Transactions of the Korean Nuclear Society Autumn Meeting
Gyeongju, Korea, October 27-28, 2016

Development of Visual CINDER Code with Visual C#.NET

Oyeon Kum [a,b]*
[a]*Institute for Modeling and Simulation Convergence, Daegu 42045, Korea*
[b]*University of Southwest America, LA 90005, USA*
[*]*Corresponding Author: okum@uswa.ac*

1. Introduction

CINDER code, CINDER'90 [1] or CINDER2008 [2] that is integrated with the Monte Carlo code, MCNPX [3], is widely used to calculate the inventory of nuclides in irradiated materials. The MCNPX code provides decay processes to the particle transport scheme that traditionally only covered prompt processes. The integration schemes serve not only the reactor community (MCNPX burnup) but also the accelerator community as well (residual production information). For the accelerator community, the integration scheme requires additional scripts to prepare multiple step computing as shown in Fig.1.

Figure 1 shows the general flow of computing procedures which combines with the MCNPX and Perl scripts. Because the task of extracting isotope production rates and flux spectra from MCNPX output and "histp" files is the same for Performing activation analyses with other transmutation codes such as ORIHET3 and SP-FISPACT, the scripts were extended to serve and execute simultaneously with these transmutation codes. The big benefit for providing these options lies in the easy cross comparison of the transmutation codes since the calculations are based on exactly the same material, neutron flux and isotope production/destruction inputs. However, it is just frustratingly cumbersome to use. In addition, multiple human interventions may increase the possibility of making errors. The number of significant digits in the input data varies in steps, which may cause big errors for highly nonlinear problems. Thus, it is worthwhile to find a new way to wrap all the codes and procedures in one consistent package which can provide ease of use.

In this study, we develop a « wrapper » (visual CINDER code) with visual C#.NET [4]. Visual C# is modern, high-level, multi-paradigm, general-purpose programming language for building apps using Visual Studio and the .NET Framework. The many innovations in C# enable rapid application development while retaining the expressiveness and elegance of C-style languages.

2. Methods and Results

2.1 Wrapping Methods

C# is designed to be a simple, modern, general-purpose, object-oriented programming language that borrows key concepts from several other languages such as Java, VB, and Xwin. Because C# is an object-oriented language and does not offer global variables or functions, everything is wrapped in classes, even simple types like « int » and « string », which inherits from the « System.Object » class. C# alone could theoretically be compiled to machine code but it is always used in combination with the .NET framework in the stage of action. Thus, application programs with C# require the .NET framework to be installed in advance on the computer running the programs. However, the .NET framework makes it possible to use a wide range of other coding languages. The reason why C# is referred to as « THE .NET » language is perhaps because it was designed together with the framework.

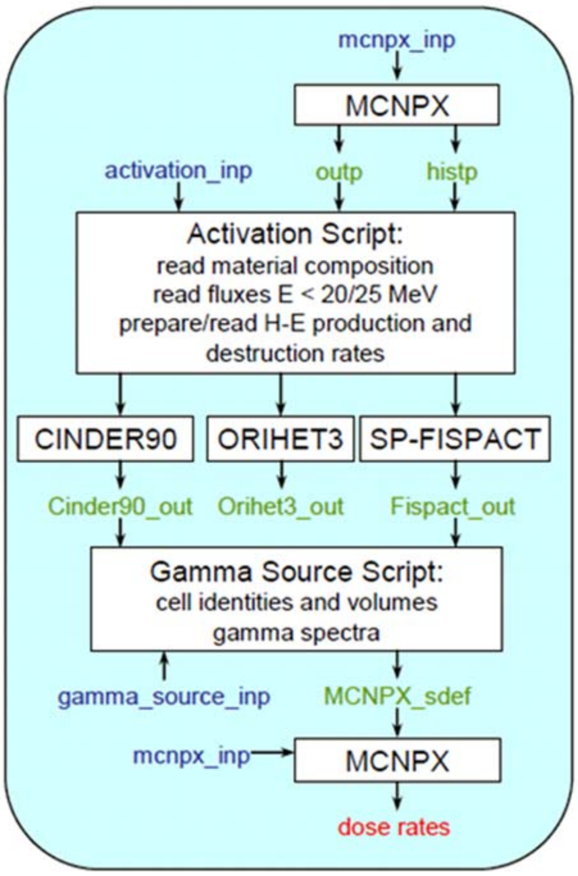


Fig. 1. Flow diagram of an activation analysis employing the activation script and the gamma source script with the MCNPX radiation transport code.

