International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering (M&C 2011)
Rio de Janeiro, RJ, Brazil, May 8-12, 2011, on CD-ROM, Latin American Section (LAS) / American Nuclear Society (ANS)
ISBN 978-85-63688-00-2

"La Java de Silène"

A GRAPHICAL USER INTERFACE FOR 3D PRE & POST PROCESSING: STATE-OF-THE-ART AND NEW DEVELOPPEMENTS

Zarko Stankovski
Commissariat à l'Energie Atomique (CEA Saclay)
DEN/DANS/DM2S/SERMA/LTSD
91191 Gif-sur-Yvette Cedex, France
zstankovski@cea.fr

ABSTRACT

« La Java de Silène »[1] **(Silène)** is an original Graphical User Interface (**GUI**), written in Java language, for generation of 3D arbitrarily shaped geometries for the neutron transport codes used at CEA. Silène provides data for APOLLO2 input format as well as for some other codes like Monte Carlo TRIPOLI4 code. In addition, the software serves as an experimental tool to explore the needs and feasibility for the GUI of the currently developing APOLLO3. This paper summarizes the principles of the software and presents some new capabilities. The interface deals simultaneously with two kinds of geometries: regular and unstructured, arranged in collections. Unstructured geometries are built visually and regular ones visually or by using text editor. Regular geometries consist of sets of predefined and parameterized components. The new developments will improve the efficiency of components with the ability to use variables instead of fixed values. Use of variables, derived components (mutants) and combination of media with generic names and meshes with physical names allow the management of collections with much different isotopic enrichment without increasing the number of fuel cells. The object of all these recent developments is to ensure an efficient and secure management of large and complex geometrical configurations. The effectiveness of the current version of Silène is illustrated with an example for a Boiling Water Reactor Assembly.

Key Words: GUI, Unstructured Geometry, Neutron Transport.

1. INTRODUCTION

The GUI Silène is developed at CEA for visually building unstructured 2D&3D geometries, to generate data for computer codes and post-processing the calculation results. Initially Silène was designed to provide input data for TDT [1], IDT[2], stand-alone applications and modules in APOLLO2 [3] code. The TDT and IDT algorithms solve two and three dimensional Boltzmann transport equation, TDT using collision probabilities and characteristics method in unstructured geometries and IDT discrete ordinates, nodal and characteristics methods in Cartesian geometries. Silène maintain a high compatibility with TDT: any geometry that can be calculated by TDT can be generated by Silene. The opposite is not true since it also generates data for

[1] **Java**: a popular Paris suburbs Saturday night dance. **Σειληνός, Sileno**, **Silène**, **Silenus, Sileno**: a reveller satyr from Greek and Roman mythology

Monte Carlo TRIPOLI4 [4] code. In addition, the software serves as an experimental tool to explore the needs and feasibility of the future GUI for the now developing APOLLO3, successor of APOLLO2. Silène already provides geometrical data for MINARET [5] solver of APOLLO3.

Originally Silène [6] was designed to visually generate and handle arbitrary geometries. In addition to geometric data Silène manages media and their spatial distribution, isotopes (for TRIPOLI4) and boundary conditions. In this paper, an arbitrary geometry mesh with the media distribution and its boundary conditions is referred as **motive**.

Silène is designed with great care for user-friendliness and simplicity of use. Starting with an elementary figure, by mouse selection and with a minimum keyboard strokes, user can build very complex geometries.

Since the previous versions Silène [7] proposes the concept of **components** that offer the ability to create predefined and parameterized geometric patterns described visually or by a symbolic language. This concept greatly facilitates the description of the large problems, such as a reactor core by representing all clads and fuel pins as well as the surrounding reflector. The available components are cells (cylindrical, rectangular and hexagonal), grids (rectangular, hexagonal and rectangular of hexagons) and boxes (rectangular and hexagonal). All these components and unstructured geometries can be nested into each other. Derived components, called **mutants** [8], can be defined from the components already defined by specifying only the parameters that change.

The latest developments concern the implementation of the concept of **variables**. In the classic version of Silène parameters are numerical values or strings. Now all parameters can be represented by variables. The variables are typed. There are four types of variables: integer, double, string and component. The double and integer variables can be bounded. Variables can be defined by elementary operations between fixed values and other variables.

