Evaluation of mendix as a rapid application development (rad) tool

Technology, Development

1 Introduction

1. 1 Overview of RAD and RAD Tools

The computers play a very important role in our lives as all the things that were done by people are now being replaced by computers. Initially all the data of an organization or a firm used to be stored in a tedious manner using papers which were very difficult to maintain but due to the fast improvement in field of computers everything is now becoming computerised. The generation of Database System was like a boon to the developers as it made storage of the huge amount of data very easy. Slowly everything started becoming computerised and then evolved the need for application development. For developing an application a specific method has to be followed so that the project can be build up on time and meet the requirements. Then evolved conceptual methodologies such as System Development Lifecycle, waterfall model, spiral method, linear sequential, etc. and were used for the step by step development of project.

Due to the advancement in thetechnologymany new methodologies have developed. Rapid Application Development was developed as a response to the non-agile methodologies like Waterfall Model. The projects using this methodology took long time to build, as a result of what the requirements used to get changed at the end of the project [1]. Though, waterfall model is still being used and still is widespread among the developers, new technologies have been developed like Rapid Application Development, Agile Methodology, etc. A Rapid Application Development approach makes the use of CASE (Computer Assisted Software Engineering tools) and prototyping

that describes the procedures to speed the development of the software. The RAD method was developed by James Martin in 1991[2]. It took time for RAD to develop so much but now many tools have developed which use RAD methodology for the development of quick designs of the proposed project. Small scale projects can be developed using RAD tools as it is still not much reliable to be used for large projects because of the automatic code generation the requirements of the project may get over looked. It uses the RAD model which is a combination of linear sequential model (mentioned later in the report in Methodology section) and prototyping model which allows the development team to generate a fully functional system within a very short time. The RAD method may use three procedures:

Prototyping: In this process a quick design of the software is created and presented to the business or eventual users for refinement. This refinement is done on the bases of feedback given from the user [3].

Iteration: Iteration and prototype go hand in hand. Iteration is the commitment of applying the refinements as soon as possible [3].

Timeboxing: This is a management procedure that focuses more on delivery above all [3].

The different methodologies will be explained further in detail in the Methodology section of the report.

I have seen my friends facing problems while developing and maintaining database in complex systems, that gave an idea of such software that can

create and maintain the database on its own without getting into trouble and RAD tools was the best solution for this. I knew some about the RAD methodology and the tools used as I have been using tools like NetBeans, Visual Studio and Visual Basic about which I have mentioned later in the report. My supervisor encouraged and guided me for developing my knowledge in this area. He introduced Mendix to me for the research and for understanding its working properly. This was the seed of an idea for choosing Evaluation of Mendix as my project topic. It was a new approach for me to fully analyse software like Mendix and comparing it with other better RAD tools. Throughout my project I will develop a Payroll application that will be developed using the main features of Mendix.

1. 2 Project Aims

Review RAD tools

In this section I will give detailed information about RAD tools, improvements in the RAD industry and the ways it has being used from the past.

Use Mendix

This section will discuss the reasons why I chose Mendix for research and detailed study on the features provided by this software that makes it easy to develop small scale applications.

Evaluate Mendix

Here I will evaluate Mendix by comparing its basic facilities with other tools by providing advantages and disadvantages of using this software.

Recommend situations where it can be used

In the end, after the full study of Mendix, I will give recommendations on what type of applications this software can develop.

1. 3 Personal Aims

Developing knowledge in RAD tools.

Try to explore Mendix as much as possible.

To efficiently develop a conclusive report.

Offer recommendations for the improvement of Mendix.

1. 4 History of project

I had read about the RAD development methodology and the tools using that methodology. It is very easy way of creating the business applications in a very short duration of time. The applications developed using this methodology are often developed by compromising with the usability, feature and sometimes execution speed [4]. Various tools have been created that use the CASE (Computer Aided Software Engineering) tools for the rapid development of the applications. These tools are basically used for creating the instant designs of the information system that is to be created so that the basic scenario of the project can be cleared. As described in the section 2. 1, RAD methodology comprises of a lot of prototyping, iterative customization and timeboxing through which the basic working of the project can be specified [4].

I found this methodology very interesting as instant implications of the ideas can be done. Also the project completion time can be approximately 60-90

days using RAD tools [4]. Firstly I read about Mendix and other tools. Then I experimented with Mendix for understanding the basic functionalities provided. I found it quite easy as everything that I used to do using long process was done easily in a very user friendly interface. Moreover things like creating a database design in the form of Domain model was done just by dragging and dropping the entities and there was no need to develop and maintain the database tables for storing data. These basic features interested me more and I decide to evaluate Mendix for my Bsc. (Hons.) Computing Final Year Project.

1. 5 Overview of report

Section 2 will be containing the methodology I will be using for my project. Section 3 will give the overview and brief information about RAD tools and also brief review of Mendix and Lightswitch, two RAD tools I am going to perform research on. In section 4, I will give complete description of Mendix including the functionalities and facilities of this RAD tool. Followed by this will come the analysis of payroll system that I am going to develop using Mendix as a part of my research. According to the used methodology I will give information on the design and implementation of the application. The full evaluation on Mendix will be performed and I will give recommendations concerning Mendix. the last section of the report will contain the conclusion concerning the project.

2 Methodology

Project development is a step by step process. All the tasks must be divided into several different parts so that development flow is maintained (reference book). There are many methodologies like Waterfall, Agile, RAD, Prototyping, Spiral, etc. using which developing a project becomes easier and quicker and these are also known as Software Development Process Models [5].