C# can be written with any text editor, like Windows Notepad, and then compiled with the C# Command line compiler, "csc.exe", which comes with the .NET framework. However, most people prefer to use an Integrated Development Environment (IDE), and

Microsoft offers several options for this. Their flagship is Visual Studio, which can be used to work on every possible aspect of the .NET framework. This product is very advanced, and comes in several editions. Microsoft introduced the so-called Express version which is free, targeted at hobby programmers and people wanting to try .NET. Although the Express versions miss some of the really advanced features of Visual Studio, it works fine for the C#. Thus, the visual C#.NET is easily available, free, and convenient tool as a good wrapper.

Since CINDER code is written in FORTRAN, first we need to build FORTRAN code into a DLL (Dynamic Link Library) to call them from C# code, and then use Platform Invoke, a service that enables managed code to call an unmanaged function or subroutines inside the DLL. Platform Invoke service locates and calls unmanaged code as an exported function. It also marshals the call's arguments, such as input and output parameters, integers, strings, arrays, and structures, as needed. It is recommended to wrap the FORTRAN function or subroutine in a managed class. Within the class, you define a static method for each FORTRAN function or subroutine to be called. The "DllImportAttribute" is used to identify the DLL and function. The definition can include additional information, such as the calling convention used in passing method arguments. Figure 2 shows unmanaged FORTRAN code calling schemes in combination with the managed C# main code.

Visual Studio is the IDE in which developers can create programs in C# for the .NET framework. It is used to create console and graphical user interface (GUI) applications along with Windows Forms or Windows Presentation Foundation(WPF) applications, 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.

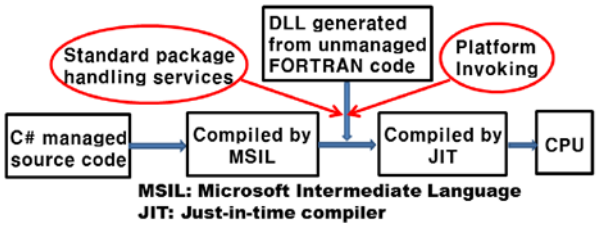

DLL generated from unmanaged FORTRAN code
Standard package handling services
Platform Invoking
C# managed source code
Compiled by MSIL
Compiled by JIT
CPU
MSIL: Microsoft Intermediate Language
JIT: Just-in-time compiler

Fig. 2. Schematic diagram to handle DLLs generated from unmanaged FORTRAN code.

The capability to create multiple forms provides C# programmers to make extensive use of forms for user interfaces. Each time they create a Windows application, Visual Studio will display a default blank form, onto which they can drag the controls over the applications main form and adjust its size and position.

To add a menu to the Visual Studio integrated development environment (IDE), a "VSPackage" must use the Visual Studio SDK architecture for command-group menus. A command-group menu enables the sharing of commands by components and the IDE. In "VSPackages", menus are defined in the Menus section of a ".vsct" file. A ".vsct" file defines menus, toolbars, groups, and commands. A command is what a user clicks to perform a function. A group is a container for commands. A menu is a container for groups. There are three basic ways that a menu can appear in Visual Studio: (1) a menu on the main menu bar; (2) a submenu of another menu; (3) a shortcut menu (typically displayed by a right-click). To include a custom picture logo, the "Window.SetTabPicture" method is used.

The usefulness of data visualization is to communicate information clearly and efficiently to others via a 2D or 3D graphics and plots. Effective visualization is very helpful in analyzing and reasoning about the scientific data or involved physics. It makes complex scientific data more accessible, understandable, and usable. Particular analytical tasks such as making comparisons or understanding causality can be effectively performed with the efficient design principle of the graphic although tables may sometimes be used. Various types of line and chart plots are implemented in this code to show patterns or relationships in the data for one or more variables with the convenient functions provided by the Visual C#.NET tool.

Data visualization is both an art and a science. Tremendous data generated by scientific computing can be shown in a succinct image. In this sense, it can be though as an art. Data created by radiation transport and transmutation simulations can be referred to as a "Big Data". Thus, processing, analyzing and communicating this data present a variety of ethical and analytical challenges for data visualization. The visual CINDER code includes 2D and 3D graphics utilities to help address this challenge.

