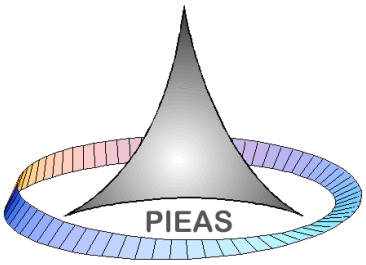
GUI Development for RELAP-5

By

Waleed Ahmed Malik

Report submitted to the faculty of Engineering at PIEAS in partial fulfillment of requirements for the Degree of MS Nuclear Engineering



Department of Nuclear Engineering

Pakistan Institute of Engineering & Applied Sciences

Nilore, Islamabad, Pakistan

October, 2013


بسم الله الرحمن الرحيم

Department of Nuclear Engineering,
Pakistan Institute of Engineering and Applied Sciences (PIEAS)
Nilore. Islamabad 45650, Pakistan

Declaration of Originality

I hereby declare that the work contained in this report and the intellectual content of this report are the product of my own work. This report has not been previously published in any form nor does it contain any verbatim of the published resources which could be treated as infringement of the international copyright law.

I also declare that I do understand the terms ‘copyright’ and ‘plagiarism,’ and that in case of any copyright violation or plagiarism found in this work, I will be held fully responsible of the consequences of any such violation.

Signature: _____

Name: Waleed Ahmed Malik

Date: 21/10/2013

Place: PIEAS, Islamabad

Certificate of Approval

This is to certify that the work contained in this thesis entitled

"GUI Development for RELAP-5"

was carried out by

Waleed Ahmed Malik

Under our supervision and that in our opinion, it is fully adequate, in scope and quality, for the degree of M.Sc. Nuclear Engineering from Pakistan Institute of Engineering and Applied Sciences (PIEAS).

Approved By:

Signature:_____

Supervisor: ***Dr. Imran Rafique Chughtai***

Signature: _____
Co-Supervisor: ***Dr. Muhammad Ilyas***

Verified By:

Signature: _____
Head, Department of Nuclear Engineering
Stamp:

Dedication

Dedicated to my striving Country,

Islamic Republic of Pakistan

Acknowledgement

Gratitude and endless thanks to Allah Almighty, the Lord of the World, who bestowed mankind, the light of knowledge through laurels of perception, learning and reasoning, in the way of searching, inquiring and finding the ultimate truth. To whom we serve, and to whom we pray for help.

I feel my privilege and honor to express my sincere gratitude to my supervisor Dr. Imran Rafique Chughtai and Co supervisior Dr. Muhammad Ilyas for their kind help, guidance, suggestions and support through the development of this project. With due respect, I would also like to thank project panel and coordinator for useful discussions.

I would like to express my most sincere gratitude and thanks to my beloved Parents, Wife Sarah, team member Afnan and all others who were with me in any aspect, for their overnice support and motivations during the work of this thesis.

Finally, I would also like to thank Pakistan Institute of Engineering and Applied Sciences and ACRE (Advance Computational Reactor Engineering) Lab for providing very conducive educational environment.

Waleed Ahmed Malik

Table of Contents

List of Figures

Abstract

Nuclear Reactor safety analysis is a systematic study to demonstrate the limits and reliability of reactor in normal as well as accidental conditions. Reactor Excursion and Leak Analysis Program (RELAP) or RELAP5 is a tool for analyzing small-break LOCAs and system transients in PWRs or BWRs. It has the capability to model thermal-hydraulic phenomena in 1-D volumes. It is used for the thermal- hydraulic transient analysis in water-cooled nuclear reactors by solving one dimensional, two-phase thermal-hydraulic equations in an arbitrarily connected system of volumes. However, the preparation of the input file and subsequent analysis of results in this code is a tiresome task.

The aim of this work was to develop a Graphical User Interface (GUI) for preparation of the input file for RELAP5. The GUI has been developed in VB.NET using Microsoft Visual Studio 2010. The whole task of GUI development was divided into three sections namely “Development of GUI Environment”, “Interfacing of Thermal-Hydraulic components" and “Interfacing of Neutronic Components”.

This thesis deals with the development of the GUI Environment named as RELAP5 Input File Generator (RIFGen). The base classes for the development of further RELAP5 Components have been designed and implemented. The major programming logic and layout of all forms is described in detail in the thesis. This GUI generates complete set of cards for Hydrodynamic Components, Core Components, Time Step Control, Minor Edit Requests, Trip Input Data, Heat Structure Input, General Table Data, Plot Request data, Control System Data, Additional Plot variables, General Code Input, Couple Control Cards and other Miscellaneous Cards. These components have been tested and validated individually. RIFGen has been validated as a whole by generating input files for several standard problems.

1. Introduction

Reactor safety analysis is an analytical study to demonstrate the limits and integrity of reactor in normal as well as accidental conditions. There are many codes available for reactor safety study but RELAP5 is the most widely used systems analysis code. There are numerous references available in the public literature describing the application of the RELAP5 to a variety of problems. It is developed by the U.S. Nuclear Regulatory Commission (NRC) for use in rulemaking, licensing audit calculations, evaluation of operator guidelines, and as a basis for the nuclear plant analyzer. Although RELAP5 was originally developed to support the analysis of postulated accidents in commercial power plants in the United States, different versions of the code have been widely distributed around the world and now are used to support a wide range of activities.

In recent years, RELAP5 has been applied most extensively to support the certification of advanced reactor designs as well as help resolve outstanding severe accidents issues. Both RELAP5 was important tools used by the USNRC to certify the expected performance of the AP600 passive reactor design while SCDAP/RELAP also played an important role in the resolution of concerns about high pressure failure of US reactor designs during postulated severe accident conditions. RELAP5 made important contributions to the reassessment of safety of Russian-designed reactors [1].

RELAP5 is also being widely used by regulatory and research organizations around the world to support international standard problem exercises and experimental programs. The impact of user experience, the ability of the code to predict thermal-hydraulic and severe accident phenomena, and its applicability to prototypic plant transient data have been extremely well characterized. Although plant data for accidental conditions is limited, the code has also been widely used to assess the performance of plants under design basis and severe accident conditions. In particular, RELAP5 was used extensively by the US Department of Energy and many international organizations to support the assessment of the TMI-2 accident.

1.1. Background and Context

RELAP5 input file is a text file of extension ".i". The system flow, components properties and all the initial conditions are described by specific card numbers. The card numbers are actually six to eight numeric numbers. For a single component these cards may range from tens to hundreds. A single problem comprising of three

components may have hundred cards therefore a user needs to remember or browse throughout the manual to write even a single card. This is a tedious job and time consuming as well. Moreover, if the solution is not converging, a lot of time and effort is required to modify this input file.