The latest version, Silène 1.2.2 was delivered on March 1, 2010. « La Java de Silène » is the name of the developing version.

2. SILENE GUI

Usually the GUI used for the computer codes are extensions or adaptations of existing applications. Ordered by code developers they are built by GUI professionals, with two disadvantages. First, developers do not know all that is possible to do and may not require the most appropriate features. On the other hand, sellers are not familiar with the requirements of the codes and could not suggest the most appropriate features by themselves. So, many easy to make useful features will be ignored.

Silène is an original graphic user interface, developed expressly for the neutron transport codes used at CEA. The internal data model, functionalities and the window system are designed specially for the geometrical configurations treated.

Silène is written in Java. Internationalization, as defined in Java, is implemented in the software so that it can be adapted to various languages and regions, French from Canada for exemple, without engineering changes. Only the library of associated text files has to be translated.

2.1. Internal Data Model

As Silène deals with two kinds of geometries: **regular** and **unstructured**, there are two operating modes for building geometries: by creating **components** or by drawing **shapes**. Both modes operate simultaneously and the final result can be a combination of both types. So the user can imagine and build any refinement of zones in a regular geometry or simplify the generation of an unstructured geometry containing regular zones.

2.1.1. Unstructured geometries

An unstructured geometry, a **motive**, is defined by a set of arbitrary meshes delimited by equations: straight line segments, circles, arcs and involutes of circles.

To describe the geometry, a set of **nodes** are defined. The **equations** are defined using these nodes. Sets of equations, forming closed figures, define 2D **meshes**. Each equation has a link to the left and right neighboring meshes. Hence, the geometrical description used by Silène is local. The code does not know whether the described motif is a cell, assembly or something else.

The 3D axially extruded geometries are represented by axial **levels**. The 2D geometry is a special case, a motive with only one level. The nodes, equations and meshes are the same for all levels. **Boundary conditions** can be defined for all equations on the external perimeter, and on top and bottom surfaces.

One or several meshes, not necessarily contiguous, on one or several levels, define a **region**: the space domain where the neutron flux will be represented by numerical approximation. In the same way that meshes define regions, sets of regions define **media**. **Macros** are contiguous groups of regions, for interface-current calculations in APOLLO2. Sets of regions can be grouped into an **output zone**; a region can belong to several outputs. A contiguous output zone can be also used to define a new motive, to be extracted from the main one.

One can associate a reference to any mesh. **Reference** is the name of a motive to be inlayed in the host motive. The external perimeter shape of the reference must be equal to that of the host mesh. Multiple nesting can be done with no limitation on the number of nesting levels. The host and the inlayed geometries should have the same axial heights but not necessarily the same levels. References are frequently used for assembly and reactor core calculations. A large number of geometries can be created by inserting different references in the same host grid.

All the information describing the geometry are stored in objects (like motives, nodes, equations, meshes, levels, regions, macros, media, output zones, boundaries), themselves stored in linked lists.

2.1.1. Regular geometries

Regular geometries are built using **components**: predefined parameterized geometries. They can be created visually or described by a symbolic language using text editor. Components can be generated, modified, assembled into **collections** and saved by Silène, with consistency ensured.

Actually Silène deals with ten types of elementary components: **medium** (homogeneous material), cylindrical, rectangular and hexagonal **cells**, rectangular, hexagonal or hexagons within a rectangle **grids**, rectangular and hexagonal **boxes**, and **motive (**unstructured geometry**).**

Components can be very complex. For all of them, except for medium, boundary conditions can be defined. In addition, according to component types,

- Medium: It can be declared with generic name, which means that name can be extended by a position identifier.
- Cell: The rings can be subdivided into slices of equal volumes and into radial sectors. Elements of windmill form may exist in the corners of the rectangular cell.
- Grid: All tree kinds of grids can be defined with symmetries. Hexagonal grid can have 30, 60, 90 and 180 degrees median or diagonal symmetry and 60,120 and 180 degrees median or diagonal rotation, making total of fifteen different configurations. Rectangular grid can have water gaps, and reflector can be drown around it. Inlayed components can have physical names. Before being inlayed components can undergo rotations and orientation changes and/or be reduced upon symmetries or unfolded.
- Box: Boxes are defined with multiple nested boxes. User may supply either the coordinates or the thicknesses of the boxes and impose different calculation meshes on all boxes.

