International Conference on Mathematics, Computational Methods & Reactor Physics (M&C 2009)
Saratoga Springs, New York, May 3-7, 2009, on CD-ROM, American Nuclear Society, LaGrange Park, IL (2009)

Current Status of the CAD Interface Program for MC Particle Transport Codes McCad

D. Große
Forschungszentrum Karlsruhe, Institut für Neutronenphysik und Reaktortechnik
Postfach 3640, D-76021 Karlsruhe
dennis.grosse@inr.fzk.de;

and

H. Tsige-Tamirat
EC - Joint Research Centre, Institute for Energy, Nuclear Design Safety Unit
PO Box 2, 1755 ZG Petten, Netherlands
haileyesus.tsige-tamirat@ec.europa.eu

ABSTRACT

The program McCad provides a CAD interface for the Monte Carlo transport code MCNP. It is able to convert CAD data into MCNP input geometry description and provides GUI components for visualization, data exchange and modeling. It performs sequences of tests on CAD data to check its validity and neutronics appropriateness including completion of the final MCNP model by void geometry. McCad has been used in several applications like for a 40° ITER torus sector. An insight in the actual status of McCad and an its functionality is given.

Key Words: CAD Interface, Monte Carlo Codes, MCNP, TRIPOLI

1. INTRODUCTION

The Monte Carlo (MC) method for particle transport can handle a given problem geometry in arbitrary detail. Unfortunately, the modeling of a complex geometry for MC particle transport problems is a time consuming and error-prone task. A Computer Aided Design (CAD) interface program offers many advantages over common techniques to build a geometry for MC particle transport. It guarantees consistency between engineering CAD drawings and neutronics models, and faciliates a fast adoption of modifications in the reference model. Furthermore, it allows the exchange of data with other analysis codes.
McCad[1] is an automatic CAD interface program for the MC particle transport code MCNP[2] which is CAD system independent. It allows the bidirectional translation of a given geometry (CAD→ MCNP, MCNP→ CAD). The program performs sequences of tests on CAD data to check its validity and neutronics suitability and implements an algorithm for geometry translation including completion of the final MCNP model by void geometry. It integrates an open source CAD kernel and is conceived as an open source project itself. The graphical user interface (GUI) provides features for data exchange and visualization. Further development efforts are devoted to implement tools for simple modeling and additional support functions, such as data card entries in the MCNP input deck.

2. MATHEMATICAL BACKGROUND

The geometric problem to be solved in MC particle transport can be formulated as a combination of ray classification and point location problem in a decomposed space. The geometry needed for MC particle transport at the algorithmic level is therefore a decomposition of the problem space into a finit collection of disjoint regions (cells) whose union is the problem space. In MC codes this is usually represented by Boolean forms of primitive solids or algebraic half-spaces. The geometry for MC particle transport can be formalized as semialgebraic cell decomposition.
A subset $S \subset \mathbb{R}^n$ is said to be semialgebraic if it can be represented by a Boolean combination of polynomial equations and inequalities with real coefficients:

$$S = \bigcup_{i}^{s} \bigcap_{j}^{r_i} \{x \in \mathbb{R}^n | P_{i,j} *_{i,j} 0\} \tag{1}$$

where $P_{i,j}$ are the polynomials over $\mathbb{R}^n$ and $*_{i,j}$ is either '$<$' or '$=$' for $i = 1, \cdots, s$ and $j = 1, \cdots, r_i$ [3]. The representation of a semialgebraic set is not unique and lacks any geometric information. Every semialgebraic subset of $\mathbb{R}^n$ is the union of finitely many semialgebraic subsets. Primitive solids, algebraic half spaces and any Boolean combination of them are semialgebraic according to the above definition.
The most often used representation scheme in CAD systems is the so called boundary representation (B-rep) scheme. A solid is a compact and regular point set, whose boundary is composed of closed oriented manifolds of dimension up to two. In B-rep solid objects are unambigously defined by their boundaries together with their topological orientation. The boundary ∂S of a solid S is defined by the union over all intersections of the solid S with the algebraic half-spaces $P_1, \cdots, P_l$ supporting the solid.

$$\partial S = \bigcup_{i=1}^{l} (S \bigcap P_i) \tag{2}$$

Because of the differences in the representation schemes a conversion is neccessary.
Given a solid in its boundary representation using algebraic surfaces only, it can be shown that its semialgebraic representation as used in MCNP can be computed[4].