The complexity of the input file can be seen by just considering these few cards in input file. The user has to browse through the user manual of the RELAP5 to input all the properties on specific cards. If a single card is misplaces then RELAP5 does not read the input file, and user has to trace and correct the error. This tedious job can be well reduced by a helpful Graphical User Interface (GUI) for input file generation. The input file needs a single click to be generated. In the GUI the nodalization diagram is first drawn, and then the parameters of components and the initial conditions are input. This file is saved and can be modified easily. This type of GUI, on one hand, makes RELAP5 user friendly and on the other hand, saves times of input file generation and modification.

1.2. Scope and Objectives

An interface for user friendly model development for RELAP5 has been developed. This includes a graphical user interface for building a nodalization diagram in accordance with RELAP5 model and its consequent conversion into RELAP5 input file. The main objective of this project is to aid RELAP5 users in the writing of input file. In other words, it may serve as a helpful tool for RELAP5. Graphics have always been successful in doing tedious problems easily and efficiently. This part deals with development of the GUI for the above mentioned purpose. It is a drag and drop type diagram development facility having all features/components supported by RELAP5. It provides an interface to other parts of the project that translates every component into RELAP5 input file syntax

Ultimate Goal: “To enable users at PIEAS as well as other PAEC establishments to use RELAP5 in a hassle free and productive manner”

1.3. Literature Review

It is a known fact that input file generation and revision is tiresome task therefore many efforts are made to develop graphical user interface for RELAP5 by many peoples and organizations in different countries.

In 1999, Dr. George Mesina developed GUI for RELAP-3d in Idaho National engineering and environmental laboratory. He used some aspects of the older GUIs of

some other codes i.e. Athena Aid, TROPIC and SNAP to establish a new friendly GUI for 3d version of RELAP5. JAVA as programming language was used which is object oriented language. There are good visualization aspects of this GUI but as far as complexity of the input file is concerned, the user has to browse through the RELAP5 manual to use this software.[2]

In 2007, K.D. Kim from Korea and Rizwan-uddin from USA made a web based simulator using RELAP5 and LabVIEW. They used RELAP5 as the solver and LabVIEW for the graphical user interface and web casting. Their software reads the input file and shows the results graphically which can be operated remotely from another site connected to the server via the World Wide Web. But it does not have the facility to generate input file through graphical user interface. [3]

In 2008, two PIEAS students Abid Afsar Khan and Muhammad Muneeb Anwar made the RELAP5 GUI for mod 3.2. They used C Sharp (C#) as programming language. They developed initialization form, design surface, connectors, and many thermal-hydraulic components modeling, but their project lacked modularity which made further improvements difficult.[4]

All the GUIs for RELAP5 made so far are not available to use commercially so there was a need to develop a new RELAP-GUI. All the imperfections and short comings in the previously developed RELAP-GUI are taken into account and a new GUI is made with a more dynamic structure having the ability to extend.[5]

DWSIM is an open-source CAPE-OPEN compliant chemical process simulator for Windows, Linux and Mac. Built on the top of the Microsoft .NET 2.0 and Mono Platforms and featuring a rich Graphical User Interface (GUI) in VB.NET, DWSIM allows chemical engineering students and chemical engineers to better understand the behavior of their chemical systems by using rigorous thermodynamic and unit operations' models with no cost at all. Since DWSIM is open source, they can see how the calculations are actually being done by inspecting the code behind during execution using free tools available elsewhere.[6]

The best features of the DWSIM open-source software are re-used to develop the GUI for RELAP5. Features like Design Surface, Component Connectors, drag-and-drop of the components to the design surface, assigning properties to the components in property grid and save/load. These features helped to make the code more dynamic and user friendly.

1.4. Thesis Layout

The GUI environment, main form, the programming language and technologies used are explained briefly in chapter 2. In chapter 3, the core classes and the main architecture of the application is discussed in detail. Also component interfacing with GUI is described and all the necessary cards and properties for input file are illustrated. The appendix contains the User Manual and Installation guide for this RELAP5 Input File Generator (RIFGen).

2. Description of GUI Environment

The required GUI named as RELAP5 Input File Generator (RIFGen) takes the nodalization data, component properties and initial conditions as input of RELAP. The program block diagram is shown in Figure 2-1.

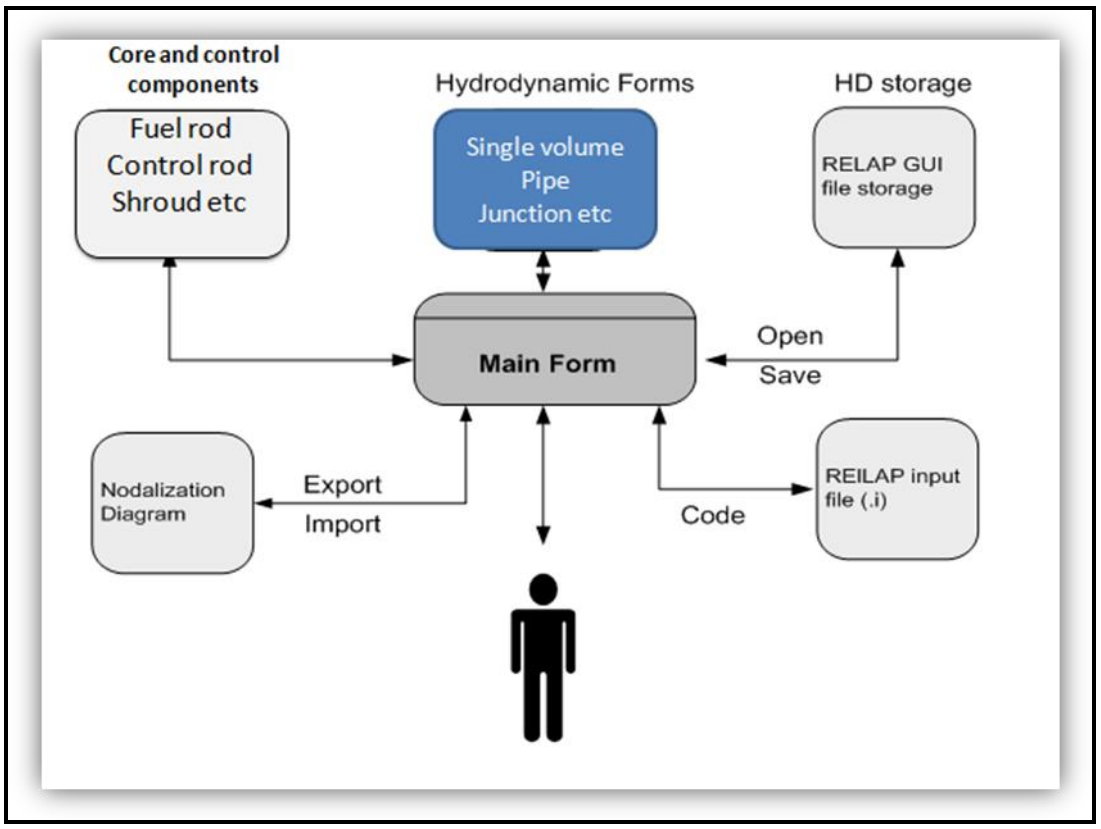


Figure 2-1: Block diagram of RELAP5 GUI development

Thermal-hydraulic components, neutronics and control systems, save and load of the file, nodalization diagram and input file generation is interfaced with the main form of the program.

