both hi- and lo-fi prototyping. It's possible with the Ozlab setup to create very lo-fi prototypes and test them in a hi-fi test environment as well as creating hi-fi prototypes and test these in a hi-fi environment (Pettersson 2001).

**1.3 Ozlab Behaviors**

Ozlab is filled with behaviors; they can also be referred to as functionality. This functionality is what makes the different scenes and images truly intractable for both the TP and the TL. There are several different behaviors available as listed below (Siponen et al. 2002):

**objektFlyttbartAvTL** Makes an object moveable by the TL

**objektFlyttbartAvTP** Makes an object moveable by the TP

**objektFlyttbartAvTLTP** Makes an object moveable by both TL and TP

**objektGömbartAvTL** Makes it possible for the TL to hide an object from the TP

**objektKnapp** Creates a clickable button out of any object

**objektKomIhågPosition** Makes is possible for the TL to bring an object back to its original position by clicking the restore button

**objektOsynligtFörTP** Makes an object permanently invisibly to the TP

**objektOsynligtFörTL** Makes an object permanently invisibly to the TL

**sidmarkör** Is used as pause-function in the interaction-shell. This function is made by adding code in the frame script channel in the Score-window. This needs to be placed in the same column as a marker in order to work properly.

**spelaUppLjudFörTP** Allows the TL to control the playback of an audio-file by clicking on director objects of the type button.

**textfältEditerbartAvTL** allows the TL to edit an object of the type text-field, can be combined with the function below

**textfältEditerbartAvTP** allows the TP to edit an object of the type text-field, can be combined with the function above.

**2. Example of Ozlab testing**

**Four in a row computer game**

Four in a row (also known as Captain's Mistress, Four Up, Plot Four, Find Four, connect Four, and Four in a Line) is a two-player game in which the players first choose a color and then take turns dropping their colored discs from the top into a seven-column, six-row vertically-suspended grid. The pieces fall straight down, occupying the next available space within the column. The object of the game is to connect four of one's own discs of the same color next to each other vertically, horizontally, or diagonally before one's opponent can do so. There are many variations on the board size, the most commonly used being 7×6, followed by 8×7, 9×7, and 10×7.

The game was first sold under the famous Connect Four trademark by Milton Bradley in February 1974.

**Requirements of the game to be made in Ozlab**

In this game the player will play against the computer according to the regular rules. When the player has chosen the color, the game will begin; either it’s the player or the computer that starts. If someone gets four in a row, the computer will tell who has won. There has to be a help function. During ongoing game, the player cannot gain any help. There has to be two different times for consideration for the computer, 5 seconds or 15 seconds. The player chooses this before the game begins. The player always has 10 seconds time for consideration otherwise he/she has lost.

**Requirements for the interaction shell**

The interaction shell should contain:

* Movable objects
* Hideable objects
* More than on stage
* Marker navigation buttons.

**2.1 Processes involved**

**Developing the information architecture**

The first step in the process was the development of the information architecture of the game and determining the flow of the game proceedings (see Figure 2). Once this was determined, the different functions that had to be included into the game were set into different categories depending on the page of use.