3. CONVERSION ALGORITHM

In general, the boundary support set of B-rep solids is not sufficient for its semialgebraic description except when all polynomials are linear, that is, all faces are planar. A solid's boundary is said to be *definable* if every subset of it can be represented as a semialgebraic set by the available elements of the boundary support set. For manifold solids supported by algebraic half spaces it is always possible to enlarge the boundary support set in order to guarantee definability. Shown in Fig. 1 is an example in two dimensional space of the enlargement of the support set in order to establish definability. The general features of the two-dimensional case remain valid in three dimensions. Figure 1*a* shows the solid 1 (blue number) defined by its four surrounding

surfaces (black) 1 to 4. In B-rep the support set is sufficient to describe the solid. But the solid can not be constructed by Boolean operations on half-spaces defined by the four given surfaces only. In the following the MCNP syntax for cell definition is used, that is, each cell of the geometry is written as a sequence of signed surface numbers together with Boolean operators that specify how the regions bounded by the surfaces are to be combined. A signed surface number stands for the region on the side of the surface where points have the indicated sense. The Boolean operators are denoted by: no symbol means intersection, ':' means union, '#' means complement. As can be seen in Fig. $1b$ the construction of a cell using the four given surfaces only would result solely in the interior of the circle described by surface 1. That is, because the cell would be defined by the intersection of the interior of surface 1, the right half space of surface 2, the upper half space of surface 3 and the left half space of surface 4. In MCNP syntax this would read $(-1\ 2\ 3\ -4)$ which is equivalent to (-1).
To avoid the faulty cell construction the support set $\{1, 2, 3, 4\}$ is enlarged by surface 5. Then the cell can be constructed as the union of $(-1\ 5\ )$ and $(-5\ 2\ 3\ -4)$ which correspond to the cells 1 and 2 (blue numbers) respectively in Fig. $1c$; written in MCNP syntax: $(-1\ 5\ ) : (-5\ 2\ 3\ -4)$.