2.1. Major Modules

The major modules used to develop the graphical user interface for RELAP5 are as under.

2.1.1. Design Surface / Flow sheet and Graphics Collection

Modeling of graphics for components requires the creation of a 2D graphic so that it can be rendered and manipulated digitally. Connection points are defined so that input and output connectors to different components are possible. This Design Surface is located in the center of the window as shown in Figure 2-2.

This is the core module that has been reused from DWSIM, it has the ability to

move around components, rotate them and to connect the components with each other. Components can be zoomed in or out. It uses the GDI+ API by Microsoft. Windows GDI+ is the subsystem of the Windows XP operating system or Windows Server 2003 that is responsible for displaying information on screens and printers. GDI+ is an API that is exposed through a set of C++ classes.[6]

2.1.2. Components List

The component list contains a list of all RELAP5 Components. These components are designed in a way that they can be added to RELAP5 in an incremental manner without disrupting the existing coding. Component coding task has been distributed among team members so that the application may be coded in parallel with optimal progress. The component list is located on the right side of the application as shown in Figure 2-2.

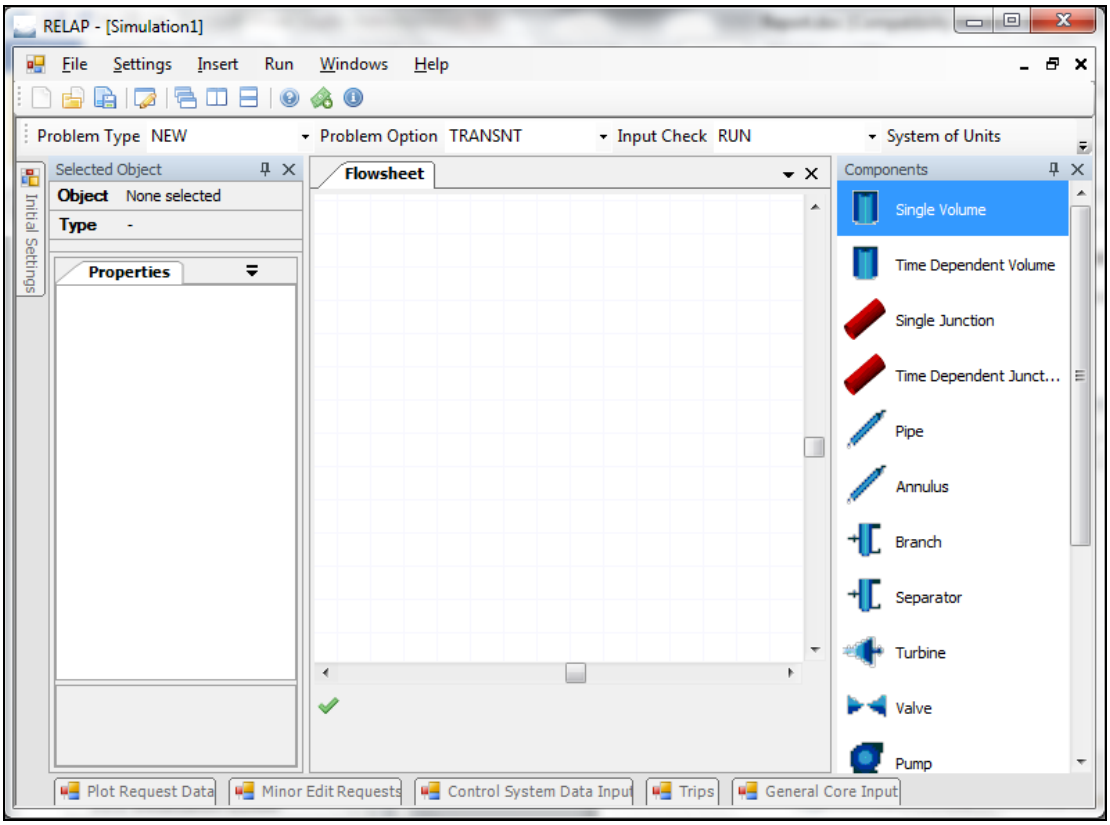


Figure 2-2: RIFGen Overview

2.1.3. Component Properties and Configuration Table

This list contains all properties of the component (Volume, Area, Wall Roughness etc), these properties are later used to create the Input file which is used by RELAP5 to perform calculations and simulations. The component properties table is located on the left of the main program as shown in Figure 2-2.

2.1.4. Input File Generator

The RELAP5 input file is written in accordance with the RELAP5 user manual. An output file stream writes the input file cards and subsequent words to a file with extension ".i". These words are fetched from the collection of the components stored in the parent form.

2.2. Technologies Used

Microsoft Visual Studio 2010 is the Integrated Development Environment used to develop RIFGen in which GDI+ technology is used. Team foundation server is the source control server to synchronize, develop and publish the code. Some external libraries are also used to assist in various tasks to make the GUI an even more friendly experience. The following technologies are used to develop the GUI for RELAP5.

2.2.1. Microsoft Visual Studio

RIFGen was developed using Microsoft Visual Studio, an integrated development environment (IDE) from Microsoft. It is used to develop console and graphical user interface applications along with Windows Forms or WPF applications, web sites, web applications, and web services in both native code together with managed code for all platforms supported by Microsoft Windows, Windows Mobile, Windows CE, .NET Framework, .NET Compact Framework and Microsoft Silverlight.

Visual Studio includes a code editor supporting IntelliSense as well as code refactoring. The integrated debugger works both as a source-level debugger and a machine-level debugger. Other built-in tools include a forms designer for building GUI applications, web designer, class designer, and database schema designer. It accepts plug-ins that enhance the functionality at almost every level including adding support for source-control systems and adding new toolsets like editors and visual designers for domain-specific languages or toolsets for other aspects of the software development lifecycle (like the Team Foundation Server client: Team Explorer).