The original CINDER code produces a maximum of about 60 text data tables. The size of each table has no preset limit, so text file handling is also a big challenge. Practically, many researchers use the Microsoft Excel spread sheet to handle these files, which is tedious and time consuming work. Visual C#.NET can provide many functions to generate Microsoft Office stub such as Excel or Word, and many manual tasks can be automated. The visual CINDER code implements these utilities and writes the data set in an excel spread sheet automatically.

2.2 Results and Discussion

Figure 3 shows the main face of the visual CINDER code. It has many menus including main menus of «

File », « Input », « Run », « PostProcessing », « Outputs », « Plots », « CrossectionPlots », « ExcelFormat », « CADImport », « Options », « View », and « Help ». Each main manu has many submenus. In the Fig. 3., main menu « Run » shows four submenus of « MCNPX », « HISTP », « CINDER », and « GAMMA_MCNPX ». However, at this moment, MCNPX computing is performed separately.

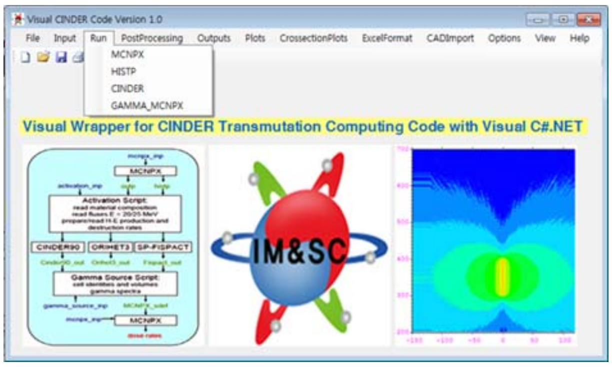


Fig. 3. Main face of the visual CINDER code.

We used the multiple form generation method to handle the graphical user interface. Thus, each submenu may generate a new form which also includes many necessary menus.

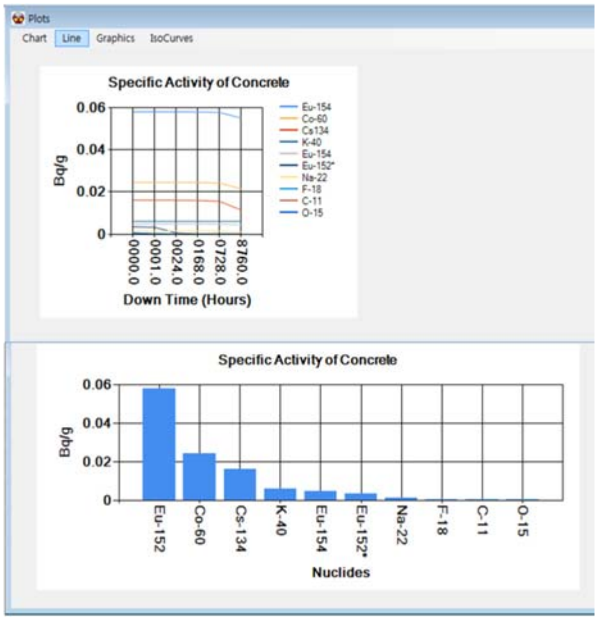


Fig. 4. "Plots" menu, submenus, and two types of sample pictures.

Manu menu "Plots" has four submenus of "Chart", "Line", "Graphics", and "IsoCurves" as shown in Fig. 4. "Chart" and "Line" menus are used to draw chart and line type of pictures as shown in the figure. The "Graphics" menu draws two- and three-dimensional color pictures and "IsoCurves" menu is for drawing iso-value curves of tally or density distributions. Figure 5 shows a sample two-dimensional tally picture overlapped with a plan view of a CAD drawing for a heavy-ion accelerator center. However, this menu is still under development.

Microsoft office provides many useful data analysis tools such as MS word and Excel spread sheet. The resulting output text files are sometimes copied to word or excel files by hand. However, manually copy and pasting by hand is tedious and time consuming. Visual C#.NET allows automation of such processes. Figure 6 shows an excel spread sheet including an example of a text output file of the CINDER code. Data copy and paste procedures were performed automatically by using an Excel application object provided by the visual C#.NET programming tool. Many different implementation examples are provided by the Microsoft Office web site in addition to other useful application examples.