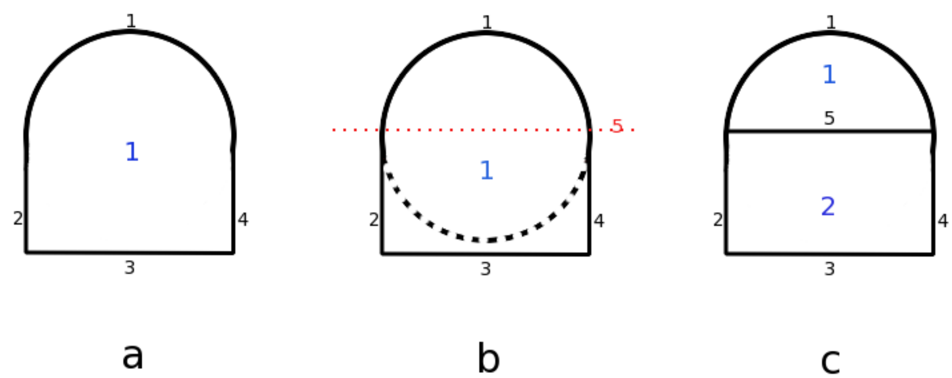

Figure 1. Enlargement of support set to guarantee definability

3.1. Sign-Constant Decomposition

The first stage in the conversion process is to establish definability if not given. This can be achieved by finding additional algebraic boundary supports as described above. For MC particle transport codes, usually only planes, quadrics and tori, which are irreducible monic polynomials are used for cell definition. In this case a defining set can be generated by adding partial derivatives and resultants to the available set. For the present case, both partial derivatives and resultants can be approximated by available data.
In the second stage a semialgebraic description of the cells defining the solid can be constructed. The availability of a defining set of boundary supports allows a sign-constant decomposition of the solids. A surface of a solid is saied to be *sign-constant* if for all points lying on the other supports the evaluation of the surface's equation has the same sign. A solid is saied to be sign-constant if all it's supporting surfaces are sign-constant. Sign-constant solids are convex. The algorithm for the sign-constant decomposition begins with the generation of sample points over all surfaces of a given solid. Next, for every surface all sample points are evaluated with the

surface's equation. If all surfaces are sign-constant the solid is appended to the list of sign-constant solids, otherwise a decomposition is neccessary. The algorithm takes the sign-changing surface with the biggest surface area and cuts the original solid along the selected surface. Instead of using nonplanar surfaces as cut faces their resultants are added to the list of sign changing surfaces as shown in Fig. 1. The resulting solids are put into the list of original solids and the algorithm is restarted. The algorithm finishes when all solids are sign-constant. Figure 2*a* shows a solid which needs to be cut in two order to establish sign-constancy. The decomposition is mathematically ambiguous. The solid could be cut along two surfaces from the support set (shown in Fig. 2*b* and 2*c*).

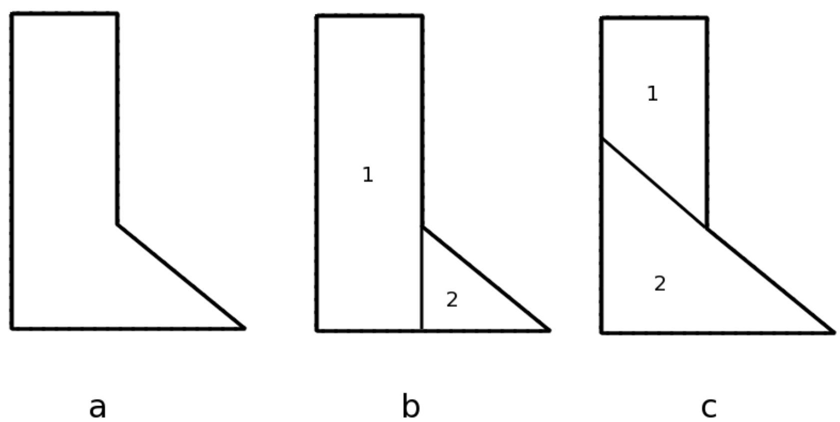

Figure 2. Sign-constant decomposition

The resulting representation of the solid is a union of product terms of the enlarged boundary support set.
The program does not introduce approximations. Therefore, the converted geometry is fully equivalent to the original one, that is, the overall shape and the overall volume is invariant against the conversion process. Shown in Fig. 3 is the relative frequency of the volume ratio of the MC-Cells to the CAD-Solids for the $40°$ ITER benchmark model. The volumes of the cells are calculated by MCNP stochastically. The minor differences are due to stochistical errors and overlaps in the CAD model.
Obviously there is an increase in the complexity of the model in terms of cells and defining surfaces. However, the complexity of the resulting representation is comparable to the complexity of the solid represented by Boolean form of primitives.
Figure 4 shows the Divertor of the Iter benchmark model. The original CAD model is displayed on the left hand side and the sign-constant decomposed model on the right hand side. The original model consisted of 91 solids while the sign-constant decomposition algorithm generated 609 solids which correnspond to the material cells in the MC input file.