Visual Studio supports different programming languages by means of language services, which allow the code editor and debugger to support nearly any programming language, provided a language-specific service exists. Built-in languages include C/C++ (via Visual C++), VB.NET (via Visual Basic .NET), C# (via Visual C#), and F# (as of Visual Studio 2010). Support for other languages such as M, Python, and Ruby among others is available via language services installed separately. It also supports XML/XSLT, HTML/XHTML, JavaScript and CSS.

Individual language-specific versions of Visual Studio also exist which provide more limited language services to the user: Microsoft Visual Basic, Visual J#, Visual C#, and Visual C++.[7]

2.2.2. GDI+

RIFGen was developed using GDI+ technology. As its name suggests, GDI+ is the successor to Windows Graphics Device Interface (GDI), the graphics device interface included with earlier versions of Windows. Windows XP or Windows Server 2003 supports GDI for compatibility with existing applications, but programmers of new applications should use GDI+ for all their graphics needs because GDI+ optimizes many of the capabilities of GDI and also provides additional features.

A graphics device interface, such as GDI+, allows application programmers to display information on a screen or printer without having to be concerned about the details of a particular display device. The application programmer makes calls to methods provided by GDI+ classes and those methods in turn make the appropriate calls to specific device drivers. GDI+ insulates the application from the graphics hardware, and it is this insulation that allows developers to create device-independent applications.

Our Major Focus is on Vector graphics, which involves drawing primitives (such as lines, curves, and figures) that are specified by sets of points on a coordinate system. For example, a straight line can be specified by its two endpoints, and a rectangle can be specified by a point giving the location of its upper-left corner and a pair of numbers giving its width and height. A simple path can be specified by an array of points to be connected by straight lines.

GDI+ provides classes that store information about the primitives themselves, classes that store information about how the primitives are to be drawn, and classes that actually do the drawing. For example, the ‘Rect’ class stores the location and size of a rectangle; the Pen class stores information about line color, line width, and line style; and the Graphics class has methods for drawing lines, rectangles, paths, and other figures. There are also several Brush classes that store information about how closed figures and paths are to be filled with colors or patterns.[8]

2.2.3. Team Foundation Server

RIFGen development was greatly accelerated in an organized manner using Team Foundation Server (TFS). It is a Microsoft product which provides for source code

management (either via Team Foundation Version Control or Git), reporting, requirements management, project management, automated builds, lab management, testing and release management capabilities. It covers the entire end-to-end software development process. TFS can be used as a back end to numerous integrated development environments but is designed to provide the most benefit by serving as the back end to Microsoft Visual Studio or Eclipse.

Source Code Control

Revision control, also known as version control and source control, is the management of changes to documents, computer programs, large web sites, and other collections of information. Changes are usually identified by a number or letter code, termed the "revision number", "revision level", or simply "revision". For example, an initial set of files is "revision 1". When the first change is made, the resulting set is "revision 2", and so on. Each revision is associated with a timestamp and the person making the change. Revisions can be compared, restored, and with some types of files, merged.

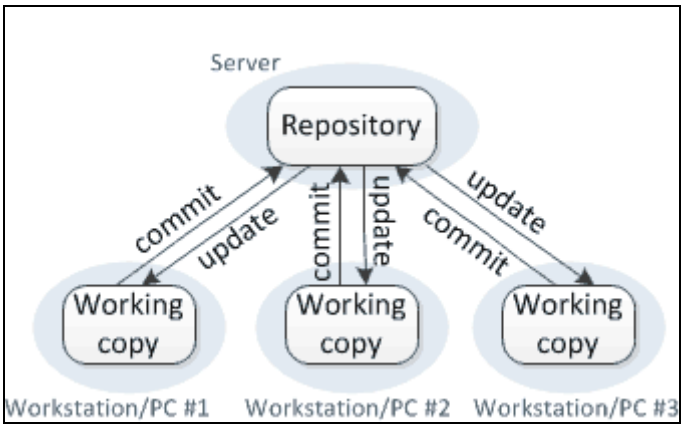


Figure 2-3: Centralized Version Control

In computer software engineering, revision control is any practice that tracks and provides control over changes to source code. Software developers sometimes use revision control software to maintain documentation and configuration files as well as source code.

As teams design, develop and deploy software, it is common for multiple versions of the same software to be deployed in different sites and for the software's developers to be working simultaneously on updates. Bugs or features of the software are often only present in certain versions (because of the fixing of some problems and the introduction of others as the program develops). Therefore, for the purposes of

locating and fixing bugs, it is important to be able to retrieve and run different versions of the software to determine in which version(s) the problem occurs. It may also be necessary to develop two versions of the software concurrently (for instance, where one version has bugs fixed, but no new features (branch), while the other version is where new features are worked on (trunk).

At the simplest level, developers could simply retain multiple copies of the different versions of the program, and label them appropriately. This simple approach has been used on many large software projects. While this method can work, it is inefficient as many near-identical copies of the program have to be maintained. This requires a lot of self-discipline on the part of developers, and often leads to mistakes. Consequently, systems to automate some or all of the revision control process have been developed.

Moreover, in software development, legal and business practice and other environments, it has become increasingly common for a single document or snippet of code to be edited by a team, the members of which may be geographically dispersed and may pursue different and even contrary interests. Sophisticated revision control that tracks and accounts for ownership of changes to documents and code may be extremely helpful or even indispensable in such situations.[9]

2.2.4. External Libraries:

- DevComponents.DotNetBar2 – To enhance textbox control for the validation of inputs.
- ICSharpCode.SharpZip.Lib – To compress ".RELAP" files.
- WeifenLuo.WinFormsUI.Docking – To dock child forms to the parent form.

3. Implementation

The execution of our GUI starts with the appearance of main form. The Nodalization diagram is created using the components on the drawing canvas. Properties for the components are available which are used to create the input file. These properties are defined in accordance with the RELAP5 User manual. When all forms are filled in, the whole file is saved on the hard disk with an extension ".RELAP". The same file can be opened and could be changed or updated easily.

Commonly used VB.NET Controls are:

- **Tab Control:** A Tab Control manages and displays to the user a related collection of Tabs that can contain controls and components. This is shown in Figure 3-1.