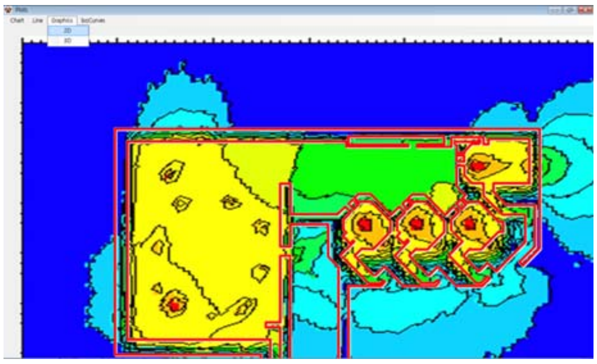
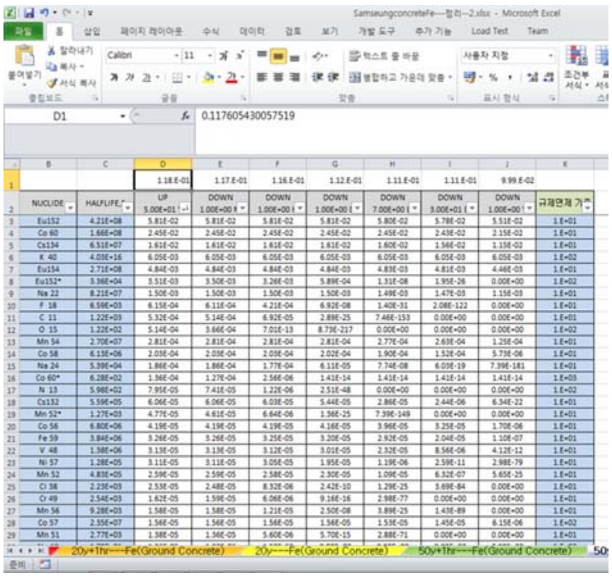


Fig. 5. "Graphics" submenu function example.


SamsungconcreteFe---정리--2.xlsx - Microsoft Excel
Calibri
11
D1
20y+1hr---Fe(Ground Concrete)
20y---Fe(Ground Concrete)
50y+1hr---Fe(Ground Concrete)