3.2. Void Completion

After the conversion process is finished the decomposed model has to be completed by voids and be translated to MC syntax. The generation of voids requires a bounding box which fits the complete CAD model. If no bounding box is provided by the user McCad generates an axis

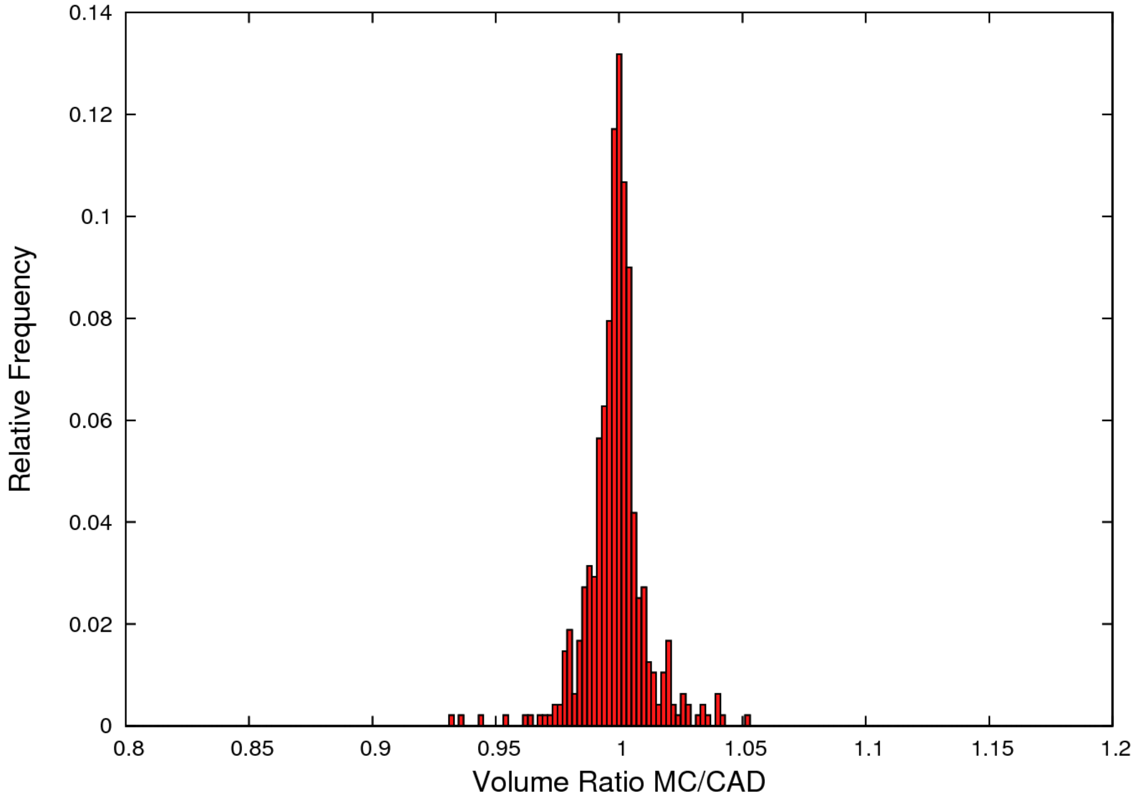

Figure 3. Relative frequency of the volume ratio of MC-Cells to CAD-Solids for the $40°$ ITER benchmark model.

parallel bounding box. This could be unfavorable since more cells could be produced than neccessary. In Fig. 5 one can see in gray a bounding box which has been provided by the CAD designer for a specific problem as a single STEP file. The red wireframed box shows the axis parallel bounding box McCad would have produced for that problem. A problem specific bounding box can consist of multiple convex solids. Some future work will be done to optimize the generation of a tight bounding box.

The void cells are generated as follows. All planar surfaces from all decomposed solids are extracted and added to a list. Redundant surfaces and surfaces with a small area are discarded. Next, all non-redundant planar surfaces are tested for sign-constancy with every solid in the list of bounding box solids. If a plane is sign-changing on a solid from the bounding box the solid is cut along the sign-changing plane. It was chosen to use planar surfaces as cut faces only, because the cutting algorithm provided with the CAD kernel is not reliable for quadrics. Because a CAD model usually also consists of non-planar surfaces, it is neccessary to cut all material cells from the void cells (cutted bounding box). After all planar surfaces are sign-constant for all solids of the bounding box a collision test is performed between all 'void solids' and the material solids. Once all collisions have been detected McCad uses MCNP's complement operator '#' to exclude overlapping material cells from the void cells.