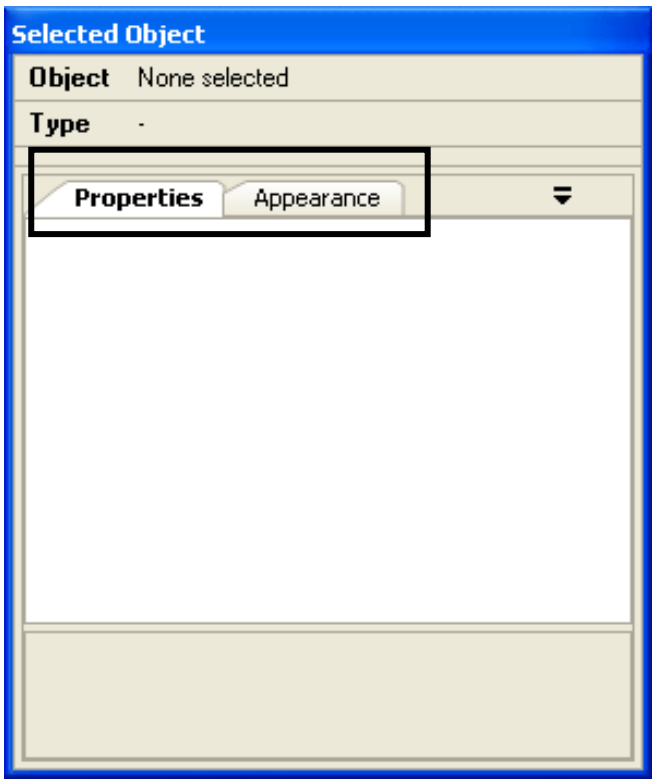


Figure 3-1: Tab Control

- **Text box:** A text box, text field or text entry box is a common element of graphical user interface of computer programs, as well as the corresponding type of widget used when programming GUIs. A text box's purpose is to allow the user to input text information to be used by the program. This is shown in Figure 3-2.

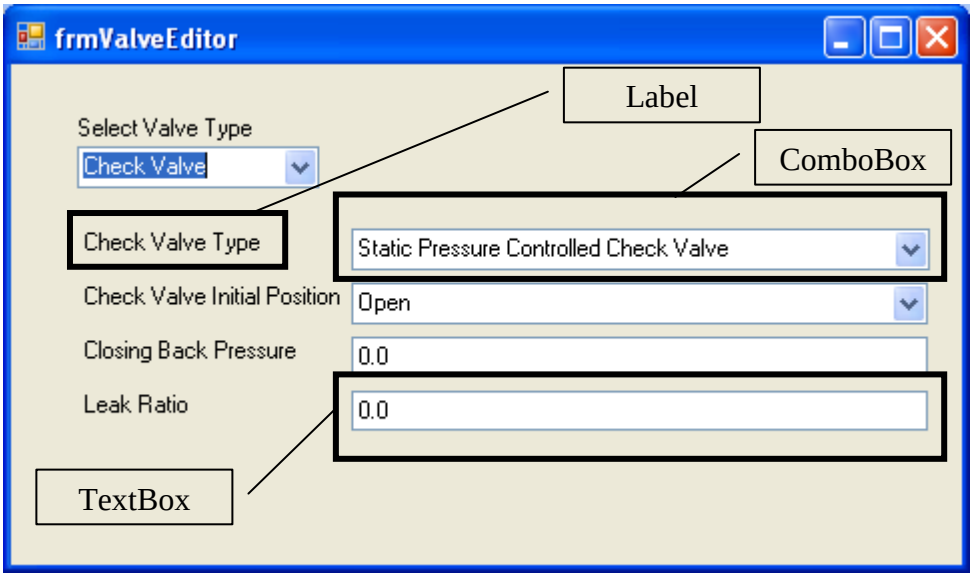


Figure 3-2: Label, TextBox and ComboBox Controls

- **Combo box:** A combo box is a commonly used GUI widget as shown in Figure 3-2. It is a combination of a drop-down list or list box and a single-line textbox, allowing the user either to type a value directly into the control or choose from the list of existing options. An example of this use is the address bar of graphical web browsers.

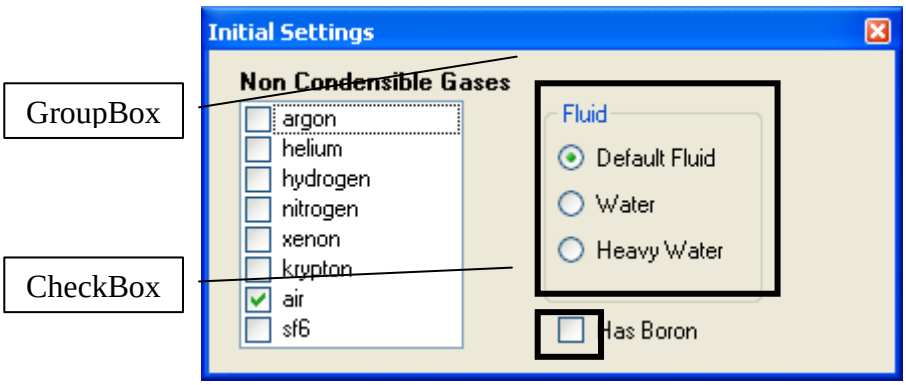


Figure 3-3: Checkbox and GroupBox Control

- **Check box:** In computing, a check box (checkbox, tickbox, or tick box) is a graphical user interface element (widget) that permits the user to make multiple selections from a number of options. Normally, check boxes are shown on the screen as a square box that can contain white space (for false) or a tick mark or X (for true), as shown in Figure 3-3. Adjacent to the check box is normally shown a caption describing the meaning of the check box.

Inverting the state of a check box is done by clicking the mouse on the box, or the caption, or by using a keyboard shortcut.

- **List box:** A list box (as shown in Figure 3-4) is a GUI widget that allows the user to select one or more items from a list contained within a static, multiple line text box.