Final geometry is generated by assembling all the components referenced in a head component.

2.2. Functionalities and Windows system

Windows system and intuitive functionalities allow the user to build data efficiently, minimizing numerical data typing and even mouse clicks. Implementation is realized with a great care to simplify and make secure the users work.

From the 152 menu items of Silène's main window, several dozen frames offer numerous kinds of actions, information or help.

In drawing mode nine specific windows enable the user to start a motive by creating elementary figures: rectangles, triangles, hexagons or ovals. Afterwards **Modify** window, which handles 176 actions on the objects, offer basic modifications, like create, modify, reorder or delete. Other functionalities are more sophisticated: import one motive into another, join or superpose two motives, extract or e one motive from another, cut, trace a grid. Motive can be reduced upon symmetry axes, rotated, translated or rescaled. Built-in utilities help the user: it is possible to verify consistency, to get information concerning different objects, to zoom and color.

In component mode specific windows for each type of component guide the user in generating and modifying components, editing scripts or displaying trees of dependencies.

Specific windows enable user to generate data in different formats for destination codes.

Three levels of help utilities are implemented. A bubble displays brief information when a graphical component in any window is pointed. On help mode, selected in main window, any menu item displays a help window describing its action. On every window the '**?**' button displays a contextual help which explains, depending of context, the action to be done.

3. NEW DEVELOPPEMENTS

All over of its development, Silène has been constantly enriched with new features. As Silène is used for larger and increasingly complex geometrical configurations, the main effort is directed toward developments useful for efficient and secure management of this kind of problems.

For a project calculation, for the same geometry, different representations should be generated for different codes or different methods. Silène should manage data for entire domain for global flux calculations, independent assemblies for flux reconstruction, cells for self shielding, equivalent geometries for post-processing, different fuel media identifiers used for burn-up calculations etc. The code must also manage different variations of this set of geometries, through different mesh refinements.

At every moment of the project, the user should be confident that the software will generate data corresponding exactly to the geometry he had imagined. Satisfaction of this requirement is a fundamental Silène's ambition.

3.1. Variables

In the classical version of Silène the parameters describing components are numerical values or strings. In the component data file the side of a cell, for example, is described as **side: 1.26** (**side:** is a keyword)**.** In the "side" text field, in the related window, the value 1.26 is displayed and can be modified, always as a digital fixed value. If the assembly contains several cells with different content and same size, the variable **side:** must be indicated as many times. If the user wants to change it, he should update descriptions of all cells. In order to facilitate and secure the management of collections containing a large number of components, the concept of variables has been implemented in Silène.

Variables are typed. There are four types of variables: **integers**, **doubles**, **strings** and **components**. In Silène all floating point values are in double precision, so there is no float variable.

Variables are managed the same way as the components. In the data file they are described in blocks. In the **Components** window the variables are listed in the components tree according to their types.

Variables are objects defined with a name and a value. The syntax is: **type: :name value**. The name is a string, preceded by the symbol **:** (colon). The value must be of the type required. Numerical variables can be bounded by a lower and upper bound. A variable declaration for a radius of 0.47 with lower bound of 0.1 and upper bound of **:rmax** will be written as:

VariableD: :radius 0.47 > 0.1 < :rmax ;

By default, the names of parameters defined by variables are displayed in the appropriated windows. To see clearly, the user can switch the display to see the current values of the variables. He can also display a table with a list of names, values and bounds of all variables declared in the collection.

Introduction of elementary operations in the definition of variables is under developpement. The four arithmetic operations should be used in the definition of integer and double variables. Operations of concatenation and reduction should be applied to strings and components.

4. USE CASE: BWR ASSEMBLY

Advantages of the new developments can be illustrated on building data for a Boiling Water Reactor Assembly. It contains a waterhole box, 14 UOX and 77 MOX fuel cells. Fuel in the fuel cells is defined with seven different enrichments, depending on spatial positions. On the left of Fig. 1 the technological mesh which should be built by Silène is shown. The UOX cels are colored orange and MOX cells green. The generated calculation mesh, containing 7064 regions is shown on the right of Fig. 1. Fuel rods with different enrichments are filled with different colors.