In a final step the MC syntax for the material and void cells is produced and printed to the output file.

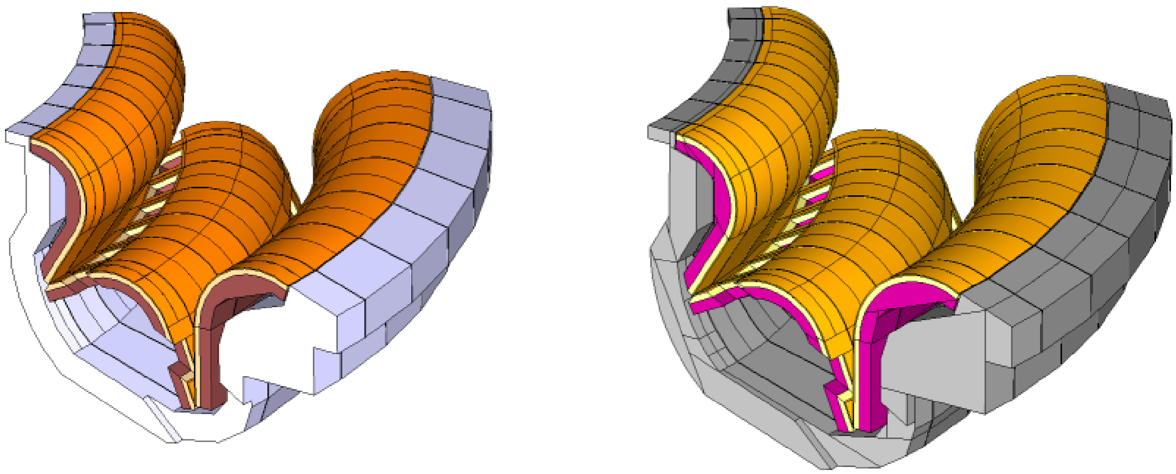

Figure 4. ITER Divertor: original CAD model (left) and sign-constant model after decomposition (right)

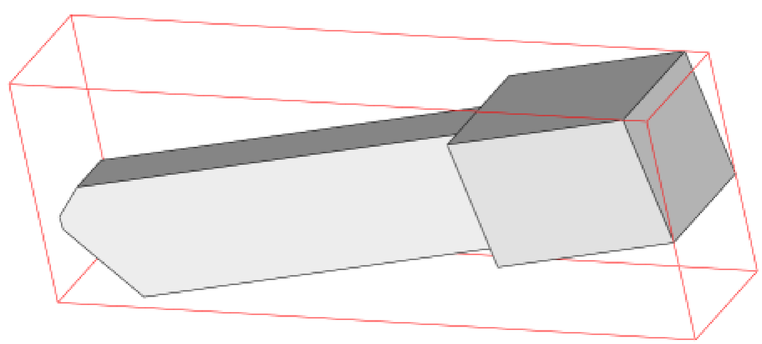

Figure 5. Bounding boxes

4. CONVERSION PROCESS

CAD design models are constructed mainly for visualization and manufacturing purposes and are often over detailed for the need of transport analysis. Also, for pure visualization purposes the need to avoid overlaps does not necessarily arise. Hence, preprocessing steps such as geometric simplification, detail suppression and model repairs are often necessary to generate a suitable model for MC codes. In this case simplification means substitution of free form surfaces, such as b-splines, by algebraic ones. It may happen that, even after detail suppression and replacement of non algebraic surfaces, the model is still too complex and requires a further simplification by manual decomposition. All geometrical simplifications need to be performed manually in the chosen CAD system. Besides the above mentioned preprocessing steps it is up to the CAD