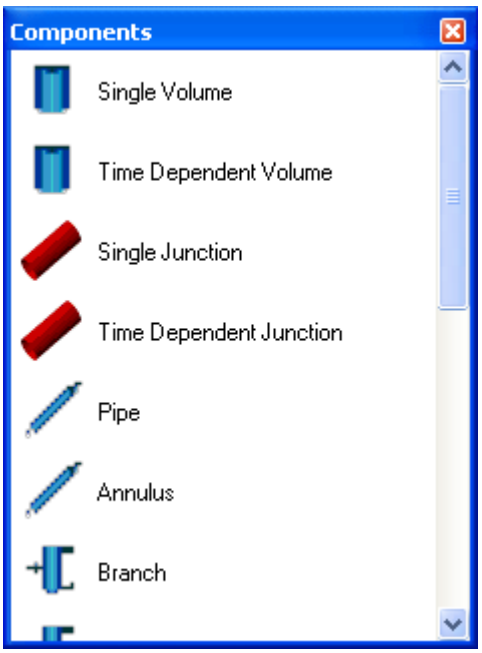


Figure 3-4: ListBox Control

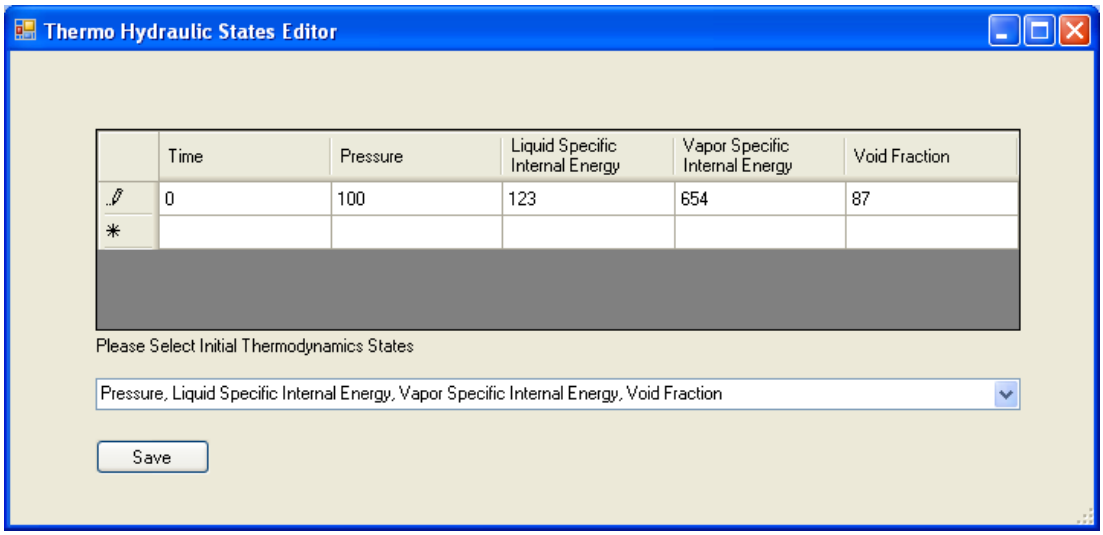


Figure 3-5: DataGridView Control

- **Group box:** A group box (as shown in Figure 3-3) is used for grouping various similar GUI widgets.
- **Label:** This control (as shown in Figure 3-2) is required to aid each aforementioned widget to aid its purpose and requirement on each form.
- **Data Grid View:** Presents the data in a tabular form, enables the user to

add/edit and remove rows on the fly. This is shown in Figure 3-5

3.1. Integrated GUI

An overview of the Integrated GUI for RIFGen is shown in Figure 3-6. It contains a Main Toolbar on the top, Initialization Toolbar also on the top, Flowsheet in the center, Property Grid Form on the left and a Component List on the right. Below are Plot Request, Minor Edit Requests, Control System Data Input, Trips and General Core Input forms.

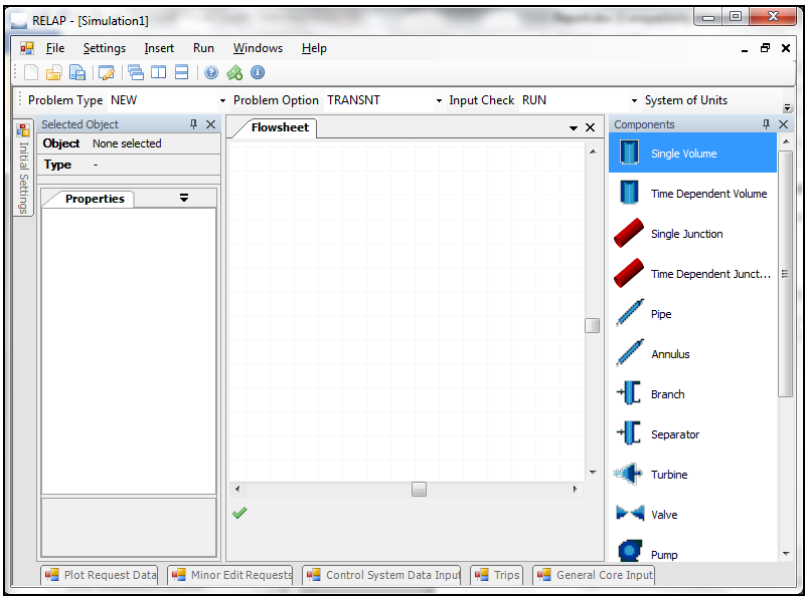


Figure 3-6: Integrated GUI

3.1.1. Main Toolbar / Menu Strip

It has the standard menu items like File, Edit, View, Copy, Save etc. These features are commonly available on all windows based applications.

3.1.2. Initialization Toolbar

This is the initialization toolbar which is always enabled regardless of the type of simulation to be modeled. This is shown in Figure 3-7.

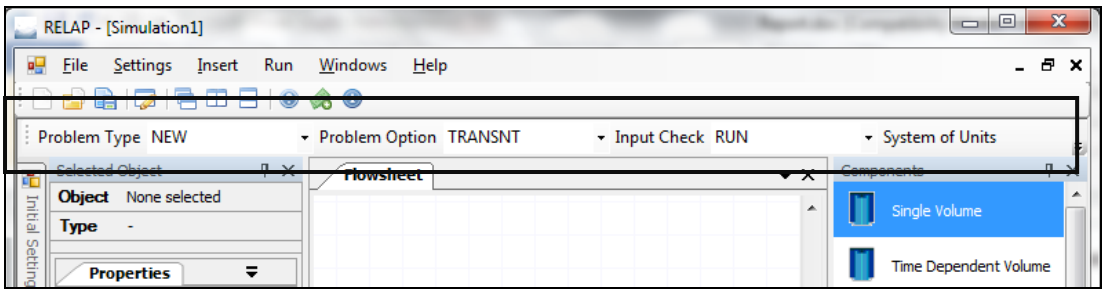


Figure 3-7: Initialization Toolbar

All the fields visible above are taken directly from the user manual for the input file generation of RELAP5.

- **Problem Type:** This is combo box that contains the type of problem as NEW; RESTART; PLOT; IN-COND; STRIP; CMPCOMS.
- **Option:** This combo box specifies INPUT CHECK or RUN.
- **Units:** This is to specify units as SI or BRITISH for both input and output.
- **Type of State:** It mentions the problem type as Steady state or Transient.