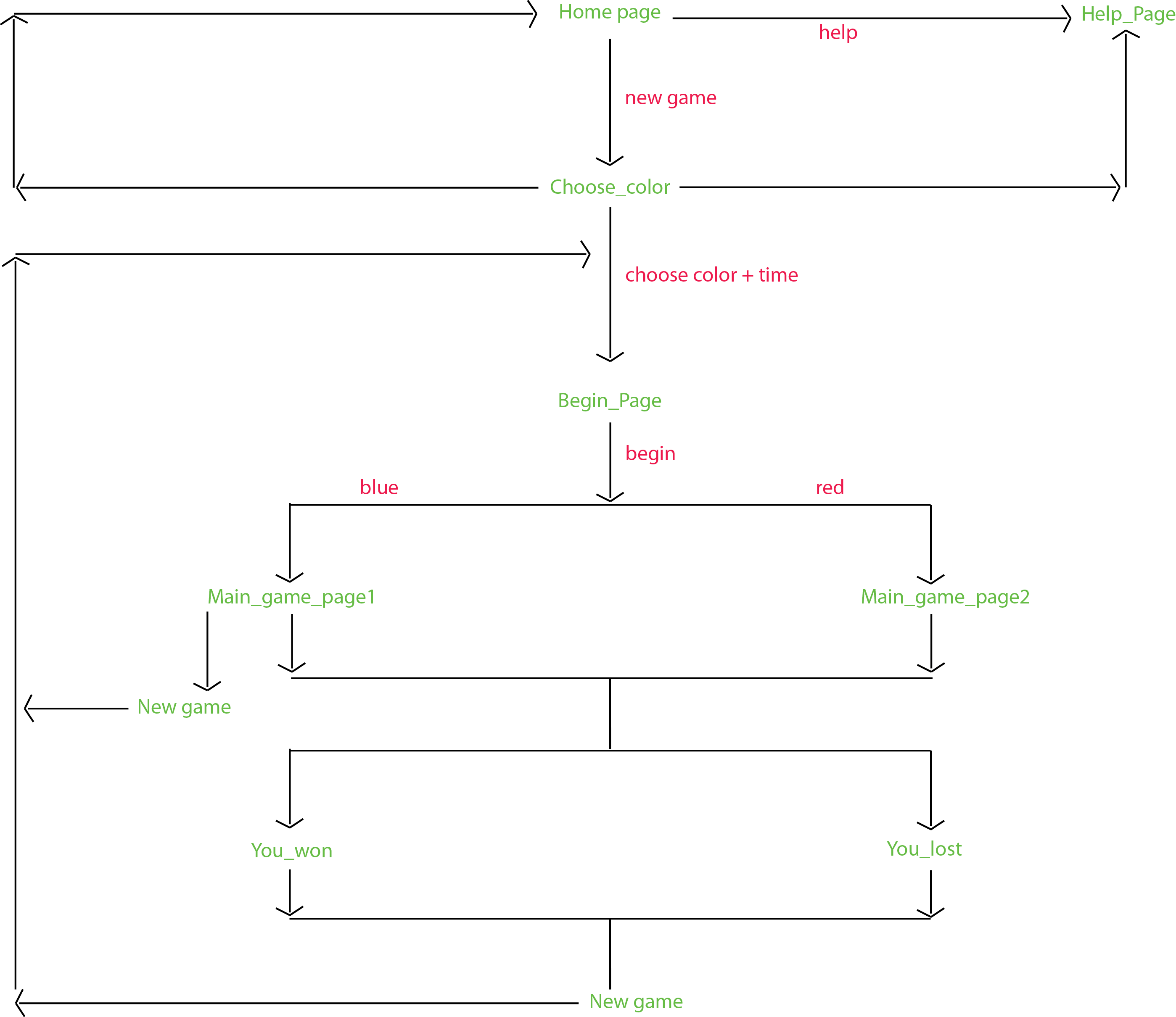


Figure 2 Flow diagram of the game

**2.2 Developing the graphics**

The graphics of these functions were prepared in Adobe illustrator and were saved as png images as jpegs would pixelate on enlargement and this was unfavorable for the end outcome.

**Importing the images into Director and adding behaviors**

Once the entire GUI was prepared, they were imported into the director and the pawns/circles were given behaviors.

The behaviors added to the graphics were:

* ObjectmoveablebyTL
* ObjecthidebyTL
* Objectsnaptocenterofsprite.

As the test leader is the person who actually would be operating the motion of the pawns, thus the wizard is given the option of hiding/un-hiding the pawn’s visibility depending on whether or not the test person could see the pawns or not. Thus the object hide behavior was added to cater to this requirement.

The object snap to center of sprite behavior was added to ensure that the pawns fell in the right sockets the moment because each time the test leader was dropping the pawns; there was a certain mismatch of alignment of the hole and the pawn.

Uploading the files on the Test Person’s computer

The interaction shells developed above were finally transferred to the Test Person’s computer through the FileUpdater.exe file.

**2.3 User Testing**

Finally the interaction shell was tested with the representatives of users. The test participants included Erica Nilsson, lab in-charge of Ozlab and four students of Karlstad University. In each test, the participant was the Test Person and the screens that were visible to him/her were controlled by the Test Leader.

**2.4 Key points gathered during first user testing**

The test person had difficulty in understanding how to put the pawns into the columns as there was no feedback or proper instruction regarding the same. So based on this feedback from the test person, proper instructions were given on each page by redesigning the GUI. Then the interaction was retested with three more users. This time, none of the users faced problems in interpreting the graphical data presented to them in the GUI as they all read the instructions which were presented in the form of a “news bulletin” form.