designer to avoid heavy overlaps among solids. Overlaps which are introduced by the CAD system because of tolerance mismatch and limited numerical precision are handled by McCad automatically.
The manual preprocessing is the most time consuming part in the process of geometry conversion. In order to minimize the effort, CAD designers are advised to construct the model such that only acceptable surfaces are used and that detail reduction can be applied quickly.
In the following a summary of the work flow is given: A CAD model, using solely analytic algebraic surfaces, is constructed in the CAD system the designer favors. Usable surfaces are limited by MCNP and the used CAD kernel [*]. For example MCNP supports elliptical tori which cannot be handled by the Open CASCADE (OCC) CAD kernel. The only algorithmic limitation is that the model must represent a manifold solid, or an assembly of manifold solids, given in the B-rep data structure. The CAD model can be read by McCad via the neutral file formats STEP and IGES, which are supported by most commercial CAD systems. Both file formats are able to transform B-Rep data structures accurately, though STEP is known to be of higher quality.
After the import, McCad automatically performs geometrical suitability and error checks and repairs faulty segments if possible. Suitability check are limited to geometric properties, that is, the check if the boundary supports are algebraic. Then the imported model is visualized and can be modified by the user via McCad's GUI (see section 5). From an open shell one can start the conversion process directly without using the GUI. Geometry CAD files and process parameters can be given to McCad via an input file. In GUI mode the geometry can be exported to the MC geometry scheme or further modifications can be applied first. The translation beginns with a check for overlaps between boundary entities and for solids which are too small to be taken into account in particle transport calculations. The collision check gives a feedback on colliding solids. The check follows the sign-constant decomposition as described above. The geometry conversion ends with another check for overlaps among the decomposed solids. Possible overlaps could arise because Boolean operations on CAD solids are performed within the range of some tolerance. It is assumed that occuring overlaps are insignificantly small, that is, that they do not have an effect on the outcome of the MC calculations. According to this assumption McCad uses the MCNP's complement operator '#' to exclude one overlapping solid from the other. The decision which solid will be complemented is done automatically.
In CAD systems the void space usually is not defined. For MC transport codes, on the other hand, the whole problem space needs to be defined. After the decomposition process, McCad automatically completes the model with voids as described above, translates the geometry description to MCNP syntax and writes it to the output file.

5. THE INTERFACE PROGRAM McCad

McCad is realized in a framework like library based on object orientated design patterns. It is programmed for Linux platforms using the programming language C++. The program implements the Open CASCADE CAD kernel and the Qt4 [†] libraries for its GUI. Since both Qt4 and Open CASCADE are available for Microsoft Windows also, there might be a version in future which supports Microsofts operating systems, too.
The CAD kernel provides core data structures, algorithms and data exchange interfaces for neutral CAD files (STEP/IGES). Since McCad reads and writes neutral CAD files which are