Waterfall Methodology:

Waterfall methodology is the most basic and frequently used methodology. Waterfall approach was the first process model to be introduced and followed in software engineering [5]. “ Many new methodologies have evolved that follow the basic steps of waterfall model but use different modelling approaches at different times” (reference book). Waterfall model follows simple procedure of SDLC (System Development Life Cycle) i. e. planning, analysis, design and implementation. Following is the figure of the waterfall approach:

FIGURE 3-1Waterfall Model

This process model is called waterfall because it moves from phase to phase same as a waterfall (reference book). Next phase starts only when the previous phase is completed. The flow is to be maintained exactly same as shown in the figure 3-1. Once a particular phase is finished we cannot take the project back to the previous phase. Following are the phases of the waterfall approach:

Requirements gathering and analysis: In this phase requirements of the proposed system are gathered from the end user so that the main goal of the software is understood correctly. Requirement is a set of functionalities that the user expects from the system [5]. Requirements are gathered by consulting the end user and understanding the actual desires of the user from the system. This gathered information is analysed for their validity and also some other possibilities are suggested to the end user for their satisfaction from the system. Finally, a requirement specification document is created that states the guidelines for using the next phase [5].

System Design: Before starting to write the code it is necessary to have a complete understanding of the system to be developed [5]. For this purpose the system design is developed according to the requirement specification document prepared in the previous phase so that the complete knowledge about the system hardware and the functionalities is obtained [5].

Implementation: Once the system design is received the work is divided into different units for the development of code [5]. This work is divided in different modules and is allotted to the development team. The requirements are first developed as small units and then integrated to form a full system. These units are tested while they are being created to check the functioning of the different units of the application. As soon as the coding of the system is completed it is forwarded to the testing team.

Testing: After the coding of the system is completed it is integrated and forwarded to the testing team for testing that how the system works as a

whole and whether it meets the requirements of the client. In large systems the testing phase may be sub-divided into different parts like alpha testing, beta testing, etc. so that the requirements are fulfilled completely.

Deployment of System: This phase includes the delivery of the system to the client a, installation of the developed system on the client system and training to the client.

Maintenance: This is the last phase of waterfall model and can be considered as the longest phase as its end time is not fixed [5]. The problems with the system that were not discovered during the development are discovered in this phase after the deployment and are solved again for its working properly [5].

According to me there are some disadvantages of using waterfall model:

As all the requirements of the system cannot be gathered at once, it is possible that the desired product may not be created.
This model is called Waterfall model because it follows a continuous flow in one direction so the problems with one phase if not solved during that phase itself, it becomes very difficult to go back to the phase and solve it. This may result in a badly structured system [5].
Client requirements always go on adding to the list [5]. This results in a completely unusable system and client dissatisfaction.
The main disadvantage of Waterfall model is that there is very less clientcommunicationand till the end client does not have the working model of the system, so the client remains ignorant of the progress of the system

and remains unaware of what is being developed is as per his requirements or not.

As there are some limitations in using waterfall methodology, it cannot be the only methodology that should be used in the systems development, so I am going to use some other methodology along with waterfall to overcome these curbs.

According to Rapid Application Development (RAD) based methodologies, special techniques and computer tools should be used to speed up the analysis, design, and implementation phases of the system development. Tools like CASE (Computer Aided Software Development Tools), Joint Application Design (JAD), fourth-generation programming languages and automatic code generators should be used for the quick completion of the system. But due to speedy development of the system, the user requirements might change dramatically. For handling this problem there are three RAD based methodologies using which the systems can be developed as per the user requirements even if the requirements keep on changing (reference book).

Phased Development:

“ A phased development-based methodology breaks an overall system into a series of versions, which are developed sequentially (reference book).” The system requirements are quickly identified in the analysis phase that leads to the design and implementation phase of the system but only with the basic and fundamental requirements identified for the version 1 of the

system. Working on version 2 starts after the implementation of version 1 is completely done. In the version 2 again analysis is performed that combines the requirements of version 1 and some new ideas and issues that arose from the users in version 1. According to the analysis performed, designing and implementation are performed and immediately work begins for the next version. In this way system is developed quickly and everytime the requirements are refined. The only drawback of using this methodology is that everytime user is presented an incomplete system intentionally so it becomes difficult to identify the important features of the system at the end.

Prototyping Methodology:

A prototype means a typical example that is used for the later stages. The main aim of prototyping model is to counter the first two limitations of waterfall model. In waterfall model the Requirements gathering and analysis stage is limited till its completion but in prototyping a quick design is created by taking the recently known requirements into consideration. This methodology contains a frequent communication with the client so product can be developed completely according to the client requirements and it doesn’t turn out to be unusable as mentioned in one of the disadvantages of the waterfall model. In a simple prototyping model the Analysis, Design and Implementation stages continue to be developed till it doesn’t match the client requirements.

FIGURE 3-2Prototyping Model

Once the requirements for the system are specified a prototyping design is created and implemented on a very small scale with minimal features and is shown to the client. This gives an idea to the client that how the system will works after its completion. The cycle of creating a prototype and presenting it to the client goes on till the system is fully developed. Each time a new sample is developed, it is in an improvised form than that the previous one. The key advantage of prototyping model is that it assures the client that the development team is working on the system and user can be well informed about the unrealistic requirements.

The main disadvantage of this model is that in the process of creating prototypes and refining the requests every time, the actual/ initial requirements of the system might be forgotten till the end of the project. This results in the loss of small but important specifications that might be useful for making the system more functional. This drawback can cause problems in developing complex systems. Throwaway prototyping approach is similar to prototyping but is presented in a different way to overcome the drawbacks of prototyping approach.