**Conclusion**

Various conclusions can be drawn from this assignment but here are some of the more important ones:

Usability of Ozlab setup is easy to learn

User interaction of Ozlab setup is limited leading to a lack of real time feedback for the user

Many ‘in-between’ software can be avoided to make the process more efficient

**3. Flaws in the existing method/setup of testing used:**

The wizard of Oz technique, as it is, is a very effective and usable tool to create and test prototypes. But there are some areas where the technique and the system can be improved upon such as:

* Test person’s interactivity with the application/scenario
* Test person’s authority on functions provided in the shell
* Lack of real time response
* Lack of transition effects in hi-fidelity prototypes
* Many ‘in-between’ software can be avoided to make the process more efficient
* Limited set of behaviors
* Little or no inheritance of functionality during import from other software
* Dependence on external software to make the shells and run the setup

**4. Recommended solutions:**

Because of the ‘Director’ problem, [As the director MX software is no longer in use since the acquisition of macromedia by adobe, it has become increasingly difficult to maintain the Ozlab setup and keep it updated with the changing requirements of the current technological requirements and keeping it compatible with other prototyping tools], there can be two routes or pathways by which this can be solved:

**4.1. Building a new dedicated platform from scratch: Possible recommendations**

**4.1.1** **Variables** This solution proposes that a dedicated ‘Ozlab’ software be built specifically to create the shells for the testing and the choice of coding language with which this platform shall be built depends on various variables like:

1. **Coding required**: The choice of language would depend on a huge extent on the amount of coding required to write and operate the platform. This also would include the amount of coding required to add new snippets and features into the platform for the shell once the platform has been built.
2. **Ease of modification**: Once templates have been done, they can easily be used to make low fidelity prototypes. But to make high fidelity prototypes, coding would be necessary as this initial base template cannot cater to the different sets of shells for various scenarios and contexts.
3. **Functionality which can be incorporated into the platform:** This is another important determinant in the choice making as we are moving into an era where the boundaries between desktop applications, mobile applications and various other modes are blending into one another and thus calls for dynamic software which can simulate these environments even in the prototype. Thus, the kinds of behaviors or functions that can be made using this platform would be vital for the growth of Ozlab as the user testing tool.
4. **Ease of building:** The ease with which the platform can be built would depend on what language is used and if the language used is a widely ‘known’ one, that is, the more the students are familiar with the language, the easier it is for them to grasp it and this is important because we are taking into consideration the face that it would be mostly students of Karlstad university from the departments of Computer Science and information Systems who would be working at the Ozlab for making the hi-fi prototypes.
5. **Compatibility with other software:** The platform should be able to export, import and be compatible with other software which are existing as this would be necessary to keep the ozlab setups ‘in working condition’ with respect to the changing technological platforms.
6. **Open source:** It is highly recommended to use open source software to build and operate the platform so as to avoid yet another case of ‘external dependence’.
7. **Popularity in terms of available theoretical material and online space:** Another variable would be the popularity of the software as in how many theoretical literature is there for someone new to learn the source code or how popular is this software in the online circles such as forums and how receptive are people to changes. Is the online forum active enough to post up snippets of self written codes which do different functions for different outputs?
8. **Updates:** How easily can this platform be updated according to external change requirements? Would the changes require a complete overhaul of the coding or is it possible to change only a specific set of code to make the desired modification?
9. **Ease of incorporating other languages with that language:** How easily can codes from different languages be put into this code of the platform? For example, a snippet of java code can be inserted into a html code.

**4.2. Using other existing prototyping tools:**

Prototyping for User Interfaces:

**4.2.1 What is prototyping?**

Prototyping literally means creating a mock up version of a thing before making the final product.

Prototyping is required whenever a new product is to be launched in the market. Car manufacturers, mobile phone manufacturers, etc do prototyping before introducing a new design. Similarly software companies also do prototyping whenever new software is launched. Sometimes a prototype is launched in the market for potential users to try. This version is called as the beta version of the software. This helps the company to get feedback from users. Any problems faced by the users are reported to the company which enables them to fix it before releasing the final version of the software.