D1 0.117605430057519

	B	C	D	E	F	G	H	I	J	K
1			1.18.E-01	1.17.E-01	1.16.E-01	1.12.E-01	1.11.E-01	1.11.E-01	9.99.E-02	
2	NUCLIDE	HALFLIFE,	UP 5.00E+01	DOWN 1.00E+00	DOWN 1.00E+00	DOWN 1.00E+00	DOWN 7.00E+00	DOWN 3.00E+01	DOWN 1.00E+00	규제면제 가
3	Eu152	4.21E+08	5.81E-02	5.81E-02	5.81E-02	5.81E-02	5.80E-02	5.78E-02	5.51E-02	1.E+01
4	Co 60	1.66E+08	2.45E-02	2.45E-02	2.45E-02	2.45E-02	2.45E-02	2.43E-02	2.15E-02	1.E+01
5	Cs134	6.51E+07	1.61E-02	1.61E-02	1.61E-02	1.61E-02	1.60E-02	1.56E-02	1.15E-02	1.E+01
6	K 40	4.03E+16	6.05E-03	6.05E-03	6.05E-03	6.05E-03	6.05E-03	6.05E-03	6.05E-03	1.E+02
7	Eu154	2.71E+08	4.84E-03	4.84E-03	4.84E-03	4.84E-03	4.83E-03	4.81E-03	4.46E-03	1.E+01
8	Eu152*	3.36E+04	3.51E-03	3.50E-03	3.26E-03	5.89E-04	1.31E-08	1.95E-26	0.00E+00	1.E+02
9	Na 22	8.21E+07	1.50E-03	1.50E-03	1.50E-03	1.50E-03	1.49E-03	1.47E-03	1.15E-03	1.E+01
10	F 18	6.59E+03	6.15E-04	6.11E-04	4.21E-04	6.92E-08	1.40E-31	2.08E-122	0.00E+00	1.E+01
11	C 11	1.22E+03	5.32E-04	5.14E-04	6.92E-05	2.89E-25	7.46E-153	0.00E+00	0.00E+00	1.E+01
12	O 15	1.22E+02	5.14E-04	3.66E-04	7.01E-13	8.73E-217	0.00E+00	0.00E+00	0.00E+00	1.E+02
13	Mn 54	2.70E+07	2.81E-04	2.81E-04	2.81E-04	2.81E-04	2.77E-04	2.63E-04	1.25E-04	1.E+01
14	Co 58	6.13E+06	2.03E-04	2.03E-04	2.03E-04	2.02E-04	1.90E-04	1.52E-04	5.73E-06	1.E+01
15	Na 24	5.39E+04	1.86E-04	1.86E-04	1.77E-04	6.11E-05	7.74E-08	6.03E-19	7.39E-181	1.E+01
16	Co 60*	6.28E+02	1.36E-04	1.27E-04	2.56E-06	1.41E-14	1.41E-14	1.41E-14	1.41E-14	1.E+03
17	N 13	5.98E+02	7.95E-05	7.41E-05	1.22E-06	2.51E-48	0.00E+00	0.00E+00	0.00E+00	1.E+02
18	Cs132	5.59E+05	6.06E-05	6.06E-05	6.03E-05	5.44E-05	2.86E-05	2.44E-06	6.34E-22	1.E+01
19	Mn 52*	1.27E+03	4.77E-05	4.61E-05	6.64E-06	1.36E-25	7.39E-149	0.00E+00	0.00E+00	1.E+01
20	Co 56	6.80E+06	4.19E-05	4.19E-05	4.19E-05	4.16E-05	3.96E-05	3.25E-05	1.70E-06	1.E+01
21	Fe 59	3.84E+06	3.26E-05	3.26E-05	3.25E-05	3.20E-05	2.92E-05	2.04E-05	1.10E-07	1.E+01
22	V 48	1.38E+06	3.13E-05	3.13E-05	3.12E-05	3.01E-05	2.32E-05	8.56E-06	4.12E-12	1.E+01
23	Ni 57	1.28E+05	3.11E-05	3.11E-05	3.05E-05	1.95E-05	1.19E-06	2.59E-11	2.98E-79	1.E+01
24	Mn 52	4.83E+05	2.59E-05	2.59E-05	2.58E-05	2.30E-05	1.09E-05	6.32E-07	5.65E-25	1.E+01
25	Cl 38	2.23E+03	2.53E-05	2.48E-05	8.32E-06	2.42E-10	1.29E-25	3.69E-84	0.00E+00	1.E+01
26	Cr 49	2.54E+03	1.62E-05	1.59E-05	6.06E-06	9.16E-16	2.98E-77	0.00E+00	0.00E+00	1.E+01
27	Mn 56	9.28E+03	1.58E-05	1.58E-05	1.21E-05	2.50E-08	3.89E-25	1.43E-89	0.00E+00	1.E+01
28	Co 57	2.35E+07	1.56E-05	1.56E-05	1.56E-05	1.56E-05	1.53E-05	1.45E-05	6.15E-06	1.E+02
29	Mn 51	2.77E+03	1.38E-05	1.36E-05	5.60E-06	5.70E-15	2.88E-71	0.00E+00	0.00E+00	1.E+01

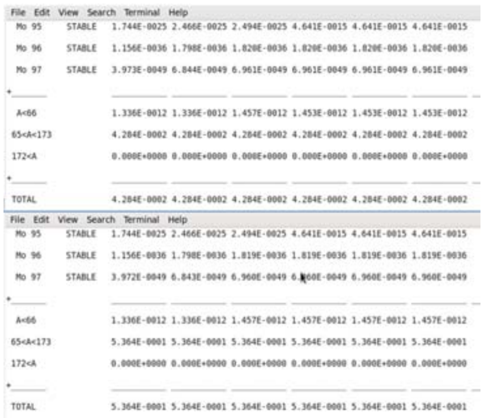
Fig. 6. Automatic excel file conversion example.

The development of visual CINDER code (wrapper) makes all previous intermediate information, for example, material data, volume, and fluxes to be transmitted directly via computer memories with full significant digits for the next step calculation. A separate input file controls the range of simulation with optional flags for simple particle transport simulation to atomic transmutation and radiation dose computations by gamma ray distributions. It is also possible that the respective input files for CINDER and MCNPX codes can be merged into one file. This process is very important because nuclear transmutation process is governed by highly non-linear differential equations.

The chaotic nature of non-linear equation bespeaks the importance of an accurate input data (i.e. number of significant digits). Thus, reducing human intervention is very important for rigorous transmutation study, and the role of wrapper code is desired because it is also easy to include intrinsic physical problem such as flux changes in a given time interval. Figure 7 is an example of the atom-density result (unit: atoms/barn-cm) of a "Blue Room" problem [5] which shows the dependence of the computed results on the number of significant digits of the input files. Upper part is the results computed from three significant digits input files which are usual outputs of activation script such as "flux" and "splsprod" files. Lower part is the results computed from eleven significant digits input files. As shown in the figure, 0.1% difference in the input data produces more than 20% difference in some results, not all results, which imply strong chaotic characteristics.