Throwaway Prototyping:

“ Throwaway prototyping approach is similar to prototyping methodologies, however prototypes are done at different stage of SDLC (System Development Life Cycle)(reference of book)”. In prototyping, analysis, design and implemetation stages are used in the prototyping cycle as a result of what the requirements get changed due to frequent gathering of

information, but throwaway prototyping based methodologies have a relatively thorough analysis phasethat completely specify the system requirements so that the basic functionalities suggested by the client can be implemented. The issues identified in the analysis stage are examined in every possible way by analysing, designing and implementing the design prototype. Once the issues have been identified and resolved the design prototypes are thrown away. The system developed using this methodology are based on several designs prototypes during the design ang implementation phase. The key benefit of using this methodology is that all the risk factors are identified before the fully functional system is developed. Complex systems can be built using this methodology which cannot be efficiently done using prototyping methodology.

FIGURE 3-3Throwaway Prototyping Model

Phased development-based methodology is one of the important categories of RAD based methodologies but because of its drawback as mentioned above, important features of the system might be overlooked at the end of development, it cannot be used in my project. Throwaway prototyping is a nice and useful approach for developing huge and complex systems but as I am not developing a large scale system using Mendix I will be using prototyping along with the Waterfall model but I would like to use throwaway prototyping methodology in my future projects.

3 Overview RAD Tools

RAD model is gaining popularity these days because of the flexibility in the time taken for the completion of the business applications. Many tools have been developed for these purposes which work on different operating systems. The first generation of RAD tools provided a high level programming language used for fetching stored data from the database of which dBase II is the best example [7]. dBase II was not very successful but proved to be beneficial for small organizations [7].

According to the humanpsychology, working can be joyful if the surroundings are favourable and because of this mind-set GUI (Graphical User Interface) Systems became popular and then arouse the need for the RAD tools that can work appropriately. Tools that could generate code automatically in backend by producing graphical notations in front end where developed and Microsoft Visual Basic (on Windows) and HyperCard (in MacOS) is its best example [7]. In both of these tools applications can be created by just dragging and dropping the required fields of the forms and associating them with an object oriented code.

Linux has been derived from UNIX and is said to be less user-friendly as compared to Windows but RAD tools have also been developed for Linux operating system using high languages like Perl, Tcl& Python, and 4G Languages [7].

There are two types of RAD tools, Non-graphical and Graphical. Linux based RAD tools used VHLL’s (Very High Level Languages) [7].

Non-graphical:

The first language used was Perl (Programmable Extraction and Report Language) which was developed by Larry Wall in 1987. Followed by Perl, some other languages like TCL (Tool Command Language), Python and Ruby were also developed. These languages cannot be considered fully as RAD tools as they don’t provide graphical interface builders [7].

Graphical:

Tk is a graphical toolkit which was originally developed for TCL (Tool Command Language) but can also be used for Perl and Python [7]. Tk is originally a graphics library that provides commands for designing interface easily which can work on windows, MacOs as well as Linux operating system. Tcl/Tk is free software that comes with Linux [7].

All the tools mentioned above still cannot be thought of as full RAD tools as a combination of a high level language and a graphical toolkit cannot make what is called as RAD tool [7]. After these advancements, people started understanding the need of RAD tools and understood that the development of business applications can be much easier. So tools such as Visual Tcl, SpecTcl which provide a very bright and colourful user interface where developed [7].

These are the basic and the initial tools developed after which many other RAD tools where developed. Now we have a huge variety of RAD tools available which can be used efficiently and amongst them I will be choosing one tool for the research.

3. 1 Important RAD Tools

As the RAD methodology has developed many tools have been developed for implementing this model and producing the applications rapidly. Tools like IBM Rational Application Developer, NetBeans, LightSwitch, Wavemaker, Mendix, etc. are the famous RAD tools [8]. Microsoft Visual Studio is also a kind of RAD tool that provides a developmentenvironmentthat is much simpler.

NetBeans is a RAD IDE (Integrated Development Environment), generally used for visual desktop, mobile, web application using languages like Java, Ruby, C/C++, PHP, JavaScript [8].

LightSwitch is a new RAD tool developed by Microsoft Visual Studio which can be said as the further version of Microsoft Visual Studio 2010 [9].

Wavemaker is a RAD tool using which developing web and Java applications can be developed and deployed on multi-tenant servers easily [10].

Joomla is a very famous Content Management System used for developing powerful web applications [11].

Content Management System: A content management system is software that keeps track of every piece of content on your Web site. This content can be of any kind text, images, videos, documents, etc [11].

3. 2 Mendix Review

Mendix can develop the business applications 5 times faster and at ? the cost of other software development platforms [12]. The applications developed can be deployed on the Mendix cloud and also on the local firewall [12]. It is new software with different set of functionalities for

developing applications. Creating a Domain model is the most important phase in developing an application that connects to the database as it signifies visibly that how the data will be maintained in the database. A domain model can be generated effortlessly by just dragging and dropping the entities. The prime advantage of using Mendix is that the database is maintained by itself. The Mendix website is very helpful for the new users to be able to use the software efficiently. After creating an account on the Mendix website, a trial version of the software can be downloaded, which works for a month, allows you to have a rough overview of the software, also the application can be deployed on the cloud. Cloud computing (reference) is a new trend in the IT (Information Technology). In cloud computing, the software and hardware services that are being used sitting on your desktop or in your company’s network ares provided by any other company over the Internet. The users using the services have no concern about who is providing the service and where software and hardware are located; it is somewhere located on the “ Cloud” for the users [13]. Similarly, the applications prepared in Mendix can be deployed on the cloud very easily.