**4.2.2 Why is prototyping needed?**

Just by looking at the designs of a product does not enable a designer to know how the product meets its purpose when put to use. Let us consider the case of a three dimensional product, say a car or to be more simple let’s take a pen. Though there are various 3D modeling software such as 3DS MAX which allow 3D representation of the designs of the pen on a 2D surface of a computer, adding a third dimension by making a prototype and testing it among users reveals questions such as how much comfortable the pen is to the user, how is the grip and other factors which cannot be revealed by just modeling on a software. Any product is after all made for the users to use. Prototyping takes care of this important aspect by revealing the difficulties faced by them which can be avoided in the final product. In addition to this, prototyping saves a lot of cost to the company by identifying the problems during the development stage of the product. Making changes to a product after it has been introduced in the market becomes very difficult and also needs a huge expenditure. It is because of these reasons that prototyping is now followed worldwide by all companies making a new product.

**4.2.2.1 Features and Variables that would determine the choice of the new software that would replace Director MX [specific to Ozlab]:**

**Low fidelity prototypes**: Low fidelity prototypes are used for creating instant wireframes and prototypes. These are used to decide the initial content before finalizing the actual design.

So, for creating this, the platform should include:

1. Wire framing Controls: Large number of sketchy wire framing controls.
2. Transition between pages: Screen based transition between pages through mouse clicks.
3. Drag and Drop controls: To allow for easy creation of wireframes.

For creation of high fidelity prototypes:

**High Fidelity prototypes**: These include wireframes and prototypes which resemble the actual design both in terms of GUI design and interactivity. There are various types of high fidelity prototypes which range from developing applications for mobile as well as web applications.

The features which limits director MX for creation of high fidelity prototypes and which should be included in our platform are:

**Behaviors**: Behaviors or components are used to add interactivity to objects. These are very few as of now in Director MX. Some of the widely used components are:

1. Button
2. Checkboxes.
3. Color Picker
4. Label
5. Text
6. UI Loader

**Tweens**: Tweens are used for creating motion graphics. These are widely used in mobile as well as web applications. So there should be an option for adding tween to objects on the screen. These include

1. Shape Tween
2. Motion Tween

**Transition Effects**: These include effects such as fading, easing, etc.

**Animations**: Various high level animations can be created in softwares such as Flash using actionscript. These are widely used for prototyping. So, there should be an option through which a user can create complex animation through coding.

**Import Options:** Director MX currently lacks import features of various visual design softwares such as Adobe Photoshop and Illustrator which are widely used the market. The platform should definitely include these options so that the visual designs created in these software can be straightaway be converted into prototypes.

**Wire framing Controls**: The number of readymade wire framing controls available in Director MX are very limited. Many wire framing software provide a large number of commonly used wire framing widgets used for creating mobile and web applications. These include:

1. Text
2. Sliders
3. Buttons
4. Image placeholder
5. Radio buttons
6. Sticky notes
7. Checkboxes

**Transition between pages:**  The only way to navigate between pages in Director MX is through the timeline or through the “change shell “method during usability testing. This is very cumbersome. A page based display of various screens for different pages would be very much useful and efficient.

**Export Options**: The export options currently available in Director MX are .BMP, .MOV and .AVI. Export options determine how much the software is compatible with other software and largely influence its usability. The various formats used for exporting nowadays are:

**Images**: JPEG, PNG, TIFF, BMP.

**Videos**: MOV, AVI, SWF. **Files**: PDF.

**4.2.3 Tools for prototyping**

There are many tools in the market used for prototyping, some of the more popular ones being Axure, Microsoft Visio, Flash Catalyst, Balsamiq, Flash Professional, Microsoft Expression and GUI Design Studio. Each of these tools specializes in some aspect and for a particular type of project selecting tools which are best suited for it makes the work of a designer much easier and also convenient. For example, catalyst does not allow animation but flash or Microsoft Expression does. Thus to a designer whose main aim is to accomplish the task at hand(prototyping), a list of the positive aspects of the various tools available would allow him to decide quickly which tool to select without delving too much in the details of others.