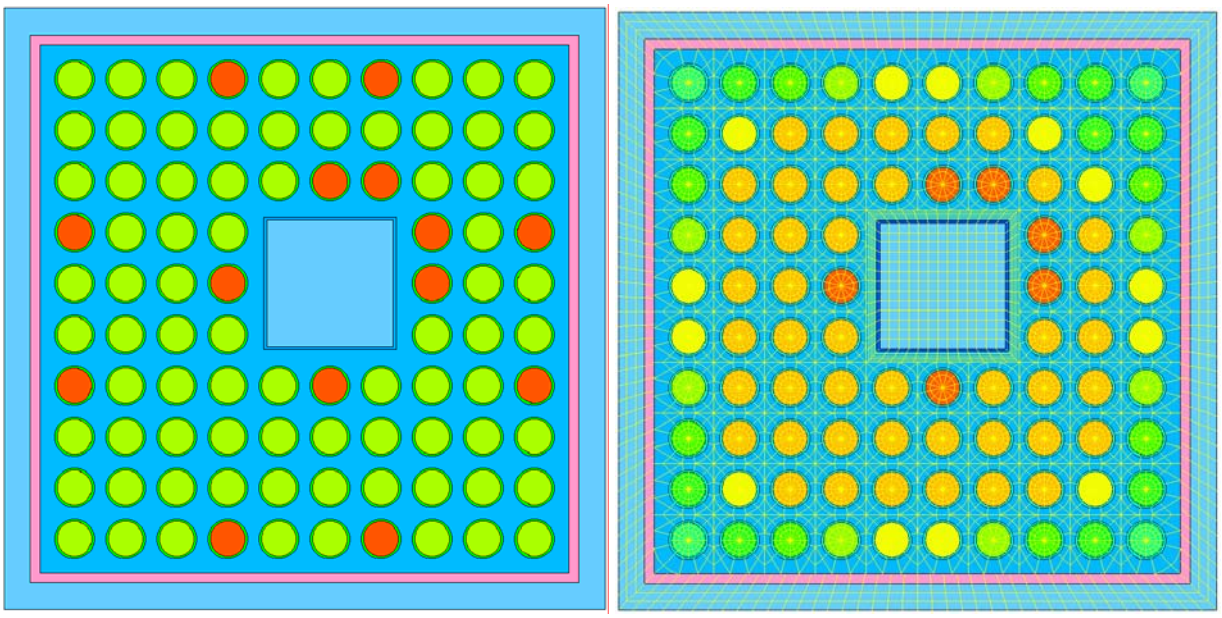

Figure 1. BWR Assembly meshing.

Data for the U cell (UOX) are generated with the Rectangular Cell Component. The most of the parameters are variables, as shown on window fragment in Fig. 2, left. On the right side the same window displays actual values of variables.

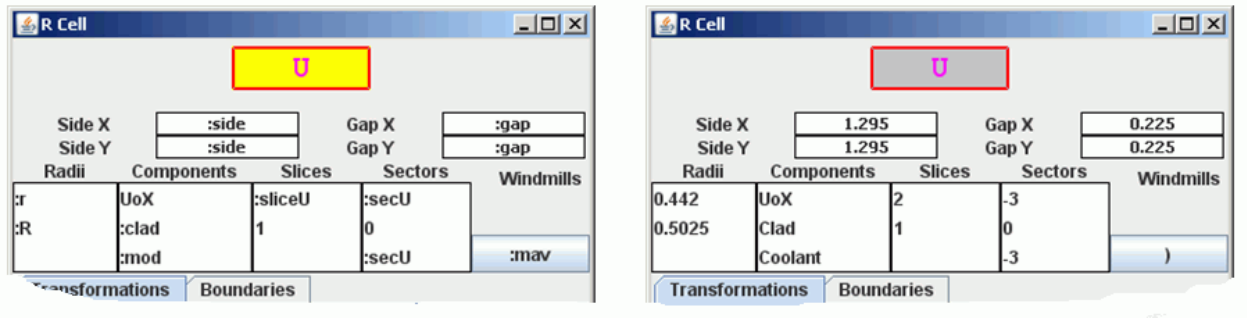

Figure 2. Rectangular Cell Component Window.