The other facility provided is that various apps are available for free on the website that can be included in the projects that you create (throughout the website applications are referred as apps and I will also be using app for denoting an application). There are about 24 apps available on the website by the Mendix Projects that help us to use Mendix more efficiently. There are five different categories of the apps in an apps store; the academy section provides the readymade projects that can be referred before starting the

development of our own project because they offer an example of the basic functionalities delivered by Mendix.

3. 2. 1 Examples from the app store:

The first app in the academy section is Wizard (with 5 steps) example project v1. 0, this is the simple example that accepts the data by the user through 5 dialog boxes, these dialog boxes are basically prepared by using input forms, and stores them in the database. This is a simple wizard that follows a series of steps.

The other interesting app is My First App; it is a small application for storing data regarding the former as well as present employees along with their images. This application includesGoogleMaps widget, again provided by the Mendix app store. The image below shows the home form of My First App, at present there is one employee in the database and the Google Map shows the location of the address provided by the employee. The Faces tab shows the snaps of the employees. The Departments tab shows all the departments in the present organisation and the last Reports tab automatically generates the charts of Member Distribution and Department Size.

The following figure shows the generated charts according to the employees stored in the database.

There are some widgets available on the website for free usage of what may enhance the application that we are creating.

Time pickercan be used to make the date time field more functional. It allows to select time from a two-directional dropdown and the monotonous job of

entering the time manually is simplified.

A help text viewerallows you to add help buttons in the forms for the new users, which enhances the forms and make it more usable.

TheCountdowncalculates days, hours, minutes and seconds between the current and specified time.

Mendix also provide themes that change the appearance of the application when it is deployed and run on the browser. The business component section provides apps like Google Maps Module, about which I mentioned before, Excel importer, Account Management Module, etc. which make application building very easy.

Apart from the app store, demo videos are also available on the Mendix website that explains the basic steps for developing applications and gives a rough gist of the way Mendix works. The videos are very helpful to the new users who are confused about the starting point of the project.

3. 3 Lightswitch review

(Reference) Lightswitch can be considered as the further version of Microsoft Visual Studio 2010. It has been released as the beta version and will be releasing as a full version after the beta testing is done. It is an RAD tool for the development of high-quality business applications for desktop as well as cloud. The free beta 2 version of this software is available on the Visual Studio Lightswitch website which can be downloaded easily. Beta testing is the test prior to releasing a computer product in market. Beta test is the last stage of testing and is often performed by either sending the product to the

beta test sites outside the organisation or releasing the product online for free trial download so that it can be downloaded and can be improved based on the feedback of people (reference). Lightswitch is a high level and fascinating software as it is developed by Microsoft which is renowned for its perfect products. There are also videos on the website for the quick understanding of the software.

Visual Studio Lightswitch beta 2 can quickly built custom line of business applications. Software should have an appealing IDE (Integrated Development Environment) that makes it easier to work comfortably. Visual Studio Lightswitch beta 2 is successful in providing a good development interface. It provides a collection of preconfigured screen templates which can be used for preforming tasks such as entering data and displaying and formatting information in a variety of user friendly layouts. Application themes and skins can be downloaded for free that gives professional look and feel to the application. One of the most tiresome jobs in developing any application or a website is to put validations on the type of data to be entered by the user. Lightswitch provides built in functionalities to validate the input from the user. Normally, the common snag with using RAD tools is that the scope of the application is restricted up to the built in functionalities provided by the software, this results in dissatisfaction of the user. In Lightswitch, the functionalities can be expanded by using C# or Visual Basic code.

Lightswitch applications can combine data from a wide selection of sources including Sql server, Microsoft Sql azure, Microsoft Sharepoint server 2010

and Microsoft . net and many 3rd party data sources. Data can be easily exported to Microsoft Office Excel 2010 for data sharing. Lightswitch uses Microsoft Silverlight technology so applications can be deployed locally as well as on the web server. But due to security context, exporting to excel is restricted to the applications running locally. The new cool feature of Lightswitch beta 2 is that applications can be deployed on cloud by using the publish application wizard.

Lightswitch provides a readymade example of Contoso TRADers. Contoso TRADers is a dummy company name used by Microsoft for examples. In the example as per shown in one of the videos on the Microsoft website, Contoso tRADers are the manufacturers and distributors of mountain bikes and its equipments around the world. They have built a simple customer relationship management application to keep track of the products and orders placed by the customers. The application isbuilt using simple 2- tier architecture that enables to practice through the data fluently. A 2 – tier architecture is where the client talks directly to the server without any intervening server. It is typically used in small environments (that is under 50 clients) (reference).

In the present example the application is running locally on the desktop storing all the data that it needs in the Sql server database in any other part of the organization. Contoso TRADers are expanding their business all over the world and so they want to convert the present application into a web based application so that the orders can be managed online. This can be done by deploying the application over 3 – tier with the server elements

running as web application over the IIS. IIS (Internet Information Server) (reference) is a group of internet provided by Microsoft Windows NT. It is generally used for running web based applications (reference). A 3-tier architecture is a well-defined client/server architecture which works on three stages, all working separately (reference).

The User Interface that runs on the client computer.
The middle tier or Application Server is a functional module that processes the data.
The Database Management System that store the data provided by the application server. This normally works on a separate server called Database Server.

This architecture enables the application to run from almost anywhere. Moreover, Lightswitch beta 2 enables users

3. 4 Common Features of RAD Tools

Development of RAD tools has reduced the leg-work in programming. There are some common features of RAD tools that are almost available in all the respective tools. Following is the list of some of the common features of RAD tools (reference):

RAD tools help in developing quick and dirty examples i. e. prototypes that do not exactly demonstrate the full working of the application but fulfil most of the customer requirements.
Adding new requirements to the project while it is evolving results in a chaotic atmosphere which becomes easy in RAD tools using the CASE

(Computer Aided Software Engineering) tools.