**4.2.4 Low fidelity prototypes:**

These are made to decide the content of the prototype without focusing too much on the design. These include sketchy and rough drawings which can be made easily and quickly.

|  |  |  |
| --- | --- | --- |
| **Aspect** | **Low interaction** | **High Interaction** |
| **Tool:**  **Starting Point:**  **Wireframing:**  **Interaction:**  **Coding Requirement:**  **Reusability**  **(Wireframes):**  **Export Options:**  **Suitable for:**  **Useful features:** | Balsamiq.  From Scratch.  Yes.  Mouse based interactions.  None.  No.  PNG, PDF document.  Web + Mobile.  1.Sketchy Widgets,  2. Drag and drop feature. | Microsoft Expression Sketchflow.  From Scratch.  Yes.  Transition effects + mouse based interactions.  None.    Yes.  Microsoft Word Document.  Web + Mobile.  1. Sketchflow Map.  2. Large number of controls.  3. Reusability of Wireframes to final prototype. |

During stakeholder interviews and meetings with the client initial ideas about the design are discussed. On the basis of these ideas, a lot of sketches are done which carry a rough summation of the elements that are needed on a particular screen. These sketches can be made on paper or sticky notes (known as storyboarding) or they can be made as low fidelity prototypes on a computer by using various tools such as Balsamiq and Microsoft Sketchflow. Low fidelity prototypes help us to get a rough overview of the design in its initial stages. The advantage of making low fidelity prototypes is that they can be made very easily and quickly. Moreover, their "sketchy" look keeps the focus of a designer on the larger level design decisions without getting distracted by subtle things such as alignment, size, colors etc.



**4.2.5 Medium fidelity prototypes:**

These include prototypes which do not exactly resemble the final deliverable but various aspects such as relative positions of content, etc are visible. These mainly include interactive wireframes.

|  |  |  |
| --- | --- | --- |
| **Aspect** | **Low interaction** | **High Interaction** |
| **Tool:**  **Starting Point:**  **Wireframing:**  **Interaction:**  **Coding Requirement:**  **Reusability**  **(Wireframes):**  **Export Options:**  **Suitable for:**  **Useful features:** | Visio.  Null.  Yes.  None.  None.  No.  PNG, JPEG, TIFF, Web page, AutoCAD Drawing, XML Stencil, XML Drawing.  Web + Mobile.  1. Large number of wireframing controls. | Axure.  Null.  Yes.  Mouse based interaction and transitions but no animation and transition effects.  Little coding required.  No.  As Image.  Web + Mobile.  1. Large number of wireframing controls.  2.Annotations and Interactions Panel. |

Medium fidelity prototypes are not exactly final but various aspects such as relative positions of content, etc are visible. In such prototypes, the actual content is represented by widgets. Images are represented by placeholders and text is replaced by the arbitrary Greeking “(Lorem Ipsum)”. Here the main focus is on the sizes and relative positions of the various elements on the screen but not on the actual content. Medium fidelity prototypes are close to wireframes. The advantage of medium fidelity prototypes over low fidelity prototypes is that the relative positions of the UI elements can be decided even though the actual content has not yet been finalised. In addition to this, wireframes can also be interactive depicting the navigation as they would be in the final design. The scope and variety of interactions possible depends on which software we use for wire framing. These wireframes can be shown to the client and various changes can be made based on the feedback received.



**4.2.6 High fidelity prototypes:**

These resemble the final design both in terms of visual design and interactivity.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Aspect** | **Low interaction** | **High Interaction** | | |
| **Tool:**  **Starting Point:**  **Wireframing:**  **Interaction:**  **Coding Requirement:**  **Reusability**  **(Wireframes):**  **Export Options:**  **Suitable for:**  **Useful features:** | Flash Catalyst  Visual Designs(Photoshop or Illustrator).  Yes.  Mouse based interactions.  None.  No.  PNG, PDF document.  Web + Mobile.  1. Transition Effects (fading, easing).  2. Heads Up Display.  3. Import Option of PSD and AI files. | GUI Design Studio  From Scratch.  Yes.  All kinds of MS windows based interactions.  None.  Yes.  Bitmap, PNG, JPEG and GIF.  Windows based applications.  1. Large number of Microsoft Windows based controls. | Microsoft Expression.  From scratch or Visual Designs.  Yes.  All kinds of animations + transition effects.  Yes (Visual Basic or C#).  Yes.  Microsoft Word document.  Advanced prototyping (Mobile + Web).  1. Events Panel.  2. User Control.  3. Reusability of wireframes. | Flash Professional.  From scratch or Visual Designs.  No.  All kinds of animations + transition effects.  Yes (Actionscript).  Yes.  Image + Movie.  Advanced prototyping (Mobile + Web).  1. Timeline.  2. Tweening. |

High fidelity prototypes resemble the final design both in terms of visual design and interactivity. Making high fidelity prototypes require making the visual design first and then adding interactions to it by converting them into buttons, links, etc. Various software generally used are Flash Catalyst, GUI Design Studio, Microsoft Expression, etc. Some software like Microsoft Expression and Flash Professional are advanced enough to create the visual design themselves. But others such as Flash Catalyst have import options which allow import of visual designs created in Photoshop and Illustrator and only interactions need to be added to them. Many of these software (Microsoft Expression, GUI Design Studio) have wire framing capabilities and allow reusability of wireframes. This means that wireframes created in the software can be converted to a high fidelity prototype by replacing the widgets with their corresponding visual design element.