[*] www.opencascade.org
[†] www.trolltech.com

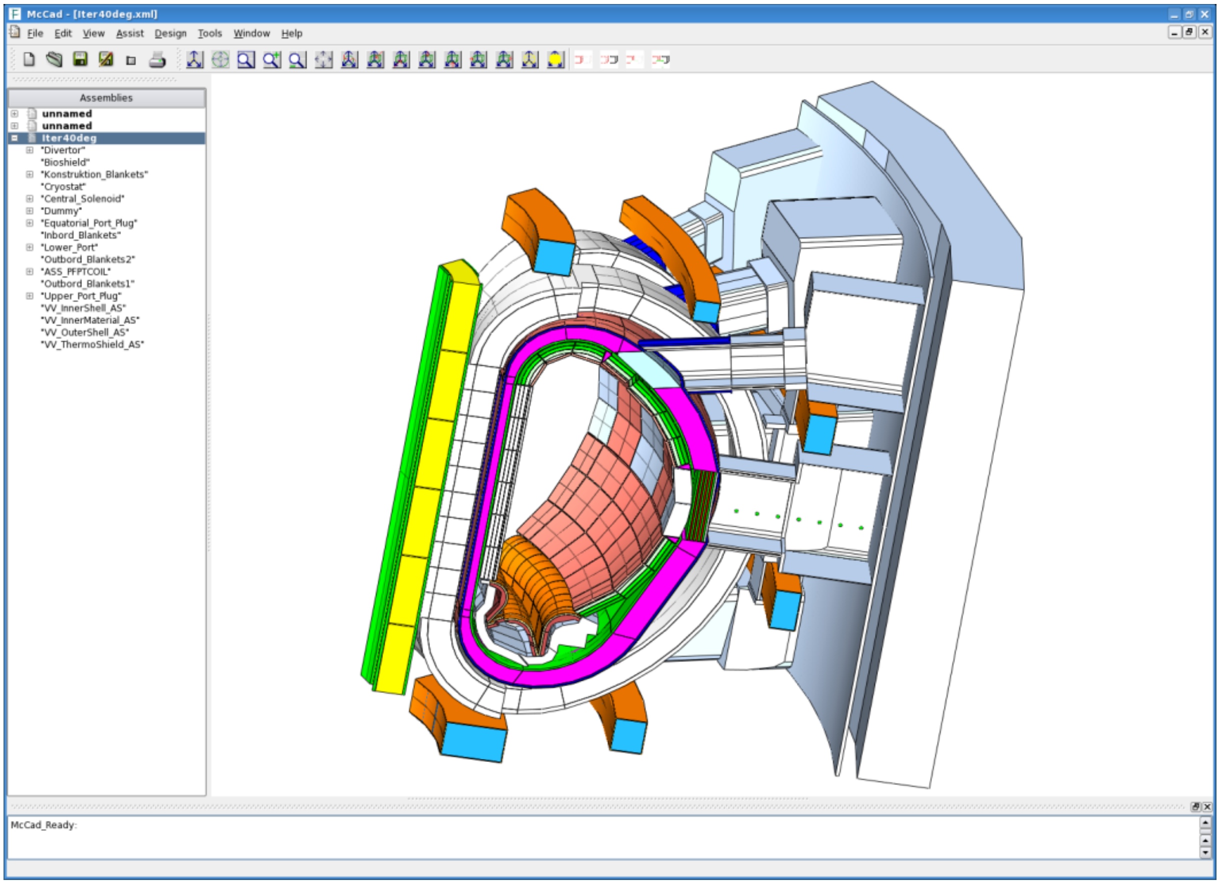

Figure 6. McCad GUI showing the $40°$ ITER benchmark model.

supported by most commercial CAD systems, it does not depend on the CAD system used for modeling. The conversion algorithm relies on the CAD kernel for its geometric and related computations. The Model-View-Controller (MVC) pattern is used to divide the interface into three components: Model contains the core functionality and data, View displays information to the user and Controller handles user operation.
McCad is conceived as open source software and will be made available in 2009.

5.1. Current Status

At present the general functions for geometry translation have been implemented, tested and are accessible via the GUI (see Fig. 6). The document management is based on OCC's Application Framework (OCAF) which provides mechanisms for saving and loading the document structure (McCad file extension *.mcd) and to undo and redo applied commands. McCad is an multiple document interface program which allows to work on several open documents simultaneously. That way the exchange of geometry and data from one model to the other can be done quickly via copy and paste. Besides the open and save functions, which exchanges the complete document structure, McCad offers import and export functionality to exchange geometry data only. The conversion process to MC geometry is started via the export function. As mentioned before the

conversion can also be run from an open shell in non GUI mode.
McCad allows the translation of geometry schemes from CAD to MCNP and vice versa, as well as their three dimensional visualization. The conversion from MCNP to CAD is not yet fully available. Translation of repeated structures are not implemented, also the surface types: general quadrics (*GQ*), tori (*TX, TY, TZ*) and surfaces defined by points (*X, Y, Z, P*) are not yet implemented in the MCNP reader.