Without writing any code applications can be developed in a user friendly atmosphere.

Often the client is not clear about the requirements unless he/she sees a functional example of the application. RAD tools are very useful in creating such examples.

The feature of reusability of code helped including Object Oriented languages in RAD tools.

Some more features.

3. 4. 1 CASE (Computer Aided Software Engineering) tools

CASE tools can be defined as " These are the tools that provide leverage in the software requirements analysis and design specification phases, as well as those tools which generate code automatically from the software design specification." (Reference of book)

4 Description of Mendix

Mendix is a newly developed RAD tool that is based on Java platform for the application development. A general RAD tool would make it easy to prepare quick designs of the specification, in the similar way you can create a design quickly using the features of Mendix. Mendix provides a wide range of components from which the required components can be easily included by just a click. Readymade forms can be added quickly which is very less time consuming. You can also insert new data into the database as well as update or delete the existing data.

The unique feature provided by Mendix is the use of microflows for handling the data as per our wish. Microflows are pictorial presentation of the code that could be used for the desired actions and events. You can use all the basic features of a general programming language like looping, start/end events, data retrieve, etc. and everything is handled through Java code at the backend. All you have to do is select the Java event and place it in the microflow and you see that automatically things are handled. There are many other interesting features of Mendix which I will explain in the later part of my report. Mendix can be appropriately used for prototyping because if the client is unhappy with the working of the provided prototype, the entire application can be rebuilt very tranquilly to match the user’s requirement specification. Also it’s easy to add new functionalities to the project if the requirements evolve in the middle of the development.

According to me if the working environment is soothing, half of the work becomes easy and Mendix is little successful in doing this. The look and feel of Mendix is pleasant and less complicated so the person who is using Mendix for the first time can understand the purpose of the software, though the appearance of Mendix is not that attractive and interactive as compared to Microsoft Visual Studio Lightswitch beta 2. If I am asked to compare the IDE of Mendix and Lightswitch and make a choice between them, I would prefer using Lightswitch even though it is more complicated than Mendix. The reason for this choice is that Lightswitch is a product by Microsoft Corporation, one of the leading software companies. Microsoft has set a trend of extremely user friendly environment in its softwares with much

functionality for example, Microsoft Word. Basically Microsoft Word was developed for text based files and it’s formatting, but along with these, features like saving the document in multiple formats, inserting tables and images, performing small calculations like summation, etc., are also incorporated. In spite of so many features majority of people don’t find Microsoft word complicated and are always ready to accept newer versions of it. That is all, I think is because of the powerful presentation services used by Microsoft. For newly developed software like Mendix, the present interface is nice.

4. 1 Features of Mendix

Mendix provide many new features that allow you to develop applications effortlessly. As mentioned in one of the above sections, creation of domain model is very simple in Mendix. Domain model, also called as Domain Object Model (DOM) is a conceptual design of the system involving various entities and showing its relationship with other entities (reference). It is basically used for developing and handling database of the system. Using Mendix it is just some clicks away. Entities can be simply dragged and dropped into the working area for a domain model. Its attributes can be set by double clicking on the entity where you can set the entity name and add as many attributes as you want along with their datatypes and constraints. There is even no code required for placing validation rules on the attributes. All you have to do is to select the attribute name and select validation type from required, unique, equals, range, regular expression or maximum length. It is so easy to validate your application just by selecting from the list of validations. A

generalization relationship can also be established by selecting the name of desired parent entity. A generalization relationship is “ a relationship in which one model element (the child) is based on another model element (the parent). Generalization relationships are used in class, component, deployment, and use-case diagrams to indicate that the child receives all of the attributes, operations, and relationships that are defined in the parent (Reference)”. Moreover relationships between entities can be established by dragging and dropping the selection from one entity to other the same way as it is done in MS Access. Multiplicity of relationships that it is one-to-one, one-to-many or many-to-many can be specified by clicking on the line connecting two entities and selecting the desired cardinality. Same as MS Access, a specific attribute/field can be given primary key by providing the validation Unique to the field. The data can be fetched in two ways, from a database or from a microflow.

The second step that is to be followed after the domain model is created is to prepare forms to provide functionalities in the app. In the project explorer section, when you right click on the name of your project and select the ADD option, it gives a huge list of components that can be added to make your application more usable and attractive. New forms can be added quickly in which different functionalities are easy to include. You can control the way your data appears in your forms by inserting table, data grids, data views etc. The properties of all these objects are managed from the properties pane on the right side of the working area.

Data Grid: A data grid displays the objects of the selected entity in the form of grid. All you have to do is place data grid in the form and connect it to an entity from the connector pane in the right side of the form. The attributed of the entity will automatically fill in the grid. Through the datagrid you can even search, add or edit the objects.

Data View: A data view is basically used for viewing the details of exactly one entity object. Generally the New or Edit button on the data grid opens a form containing data view that shows the details of the object that is already selected in the data grid. A data view comprehends a table like structure that contains input options like text boxes. The Data Source properties of a data view indicate that which object will be shown in the data view. Data can be obtained in two ways in a data view, from an entity or a microflow that opens a form.

Template grid: It is exactly same as the data grid but the only difference is that it displays the objects of entity in a tile view.

Table: the layout of the form can be changed by using table. Several rows and columns can be added and desired objects or buttons can be placed inside a table. In short we can create our own form by using table.

Tab control: A tab control can be used when the information that is to be displayed is more than the screen space. Each tab page contains different information.