The only difference between U and M (MOX) cells is the fuel component. So the most efficient way to define it is by using a mutant component, defined as:

CellR: M inherite: U replace: UoX MoX ;

The different fuel enrichments are defined in the description window of the rectangular grid. UOX and MOX media are defined with the "generic" attribute. In the grid window, as shown on Fig. 3, the cells with different enrichments have different physical names, here the numbers 1 to 7. So from initial UOX and MOX media, media: MoX_1 to MoX_5, UoX_6 and UoX_7 are generated

Figure 3. Rectangular Grid Component Window.

The assembly **Assembly** from the Fig. 3, containing the homogeneous box **Box** and the cells **U** and **M** are placed in the **BWR** box, Fig. 4. Three boxes are nested in **BWR**. The assembly is surrounded by boxes containing **ChanelBox** and **Water** media, which are meshed.

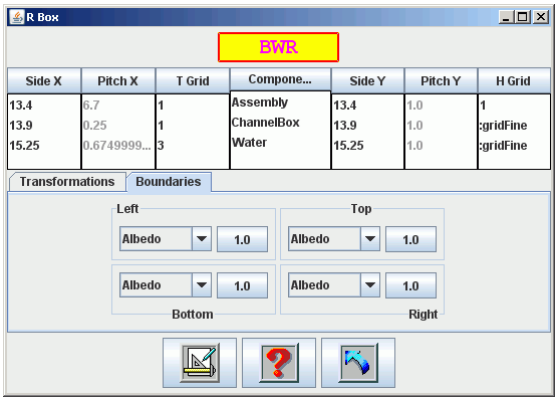

Figure 4. Rectangular box Component Window.

The final result is the geometry shown on the Fig. 1, right.

5. CONCLUSIONS

Initially Silène was designed to generate data that no software could do. Over time Silène was taken to manage very complex sets of geometries and to generate great amount of data. So, actual challenge is to make the software able to handle this kind of problems and also make user confident that generated data match what he wanted to do. The introduction of variables in the declarations of the components, in conjunction with the mutants and the media with generic names, can make work safer and more efficient.

REFERENCES

1. R. SANCHEZ, L. MAO, S. SANTANDREA, “Treatment of Boundary Conditions in Trajectory-Based Deterministic Transport Methods,” *Nucl. Sci. Eng.*, **140**, 23-50 (2002).
2. I. ZMIJAREVIC, "Multidimensional Discrete Ordinates Nodal and Characteristics Methods for the Apollo2 Code," *Proc. Int. Conf. on Math. and Computations, Reactor Physics and Environmental Analysis in Nucl. Applications*, Madrid, Spain, Sept. 1999., p. 1587.
3. S. LOUBIERE, R. SANCHEZ, M. COSTE, A. HEBERT, Z. STANKOVSKI, C. VAN DER GUCHT, I. ZMIJAREVIC ,“ APOLLO2 twelve years later,” *M&C 99*, Madrid, September 27 – 30, 1999, Vol. 2, pp.1298-1315 (1999).
4. J.-P. BOTH, Y. PENELIAU, “The Monte Carlo Code TRIPOLI-4 and its first benchmark interpretations." *PHYSOR 96*, p. C 175, Mito, Ibaraki (Japan) Septembre 1996.
5. J-Y. MOLLER, J-J LAUTARD, “Minaret , A Deterministic Neutron Transport Solver For Nuclear Core Calculations “, *This Meeting,*

6. R.SANCHEZ and Z. STANKOVSKI, "Silène & TDT: A Code for Collision Probability Calculations in XY Geometries," *Proc. 1993 ANS Annual Meeting*, June 20-24, 1993, San Diego, California
7. Z. STANKOVSKI, "Implementation Of Component Concept In Silene 2d/3d Pre & Post Processing GUI," *Joint International Topical Meeting on Mathematics & Computation and Supercomputing in Nuclear Applications (M&C + SNA 2007)* Monterey, California, April 15-19, 2007, on CD-ROM, American Nuclear Society, LaGrange Park, IL (2007)
8. Z. STANKOVSKI, Mutant Components and Scripting in Silene 2d/3d Pre & Post Processing GUI," *International Conference on the Physics of Reactors "Nuclear Power: A Sustainable Resource"* Casino-Kursaal Conference Center, Interlaken, Switzerland, September 14-19, 2008.