3.1.3. Flow Sheet

RIFGen's major module is the Flow Sheet, also known as the drawing canvas or design surface. GDI+ Plus based drawing canvas has been implemented so that Components may be easily dragged and dropped, repositioned, rotated or transformed into any way. Graphics objects can be connected to each other using predefined connection points these have also been implemented so that the process flow of the components is better visualized in the nodalization diagram. A graphic object is drawn using the curves and lines; defining points to draw a curve or a line does this.

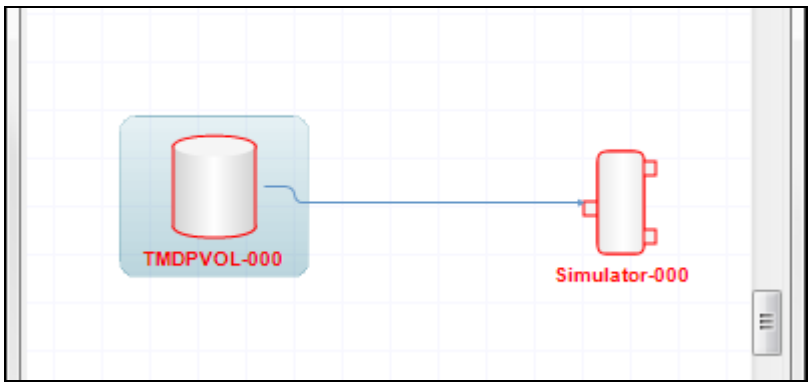


Figure 3-8: Example of Connectors

The graphic object also contains the tag which is used to identify the component, this identification tag is incremented automatically for example Time Dependent Volumes are tagged as TMPDVOL-000, TMPDVOL-001 and so on. This tag may be modified as per user's need. An example of connected components is illustrated in Figure 3-8

3.1.4. Property Grid Form

This form has 2 tabs, one is the property tab and the other is the appearance tab. All component specific properties are visible in this toolbar once the component has been selected on the drawing canvas. These include properties related to the component defined in the RELAP5 User Manual. Values of these properties are formatted according to their data type. As seen in Figure 3-9 the Volume Flow Area is formatted as a real number formatted up to 2 decimal places. These properties are further

categorized for easier user readability and navigation. Detailed description for each property is displayed at the bottom of the form. In the appearance tab the graphics object may be modified Complex properties are further tackled in separate forms called from this parent Property Grid Form using UI Editors.

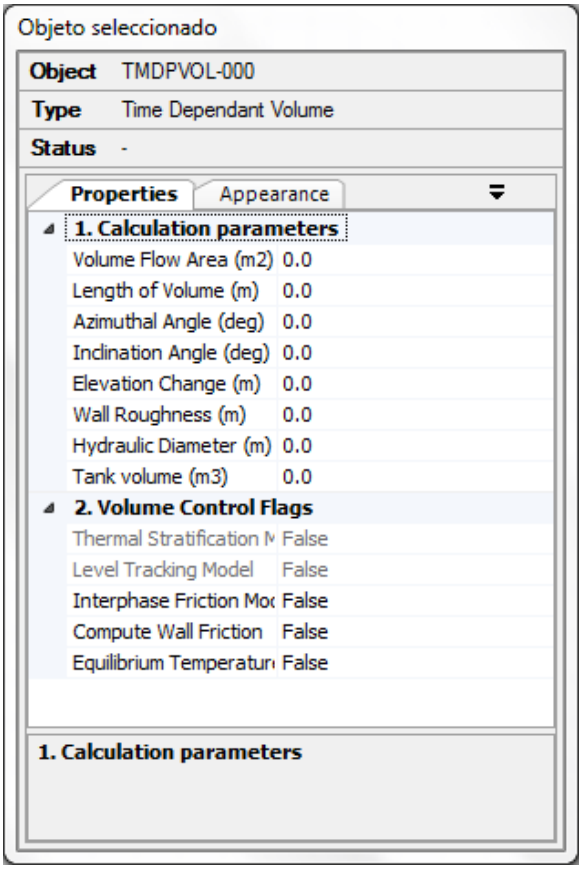


Figure 3-9: Property Grid Toolbox for Time Dependent Volume

3.1.5. User Interface (UI) Editors

These provide a custom design-time experience for complex property types by implementing a user interface (UI) type editor.

Displaying and Editing Custom Types

When a custom type as a property is exposed, there are three ways to edit the property's value in a PropertyGrid:

- Edit the property in place as a string. This requires a TypeConverter for the custom type
- Edit the property with a drop-down UI. This is especially useful for properties that can be set with a single click.
- Edit the property with a modal dialog box. If the property is particularly complex, a full dialog box may be necessary to edit it properly.

To enable either single-click or modal dialog box editing, one needs to implement a

UI type editor to interact with a PropertyGrid.

Drop-down Editors

Drop-down editors are ideal for types that can be set with a single click. For example, to edit the Dock and BackColor properties of the Control class in a PropertyGrid with a drop-down editor.

Access a drop-down UI type editor by clicking on the arrow button that appears next to the selected property entry in a PropertyGrid. The custom UI appears, attached to the PropertyGrid. The top of its window is positioned along the bottom of the property entry, and its width matches that of the property entry. This editor window must also be closed after the user makes a selection. The implementation must call the DropDownControl method to position and size the UI type editor window in the design environment, and must call the CloseDropDown method to close the window.[10]

Modal Dialog Editors

Modal editors are useful for types that require a fully interactive UI. For example, collection editors like the TabPage Collection Editor of TabControl or the Edit Columns dialog box of the DataGridView control are modal editors.

Access a modal UI type editor by clicking on the ellipsis button that appears next to the selected property entry in a PropertyGrid. The modal dialog box appears, and the user interacts with it like a typical dialog box. The implementation must call the ShowDialog method to position and size the dialog box in the design environment.

Implementing a UI Type Editor

To implement a custom UI type editor, the following tasks must be performed:

- Define a class that derives from UITypeEditor.
- Override the GetEditStyle method to inform the PropertyGrid of the type of editor style that the editor will use.
- Override the EditValue method to handle the UI, user input processing, and value assignment.

The following tasks need to be performed to add additional support for painting a value's representation in a PropertyGrid:

- Override GetPaintValueSupported to indicate that the editor supports displaying the value's representation.
- Override PaintValue to implement the display of the value's representation.
- Override the UITypeEditor constructor method if the editor should have

initialization behavior.

3.1.6. Component List

This list contains a list of all RELAP5 Components, these components are designed in a way that they can be added to RELAP5 in an incremental manner without disrupting the existing coding. Component coding task has been distributed among team members so that the application may be coded in parallel with optimal progress.