Horizontal split pane: A horizontal split pane divides the region into two parts horizontally. A horizontal split pane can be used again with in any of the (horizontally or vertically) divided regions.

Vertical split pane: A vertical split pane is same as a horizontal split pane, but the only difference is that it divides the region into two parts vertically. It can be used again in any of the (horizontally or vertically) divided regions.

Textbox: A textbox is used to display or add a value.

Text Area: A text area can be used to display or add a long text value that may be appear in many lines. For example, for inputting address, a text area can be used.

Dropdown: A dropdown can be used for displaying the enumeration value.

Checkbox: A checkbox is a small box that appears with a tick or without a tick. It is generally used for editing a truth value. A checkbox must be connected to the attribute of type Boolean. When the checkbox is ticked, the value is 1 i. e. true and when it is unchecked, the value is 0 i. e. false.

Date picker: A date picker is used to display or edit date value. It should be associated with the object having Date datatype. It shows a localized calendar from which you can pick any date.

There are other form elements as well like:

Reference selector

External link

File manager

Image uploader

Image viewer

Microflow trigger: used to trigger an already created microflow.

Mendix also provides a report widget that helps in automatic report generation.

No code is required for connecting the forms to the database; this is again handled in a simple way by selecting the desired entity that has already been created while producing the domain model and the form fields are automatically filled up. Most of the work can be done without writing code, but when complex calculations are to be done, microflows come in action. As stated above, a microflow is a pictorial presentation of the code that can be used for performing desired actions. Microflows can be considered same as the flow charts that are used at the early stage of programming for making the programmers familiar with the programming logic, even though the microflows are much more complex and difficult that the simple flow charts. Same as flow charts, microflows also have a start and end event between which the whole process of programming is handled. Below is the example of a simple microflow that is taken from the examples available on the Mendix website.

In this example, microflow is created to show a form called Department_Show that returns a true (Boolean) value. So whenever this microflow is called, it shows the Department_Show form and returns the

value true. Here, Department is used as a parameter. A parameter is a data that serves as input for a microflow (reference).

This was the most basic example of a microflow being used. More intricate and bigger applications can be developed by making use of the activities provided by Mendix. These activities are divided into 6 types: Object, List, Action call, Variable, Client and Integration. The object activity can be used for directly performing actions on the entities by creating their objects. The object activities can be used for creating, changing, retrieving, casting and deleting the objects.

List operations are used for handling list of the data of an object rather than a single value for an object. An XPath query can be used in the list functions. An XPath is a type of query language used by Mendix for retrieving data. XPath query can be written by specifying the path expressions that select the data of Mendix entities and its attributes and associations (reference).

Action call activities can be used for calling any previously created Java action or calling a microflow in another microflow.

Variable activities are used to perform tasks on variable; the variables can be created as well as changed as per the requirement of the microflow.

Client activities refer to the actions performed on forms.

“ Mendix OQL (Object Query Language) is a relational query language just like SQL. The major advantage of OQL is that OQL uses entity and association names instead of the actual database table names (reference).”

It uses the clauses that are used SQL queries like SELECT, FROM, WHERE, etc.

5 Analysis of Payroll Application

For understanding the working of Mendix it was necessary to choose a topic developing which I could experiment with Mendix features. So, for the full evaluation of Mendix I decided to develop a Payroll application that will use and understand all the possible features provided by Mendix. I used to work part time in a shop in my home country (India). In the shop the wages of the employees were handled through a payroll system. I was impressed by the working of the system. When I got a chance to create a working application using Mendix the first clue that hit me was to develop a payroll application. This is how I decided to develop a payroll application for my research.

Payroll System: The term payroll refers to the details of a company’s employee that receives wages and other rewards regularly. Some of the employees may get wages on monthly basis while others may get based on hours worked daily or weekly. Using the computerized payroll system the wages can be generated by calculating the bonuses or allowances and even deductions like Provident Fund, medical insurance, charitable contributions, loans, taxes, etc. A payroll application is used by a payroll specialist and pay checks are generated (reference). The sign in and sign out time can also be kept track by making use of a payroll application. The employers keep constant tally on the details of the employees and all the data is maintained.

The basic requirement of a payroll system is that it collects the hour details of an employee and calculates gross wages, subtract all the deductions and print checks. A payroll application can also be used to maintain the records of company’s retirement accounts.

Payroll software does the critical calculations just by taking a small input from the user i. e. employee’s pay amount and the total hours worked. The salary is calculated on its own without taking any pain. In higher level payroll softwares, automatic updates are received for the taxes every time the law is changed. Computerized payroll software reduces the tedious work that used to be done in handling the employee details manually and it also saves huge amount time.

As mentioned above in the Methodology section of my report, among the RAD based Methodologies (phased development, prototyping and throwaway prototyping) I am going to use Prototyping. Therefore I’m going to do a first domain model, implement the most important functions and evaluate the prototype. After the evaluation of the prototype I will refine the requirements and develop a second prototype. This process will be followed till the whole system is developed fully.