4.2.7 Mind map of different software options

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | **Axure** | **Catalyst** | **Balsamiq** | **Microsoft Expression** | **Flash** | **GUI Design Studio** |
| **5. A Morphological chart of the analysis** | | | | | | | | |
| **Aspects** | | | **Axure** | **Catalyst** | **Balsamiq** | **M.S Expression blend** | **Flash** | **GUI design studio** |
| **Interaction** | | **Animation** | Fake animations can be created. | Can be done only through coding by opening in Flash Builder. | Not possible | All kinds of animations possible | All kinds of animations possible | Not possible. |
| **Transition** | Click based transitions possible but no transition effects like fading,easing. | Both transition between screens and transition effects possible. | Not possible | All kinds of transitions and transition effects possible. | All kinds of transitions and transition effects possible. | Click based transitions possible but no transition effects like fading,easing. |
| **Coding Involved** | | | No coding involved. | No coding required to make prototypes.Though coding is generated in the background and can be edited by opening in Flash Builder.  Language:XML | No coding involved. | Without coding, the variety of prototypes that can be made is possible but scope is limited.  Coding allows us to build any type of prototype according to our wish.  Language:  Visual Basic or C#. | Coding is required.  Language: Actionscript 3.0 | No coding involved. |
| **Creating from Visual Designs:** | | | Not possible.  Reason: No import options of Photoshop/ Illustrator files. | Import options of Photoshop and Illustrator files allows creation of prototypes from VDs. | Not possible.  Reason: No import options of Photoshop/ Illustrator files. | Import options of Photoshop and Illustrator files allows creation of prototypes from VDs. | Import options of Photoshop and Illustrator files allows creation of prototypes from VDs. | Not possible.  Reason: No import options of Photoshop/ Illustrator files. |
| **Fidelity of prototypes possible** | **High fidelity** | | Not possible since it has no import option of visual design software files such as Photoshop/Illustrator. | Import options of Photoshop and Illustrator files allow creation of high fidelity prototypes provided there is no animation involved. | Not possible. | Possible since visual designs can be created in the software itself or can be imported from Photoshop and Illustrator. | Import options of Adobe’s Visual Design softwares(Illustrator, Photoshop , InDesign) allows creation of high fidelity prototypes. | High fidelity prototypes for Microsoft Windows based applications can be made. |
| **Medium fidelity** | | Medium fidelity prototypes showing basic interactions are possible. | Possible. | Not possible. | Possible. | Possible. | Possible. |
| **Low fidelity** | | Not possible. | Not possible. | This software is purposefully made for creation of only “sketchy” low fidelity wireframes and prototypes that can be made quickly. | Microsoft Expression comes with a Sketchflow application that allows us to quickly make low fidelity mockups. | Not possible. | Not possible. |
| **Applications Suitable:** | | | Wireframing:  Both mobile and web based applications.  Prototyping:  Prototypes having mouse over, mouse-click based interactions can be created with this software. | Wireframing:  Both mobile and web based applications  Prototyping:  High fidelity prototypes having mouse over, mouse-click based interactions and transition effects such as fading, easing can be created with this software. | Used for creating rough initial mockups for all types of applications which is basically done to finalise the “content” before making the wireframes on more advanced softwares like axure, viso or flash catalyst.  Prototyping:  Click based rough prototypes can be created. | Wireframing:  Very suitable for creating highly interactive wireframes for all types of applications.  Prototyping:  All kinds of prototypes can be in this Microsoft expression.  Most suitable when creation of high fidelity and interactive prototypes are needed to be shown to client. | Wireframing:  Not possible.  Prototyping:  All kinds of prototypes can be created but this involves a huge amount of coding. | Wireframing:  Exclusively used for creation of Microsoft Windows applications.  Prototyping:  Highly recommended for windows based applications. |
| **Reusability:** | | | Wireframes and prototypes cannot be converted to final designs. | Wireframes and prototypes cannot be converted to final designs. | Wireframes and prototypes cannot be converted to final designs. | Wireframes and prototypes can be reused as they can be converted to final designs. | Wireframes and prototypes can be reused as they can be converted to final designs. | Wireframes and prototypes can be reused as they can be converted to final designs. |
| **Export Options:** | | | Axure files can be exported as Image. | Catalyst files can be exported as HTML, SWF. | Balsamiq files can be exported as PNG image and PDF document. | Expression Blend files can be exported as Microsoft Word documents. | Flash files can be exported as Image and Movie. | GUI Design Studio files can be exported as Bitmap, PNG, JPEG and GIF files. |