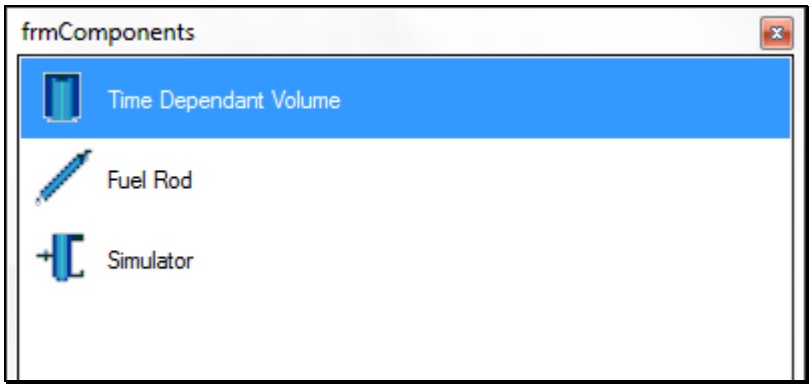


Figure 3-10: Components List

3.2. File Save/Load

This has been achieved by writing export files for each object in the GUI and zipping them into a single file with the extension of ".RELAP". Objects and GUI layout is serialized and exported to binary files. This may later be used to read and load the layout back again. The properties of the components are in this way saved and may be reused later when needed.

3.2.1. Serialization

In computer science, in the context of data storage and transmission, serialization is the process of translating data structures or object state into a format that can be stored and restored later in the same or another computer environment. When the resulting series of bits is reread according to the serialization format, it can be used to create a semantically identical clone of the original object. For many complex objects, such as those that make extensive use of references, this process is not straightforward. Serialization of object-oriented objects does not include any of their associated methods with which they were previously inextricably linked.

This process of serializing an object is also called deflating or marshalling an object. The opposite operation, extracting a data structure from a series of bytes, is deserialization. In the .NET languages, classes can be serialized and deserialized by adding the Serializable attribute to the class.

If new members are added to a serializable class, they can be tagged with the OptionalField attribute to allow previous versions of the object to be deserialized without error. This attribute affects only deserialization, and prevents the runtime from throwing an exception if a member is missing from the serialized stream. A member can also be marked with the NonSerialized attribute to indicate that it should not be serialized. This will allow the details of those members to be kept secret.

Objects may be serialized in binary format for deserialization by other .NET applications. There are also third party binary serializers that are documented, portable, use less memory footprint and CPU.

The framework also provides the SoapFormatter and XmlSerializer objects to support serialization in human-readable, cross-platform XML.

3.3. Publishing of Software

The beta version was published on Codeplex on 2nd May 2013. Codeplex enables the use of Team Foundation Server (TFS) which was our main asset to help us in the integrated team development. The project URL is http://relap.codeplex.com.

3.3.1. About CodePlex

CodePlex is Microsoft's open source project hosting web site. It can be used to create new projects to share with the world, join others who have already started their own projects, or download open source software on this site and provide feedback. Hosting over 30,000 projects, CodePlex is one of the fastest growing and most popular open source project hosting sites. CodePlex provides a rich set of functionality for hosted projects including:

- Team Foundation Server, Git, or Mercurial for project source control
- Project contributor forks or patches
- Project release downloads
- Discussion forums & mailing lists
- Wiki and documentation pages
- Bug and feature request tracker
- Project usage statistics

4. Conclusions and Recommendations

4.1. Conclusions

It is concluded that a GUI for the preparation of the input file for RELAP5 has been developed using VB.NET as programming language, using GDI+ as the main module for drawing purposes. This Software is named as RELAP5 Input File Generator (RIFGen). It is capable of reducing work of hours to few minutes. It has been validated for individual components as well as on complete test cases as a whole. RIFGen exhibited correct results and is ready to be used for any PWR type reactor. Hence, the aim of this project is achieved.

4.2. Recommendations

There is always room for improvement. RIFGen currently works for PWR reactors only. It can be improved to work for both BWR and ATR. It's input file generator outputs files according to Mod. 3.2. It can be improved to work for latest Mod. 4.0. There can be options to read output file and plot results. There can be an option to convert an already prepared input file to a nodalization diagram on the drawing canvas. Inclusion of post processing of results and grouping of components into a subsystem may also be a useful feature. A feature to import existing nodalization diagrams may be included.

References

[1] Daniel Wagner Oliveira de Medeiros, *"DWSIM - Process Simulation, Modeling and Optimization"*, Version 2.1, Revision 0, January 2012 Source Code Guide.

[2] CAPE-OPEN Methods and Tools Guidelines by "Industrial and Materials Technologies.

[3] RELAP/MOD Code Manual, volume I: code structure, system models, and solution methods, NUREG/CR-5535, June 1995.

[4] SCDAP/RELAP/MOD3.2 Code Manual, NUREG/CR/6150, rev.1, July 1998.

[5] E. A. Harvego, et al., "Developmental assessment of the SCDAP/RELAP/MOD3.2 code" 6th Inter. Conference on Nuclear Engineering, ICONE-6, May 1998.

[6] James J. Duderstadt and Louis J. Hamilton, "*Nuclear Reactor Analysis*", John Wiley & Sons, New York/London/Sydney/Toronto, 1976.

[7] Dr. George Mesina, "*Developments and New Directions for the RELAP5-3D Graphical User Interface"* , 2001 RELAP5 International Users Seminar.

[8] RELAP/MOD Code Manual, volume 5A:mod 3.2 Assessment, NUREG/CR-5535, June 1995.

[9] Microsoft® 2013, "*.NET Framework Development Guide*", Islamabad Office: Microsoft Corporation Pakistan, Liaison Office, House No. 10/A, Street No. 71, Sector F-8/3 Islamabad.
Website: <http://msdn.microsoft.com/en-us/library/hh156542.aspx>

[10] Microsoft® 2013, "*The Three Parts of GDI+",* Islamabad Office: Microsoft Corporation Pakistan, Liaison Office, House No. 10/A, Street No. 71, Sector F-8/3 Islamabad.
Website:<http://msdn.microsoft.com/en-us/library/windows/desktop/ms536384(v=vs.85).aspx>

Vita

The author, Waleed Ahmed Malik, was born in Rawalpindi, Pakistan on 30th November 1987. His initial schooling until grade 9 was in Al Nujoom International School Jeddah, Saudi Arabia. Completed O' and A' Levels from OPF Boys College H-8/4, Islamabad. Graduated with a degree in Computer Information Sciences from PIEAS, Islamabad. Later he completed his MS in Nuclear Engineering from PIEAS as well. The author also enjoys working as a freelance programmer and has a rich 6 year experience of the corporate world of Software Engineering.