6 Design

After the analysis I came to know the basic requirements of a payroll system. The basic details that need to be stored in the database are the employee’s personal details, job details, payment details, deduction and timecard

details. The first thing to be done for the first prototype was to create a domain model. The first domain model was as follows:

In the domain model-1, I used 8 entities: emp, job, time_card, deduction, payment_schedule, payment_method, bonus and payment. This is the very first domain model I created when I started using Mendix. I was learning to use the features properly and was confused regarding the relationships that should be established between the entities. I have used Microsoft Access before as a database for my application and according to the rule for establishing relationship between tables in MS Access, each table should have a primary key and a foreign key then only relationship can be established. A foreign key is the primary key of other table with which relationship has to be established. So, initially I was working on this principle, that’s why I included the fields of on table in another as foreign key for the relationship but then I discovered that this not the case with Mendix. Mendix does not work on the MS Access principles; we don’t need to include the fields of one table in another for establishing a relationship, it can be done directly. The type of relation can also be selected i. e. one to one, many to many or one to many by clicking on the line joining the two entities. For the system to work properly, it is necessary that the relations between the entities are correct and sensible. I was not able to decide relations between some of the entities and I found the model impractical so I decided to develop a new improved version of the domain model in which I decided to remove some of the entities that were of no use in my application. The new version of my domain model is as follows:

In the version 2 of my domain model I removed payment_schedule, payment_method and bonus entities because I found them useless looking at the requirement of the system at that time. I needed to retrieve the employee details like employee's first name, middle name and last name and display them in the payment table so that it becomes easy to store and calculate the payment details of the desired employee. For this I used microflows that would fetch the details of employee's first name, middle name and the last name and view them in the payment entity. Following is the microflow that I used to retrieve the employee's first name from the database:

In the above microflow, a parameter of the Payment object with the variable name Payment is used for input for the Payment object. Whenever the microflow is triggered, it retrieves the employee details from the Employee entity which is saved in the variable called Employee_fname and returns the emp_fname attribute from the Payment entity. I developed the same kind of microflows for retrieving other details of employee and tried to implement this practically in the forms but this didn't worked the way I wanted. I consulted to Mr Jim Longstaff about this problem and he guided me to use the reference selectors in the forms through which we can directly fetch data from the desired entity without using microflows. For this we do not need the attributes of other entities in our desired entity, it can be directly fetched from the referenced entity. For this the only thing you need is the relationship between the two entities.

I discovered the problem with the above microflow when I understood the working of microflows in the later stage of my project. To fetch the details from Employee table, I should have taken object Employee instead of Payment and the output should have been of type $Employee/emp_fname. But there was no need of using microflows for this purpose now. To implement the reference selectors instead of microflows I had to develop another version of the domain model. So I created a 3rd domain model as follows:

After creating the version 3 of domain model I developed the forms and used reference selectors in the form fields as suggested by Mr Jim Longstaff which fulfilled my requirement. Again I discovered a problem in the present domain model, the relationships between Employee – Payment, Employee – Job and Job – Payment were one-to-one which was not realistic and should have been a one-to-many relationships and needed improvement. Also I decided to remove the attributes bonus_amt and gross_pay from the Payment entity because calculating bonus amount and gross pay would be difficult and it was possible that it would have required more time for understanding the full functionality and then implementing it. Thus, I created a last and final domain model as follows:

In the domain model version 4 I included validations on the fields of entities like emp_id of the Employee entity was given required, that means if that field has not been entered an error message may occur (if specified) and gave a validation on the max length of the phone attribute i. e. phone number should be of maximum 11 digits. The Job_Title attribute of entity Job

and date_time attribute of TimeCard were given the validation Required. Also the type of relationship was changed to one – to – many from one – to – one. I have given float datatype to all the time related attributes as it would be easy to calculate a float entry rather than a time entry. I have set a single value as a pay amount for each employee i. e. ? 4. 50 as calculating wages for employees of different levels would have made my application more complex to develop. After the creation of this final domain model I started implementing these in the forms for making my application functional.

7 Implementation

Implementation in my application was intended to develop forms that would include the entities created already in the domain model. The first form that I made was EmpDetails form that contained a grid that would show all the details of all the employees.

The above form can display the details of all the employees all together. It is necessary to specify another form through which you can edit and add new employee details on the new and edit buttons of the grid otherwise it shows errors. The form that opens by clicking on New or Edit buttons looks like a simple form that contains labels, textboxes and buttons.

Here the field Job Title in the EmpNewEdit as well as EmpDetails form is a reference selector that fetches data form the Employee_Job association. An interesting thing to notice here is that there is the same form for adding new employee details as well as to edit the existing details. This is because data grid has to be used for both the purposes so if you want to add new detail in

the database you can just click on the New button and a fresh form will open and if you want to edit a particular employees details, select the employees details first and click on the Edit button, an already filled in form will appear. This is so simple. Similarly other forms were created.

Forms related to entity Job:

The above form can be used to add job details at runtime.

Forms related to entity Payment:

Forms related to entity TimeCard:

I created an enumeration called Weekday which acts as an attribute for the TimeCard entity through which the day can be selected from the list of seven days while entering timecard details for an employee.

Enumeration:

After implementing all this I wanted my application to look more attractive and easy to use. I wanted to display all the details of an employee together in different tables on the same page. That could be possible using the Listen Target property of a data view. For displaying three different grids on the same page and connect them with each other Horizontal and Vertical Panes were used. This form was named EmpOverview and became the home form of my application. In this form when an employee is selected, all the details like payment and timecard details related to that employee automatically start appearing in other two tables. The new EmpOverview form is as follows:

This is how the above form appears after running the application:

Above is the home form showing three tables together. In the present scenario, there are only three employees in the database, so selecting the employee named Amyn Khoja in the employee table displays the Payment and TimeCard details of the same employee. The navigation bar at the top of the Employee Overview form contains four tabs through which a user can navigate between different forms.

8 Evaluation of using Mendix

Mendix has many impressive features but the main feature that I found most difficult to understand was the use of microflows. I tried to eliminate the use of microflows as much as possible but for handling the main calculations that were supposed to be carried out automatically, microflows were the only feature using which this could be possible. The calculation of total hours per day and the per day wages of employees was supposed to be handled through microflows.