And thus, based on the above analysis, here is a chart of of the most suitable software for different contexts:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Aspects** | **Axure** | **Catalyst** | **Balsamiq** | **Expression Blend** | **Flash** | **GUI Design Studio** | **Visio** |
| Fidelity | Medium-High | High | Low | High | High | Med-High | Med-High |
| Interaction Level | Medium | Medium | Medium | High | High | Medium | Medium |
| Starting Point | Null | Null, Visual Design | Null | Null, Visual Design | Null | Null | Null |
| Coding | Null | Null | Null | Medium | High | Null | Null |
| Wireframing Controls | Many | Many | Many | Many | Few | Many | Many |
| Export Options | Image | Html, swf | Image, PDF | MS Word document | Image, Movie | Image | Image |

**6. Conclusion**

In this report we focused on the shortcomings in the current usability testing method used by Ozlab and came up with possible solutions that could be used to enable Ozlab to efficiently carry out usability testing for all types of prototypes. In the first solution we came up with a new platform for Ozlab to be built in an open source language. This would cater to the problem which has severely affected the use of Director MX after Macromedia was bought by Adobe in 2005. Building the platform in an open source language will ensure that the system is unaffected by mergers between companies or shunning down of a “company owned” language in the near future.

As a result of a large number of prototyping tools currently available in the market, often designers remain confused as to which tool to use for making a particular type of prototype. In the second solution we have analyzed some of the popular wireframing and prototyping tools currently used by designers and developers throughout the world. Various pros and cons of the softwares were taken into account in terms of the amount of coding and the level of prototype that it supports (low or high fidelity prototype). Based on this a chart was prepared to help a novice designer to quickly decide just by looking at the chart the prototyping tool that would be best fit to the kind of prototype that he or she has to make.

**References**

Brundin, A. (2011) “Possibilities on incorporating Adobe: Photoshop as a shell builder for Ozlab”. Report from course ISGC03 April 2011

Lamberg, C. (2011) “HTML5 for Ozlab”. Report from course ISGC03 Future web standards and mobile multimedia. April 2011, Karlstad University

Pettersson, J.S. (2002) Visualising interactive graphics design for testing with users. *Digital Creativity* Vol. 13: 144-156.

Rubin, J. & Chisnell, D. (2008). *Handbook of Usability Testing - how to plan, design, and conduct effective tests*. Indianapolis: Wiley Publishing, Inc.

Siponen, J. Pettersson, J.S. & Alsbjer, C. (2002). ”Ozlab Handhavandemanual”. Karlstad: Institutionen för informationsteknologi Informatik/Centrum för HumanIT, Karlstads Universitet

Pettersson, J.S. (2001) Presentation av idéerna bakom Ozlab. [Electronic]. Available: http://www.is.kau.se/~jsp/ozlab/n.php?goto=Swe/Presentation\_av\_ideerna\_bakom\_Ozlab\_01 0701.html [2011-03-24]

Cagan, M. (2008). High-Fidelity Prototypes. [Electronic]. Available: http://www.svproduct.com/high-fidelity-prototypes/ [2011-02-28]

Links:

<http://blogs.sitepoint.com/tools-prototyping-wireframing/>

<http://c2.com/cgi/wiki?GuiPrototypingTools>

<http://www.1stwebdesigner.com/design/wireframing-mockup-prototyping-tools-plan-designs/>

<http://mashable.com/2010/07/15/wireframing-tools/>

<http://www.adaptivepath.com/ideas/rapid-prototyping-tools>

<http://en.wikipedia.org/wiki/Website_wireframe>

<http://www.dexodesign.com/2008/11/07/review-16-user-interface-prototyping-tools/>