5.2. Applications

McCad has successfully been used in different tasks with complex geometries (number of cells in the order of 1000) in the area of fusion technologies such as for JET[5], ITER[6] and DEMO[7]. It allows to define initial cell and surface numbers and to assign material numbers and densities to solids. Therefore it is possible to convert new modeled CAD parts for an already existing MCNP model to MCNP geometry data and to implement it in the existing MC model in a short amount of time, as has been done at FZK[8].

5.3. Future Work

The current development efforts aim to finalize the GUI for a first open beta release. In future releases McCad will provide: basic CAD features, that is, adding of new primitive solids and appliance of transformations and Boolean operations to solids (this feature is in progress); advanced geometry features for MCNP, such as repeated structures (planned); data card features, such as material specification and tallies (planned); properties tools to measure distances and angles and to show solid properties.
Furthermore McCad will be extended in order to support the MC transport code TRIPOLI[9] in 2009.

6. CONCLUSIONS

The CAD interface program McCad is presented. The method McCad uses is CAD system independent and uses neutral files for data transfer. An algorithm for the conversion of a CAD geometry into a representation appropriate for MC particle transport has been described.
McCad implements the conversion algorithm itself and a graphical user interface which allows data exchange, visualization and to apply modifications to a given model. The data exchange allows the conversion from CAD to MC data and vice versa.
The use of an interface program such as McCad can significantly reduce the effort needed for the geometric modeling of complex parts for MC transport codes. McCad will be freely available in 2009.

ACKNOWLEDGEMENTS

This work has been performed in the framework of the nuclear fusion program of Forschungszentrum Karlsruhe and is supported by the European Union within the European Fusion Technology Program.

REFERENCES

[1] H. Tsige-Tamirat and U. Fischer, *CAD Interface for Monte Carlo Particle Transport Codes*, The Monte Carlo Method: Versatility Unbounded in a Dynamic Computing World, Chattanooga, Tennessee, April 17-21, 2005, on CD-ROM, American Nuclear Society, LaGrange Park, IL (2005).

[2] J.F. Briesmeister (ed.), *MCNP - A General Monte Carlo N-Particle Transport Code, Version 4C*, LANL, Report LA-13709-M, 2000.

[3] J. Bochnak, et al., *Real Algebraic Geometry,* Springer Verlag, 1998.

[4] H. Tsige-Tamirat *On the Use of CAD Geometry for Monte Carlo Particle Transport*, Proceedings of the Monte Carlo 2000 Conference, Lisbon, 23-26 October 2000, Springer Verlag, 2001.

[5] H. Tsige-Tamirat, et al., *Automatic generation of a JET 3D neutronics model from CAD geometry data for Monte Carlo calculations*, Fusion Eng. Des., Volumes 75-79, Nov. 2005, pg.891-895, Proceedings of the 23rd Symposium on Fusion Technologies - SOFT 23

[6] H. Tsige-Tamirat, et al., *Use of McCad for the conversion of ITER CAD data to MCNP geometry*, Fusion Eng. Des. (2008), in press, doi:10.1016/j.fusiongdes.2008.07.040

[7] U. Fischer, et al., *Neutronics design analysis of fusion power reactors based on a novel methodological approach*, Symposium on Fusion Technologies, Rostock, 2008, SOFT 25.

[8] A. Serikov, et al., *Nuclear safety and waste management aspects of the EP ECRH upper launcher for ITER*, Fusion Engineering and Design (2009), doi: 10.1016/j.fusengdes.2008.11.077

[9] J.P. Both, et al., *A Survey of TRIPOLI-4*, Proceedings of the 8^{th} International Conference on Radiation Shielding, Arlington, Texas, USA, 24-28 april, 1994, pp. 373-380.