The first thing I wanted to calculate was the total hours an employee worked on a particular day. Before calculating the total hours worked in a day, hours worked before and after meal were necessary to be calculated. For this I created two microflows. The first microflow was to calculate total hours before meal.

This microflow takes the TimeCard as an input and returns the total_beforeMeal attribute. On the onChange event, total_beforeMeal is calculated by using XPath query:

$TimeCard/timeOut_beforeMeal – $TimeCard/timeIn_beforeMeal

The above query subtracts the value of timeIn_beforeMeal with timeOut_beforeMeal and stores the result in total_beforeMeal attribute as shown in the figure below.

The second microflow calculated total hours after meal.

Same as the above microflow, this microflow takes the TimeCard as an input and returns the total_afterMeal attribute. On the onChange event, total_afterMeal is calculated by using XPath query:

$TimeCard/timeOut_afterMeal – $TimeCard/timeIn_afterMeal

The above query subtracts the value of timeIn_afterMeal with timeOut_afterMeal and stores the result in total_afterMeal attribute as shown in the figure below.

After the microflows for the calculation of total hours before and after meal are done, another microflow has to be created that takes the value of total hours before meal and after meal and adds both the values. The microflow below does the same:

The above microflow takes TimeCard as an input and on the onChange event it calculates the total hours by adding the values of total_beforeMeal and total_afterMeal attributes through this XPath query:

$TimeCard/total_beforeMeal + $TimeCard/total_afterMeal

After the microflows had been developed, these microflows had to be applied on the form fields so that they can be triggered to carry out calculations. For this purpose on the onChange event of Time before meal 2 field and on onEnter event of Total before meal field microflow TC_calBeforeMeal is triggered. Similarly, on the onChange event of Time after meal 2 field and on onEnter event of Total after meal field microflow TC_calAfterMeal is triggered. On the onEnter event of Total Hours field TC_CalTotalHours microflow is triggered.

The example below shows how the calculations are handled through the microflows when we run the application on localhost: 8080:

Here when you create a new TimeCard for an employee the total hours are automatically calculated after entering the details forTime Before Meal1&Time Before Meal2andTime After Meal1&Time After Meal2.

After the TimeCard calculations started working properly the other complex calculations were the calculations for Payment entity. For calculating the total payment amount of an employee it is necessary to fetch the total hours worked by an employee from the TimeCard entity. For this I developed a microflow that retrieves all the data from the Payment_TimeCard association

through Retrieve event and returns the total_hours from the TimeCard. Following is the microflow PaymentTotalHours:

The microflow for calculating the wages of an employee is as follows:

The above microflow takes Payment as an input. In the first activity PaymentTotalHours microflow is called because it returns the value of attribute total_hours from the TimeCard entity. An exclusive split has been used which works on the following condition:

if$TotalHours= 8thentrueelsefalse

If the above condition is true, a Change object event is triggered and net_pay is calculated and the event is ended that returns the value of net_pay:

But if the above If condition becomes false another exclusive split is used that checks the condition for the deduction amount:

if$Payment/Deduction= 0thentrueelsefalse

In the query above, if the condition becomes true, net_pay is calculated and event is ended but if the condition becomes false, net_pay is calculated by subtracting the deduction value from net_pay:

So far I was able to do what I wanted to do using microflows. I needed to include more features and wanted to experiment more with microflows but due to the lack of time I was unable to do that.

The next functionality that I wanted to include in my application is the monthly report generation for the employees. This was a difficult task because I consumed maximum time in understanding the working of microflows so there was no time left to understand the working of Report widget. But I found another feature as an alternative to generating reports and it isExport to Excelbutton in data grid. Using this button all the details of the data grid can be exported to Microsoft Excel and can be printed out from there. I have used this button in the main data grids so that a kind of report can be generated.

9 Recommendations concerning Mendix

Mendix is wonderful software and defines the RAD tools properly up to some extent. Even though it has been recognised very quickly and has been in the top lists of IT awards, according to me it requires many improvements that will make it easier to use. I think that the first thing that should be improved is its look and feel. I didn’t find the present IDE so user friendly that would make me work more to explore it. The other thing I found annoying about Mendix is that sometimes while running the application in the browser, the application becomes very slow and tasks are not performed properly. As compared to Microsoft Visual Studio LightSwitch, Mendix needs many improvements to be in competition with LightSwitch.

Mendix can be efficiently used for creating prototypes as large projects cannot be developed because of the limitation of the use of code. Large and complex systems like Banking, HR Management systems for huge multi-

national companies, Stock Exchange systems, railways and Aircraft Management systems, etc. need more reliable softwares. While experimenting, using and evaluating Mendix I found that it is difficult to develop large applications because for developing huge systems, very complex coding is required for fulfilling the requirements of the system, but Mendix provides a very limited set of coding that would not even be useful for developing half of such systems. So Mendix can be used to develop the prototype of such systems for presenting it to the client and then the prototype can be handed over to the developing team for developing the system using proper code.

10 Conclusions

10. 1 A Project Summary

The main objective of my project was to evaluate Mendix Business Modeler and compare it with other RAD tool. While evaluating Mendix I was supposed to develop a functional application that would use all the possible features of Mendix. The main feature of Mendix is Microflow and I was successful in using and understanding the events that are used in a microflow for carrying out calculations which I think now I am capable of doing. The Payroll system application that I have developed is a very basic level application that performs basic tasks. While evaluating Mendix I performed research on Microsoft Visual Studio LightSwitch and understood its basic features.

10. 2 Future Developments

I have developed a basic version of my Payroll Application because I took maximum time in learning the features of Mendix, therefore there are many functiona