3. Conclusions

The visual CINDER code development is underway with visual C#.NET framework. It provides a few benefits for the atomic transmutation simulation with CINDER code. A few interesting and useful properties of visual C#.NET framework are introduced. We also showed that the wrapper could make the simulation accurate for highly nonlinear transmulation problems and also increase the possibility of direct combination a radiation transport code MCNPX with CINDER code. Direct combination of CINDER with MCNPX in a wrapper will provide more functionalities for the radiation shielding and prevention study. Moreover, convenient graphical user interface provided by the visual C#.NET framework will be a great asset for the use of CINDER code.



```
 Mo 95      STABLE    1.744E-0025 2.466E-0025 2.494E-0025 4.641E-0015 4.641E-0015 4.641E-0015
 Mo 96      STABLE    1.156E-0036 1.798E-0036 1.820E-0036 1.820E-0036 1.820E-0036 1.820E-0036
 Mo 97      STABLE    3.973E-0049 6.844E-0049 6.961E-0049 6.961E-0049 6.961E-0049 6.961E-0049
+_______              ___________ ___________ ___________ ___________ ___________ ___________ _
 A<66                 1.336E-0012 1.336E-0012 1.457E-0012 1.453E-0012 1.453E-0012 1.453E-0012
65<A<173              4.284E-0002 4.284E-0002 4.284E-0002 4.284E-0002 4.284E-0002 4.284E-0002
172<A                 0.000E+0000 0.000E+0000 0.000E+0000 0.000E+0000 0.000E+0000 0.000E+0000
+_______              ___________ ___________ ___________ ___________ ___________ ___________ _
TOTAL                 4.284E-0002 4.284E-0002 4.284E-0002 4.284E-0002 4.284E-0002 4.284E-0002
```

```
 Mo 95      STABLE    1.744E-0025 2.466E-0025 2.494E-0025 4.641E-0015 4.641E-0015 4.641E-0015
 Mo 96      STABLE    1.156E-0036 1.798E-0036 1.819E-0036 1.819E-0036 1.819E-0036 1.819E-0036
 Mo 97      STABLE    3.972E-0049 6.843E-0049 6.960E-0049 6.960E-0049 6.960E-0049 6.960E-0049
+_______              ___________ ___________ ___________ ___________ ___________ ___________ _
 A<66                 1.336E-0012 1.336E-0012 1.457E-0012 1.457E-0012 1.457E-0012 1.457E-0012
65<A<173              5.364E-0001 5.364E-0001 5.364E-0001 5.364E-0001 5.364E-0001 5.364E-0001
172<A                 0.000E+0000 0.000E+0000 0.000E+0000 0.000E+0000 0.000E+0000 0.000E+0000
+_______              ___________ ___________ ___________ ___________ ___________ ___________ _
TOTAL                 5.364E-0001 5.364E-0001 5.364E-0001 5.364E-0001 5.364E-0001 5.364E-0001
```

Fig. 7. Atomic density (unit: atoms/barn-cm) result for two input files with different significant digits.

REFERENCES

[1] W. B. Wilson, S. T. Cowell, T. R. England, A. C. Hayes and P. Moller, "A Manual for CINDER'90 Version 07.4 Codes and Data", LA-UR-07-8412, LANL (2008).

[2] S. T. Holloway, W. B. Wilson, Charles T. Kelsey, IV, Hannah Little, Vladimir Mozin, "A Manual for cinder2008 Codes and Data," LA-UR 11-00006, (2011).

[3] Denise B. Pelowitz (editor), "MCNPXTM USER'S MANUAL Version 2.7.0", LA-CP-11-00438, April (2011).

[4] Ken Carney, "Visual C# .NET", Home and Learn, November (2013).

[5] Bradley J. Micklich, Erik B. Iverson, Franz X. Gallmeier, Wei Lu, Holly Trellue, Charles Kelsey, and Michael Wohlmuther, "DEVELOPMENT OF A SAMPLE PROBLEM SUITE FOR VALIDATION AND VERIFICATION OF CINDER'90 AND SCRIPTING TOOLS", AccApp'07, Pocatello, Idaho, July 29-August 2, 2007.