A link between CAD models and physics codes

M. Telenta[1], L. Kos[1], R. Akers[2], I. Lupelli[2] and EUROfusion MST1 Team *

[1]*University of Ljubljana, Mech. Eng., LECAD, Ljubljana, Slovenia*

[2]*CCFE, Culham Science Centre, OX14 3DB Abingdon, United Kingdom*

The scientific workflow for which the CAD model is read or modelled, *meshed* and then stored as an input to be used by the code should be made easier and more efficient for retrieval and reuse with emphasis to data provenance. A link between the CAD models and physics codes is being developed and will provide a programmable access within the scientific workflows that will be more flexible regarding the input controls and scenarios for different tokamak machines. In this paper we describe the process of CAD geometry programmable modelling from simple to complex geometries organized in different *levels of details*. Furthermore, meshing of the geometry for visualisation purposes and storage of the mesh and attributes in a general grid description (GGD) is provided automatically. 3D codes for RWM, heat flux, fast particle simulations, and 2D SOL, or core can be directly linked to the presented CAD interface library.

Tokamak CAD programmable modelling

CAD model of the tokamak is programmably modelled using the OpenCASCADE CAD kernel. This approach is used for exploration purposes of cataloguing the CAD model complexity in different level of details. The main source of the CAD data is from the proprietary software used for the machine modelling. Thus, the input for the CAD service library is a STEP file exported either from the proprietary or the open source software if available. Also, proprietary CAD files can be read by the OpenCASCADE if the licence is purchased.

OpenCASCADE is an open source CAD kernel that consists of reusable C++ object libraries and offers programmable flexibility in modelling and meshing of a CAD models, and various operations related to the CAD data. OpenCASCADE is a software development platform which provides services for 3D surface and solid modelling, CAD data exchange, and visualisation [1]. These are the main reasons for implementing the OpenCASCADE kernel in the workflow where CAD creation (if needed), exchange, operations and visualisation is serviced. All these processes are used in creating the link between the CAD model and the physics codes. OpenCASCADE offers the possibility to develop an inexpensive tool to be used in the presented workflow.

In this paper the CAD model is reverse engineered in order to programmably capture the

*See http://www.euro-fusionscipub.org/mst1

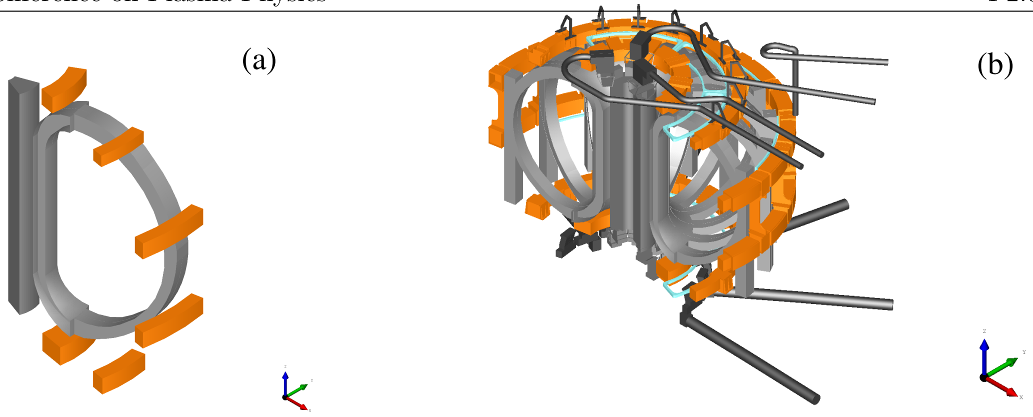

Figure 1: CAD modelling of the magnet segment with different level of details and number of repetitive pieces (a) simplest geometry with one repetitive piece, (b) more complex geometry with ten repetitive pieces.

various CAD details and organise it into different levels of detail. The different levels of details are created in the process of programming the model geometry where different CAD model complexities are classified in different level of details. This is the opposite approach to the defeaturing workflow of a complex model [2] where manual reduction of the CAD features are done by simplifying and removing the details. The complexity of the model in a given level of detail is defined by the complexity of the features and number of the repetitive pieces used to create the CAD model.

As one can see from Fig. 1, the CAD complexity of the tokamak magnet segment is divided into three level of details. Also, each of the level of detail can have different number of repetitive pieces, subjected to the needs of the physics code in question. This CAD modelling principle is taken in modelling the rest of the tokamak machine segments, such as divertor, blanket, vessel, and cryostat. [3]

CAD connection to the physics codes

In this contribution we report on a CAD service library presented in Fig. 2 which presents a link between the CAD data and the physics codes. The CAD service library serves the physics codes with appropriate data by a specific request. Variety of the data could be offered depending on the physics code in question, such as STEP file for 3D geometry, cross sections of the tokamak in human readable format such as CSV, basic mesh, surfaces of specific machine components, etc.

CAD geometry in neutral format such as STEP is used in codes which are using the 3D geometry for various calculations, such as COMSOL [4] and ANSYS [5], [6] for electromagnetic and CFD numerical simulations, respectively. The STEP format is the most commonly

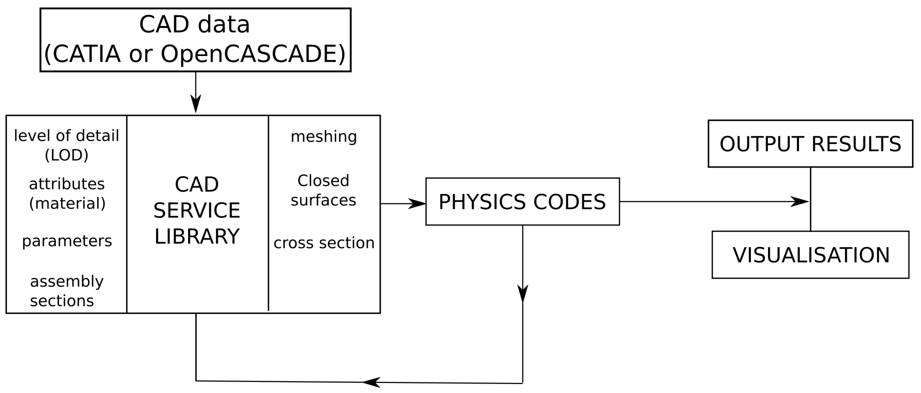

Figure 2: A CAD service library in a workflow provides various services to the physics codes

used standard data files format for CAD data exchange.

The storage of the CAD data itself differs. Integrated Data Access Management system (IDAM), [7] stores the CAD data from CATIA in STEP file format. The access to the CAD file is through a plug-in which stands between the user and the CAD database. The user only gives the abstracted name of the CAD file and the job of the plug-in is to locate the file and serves to the user. On the other hand, EU-IM database stores the CAD geometry in a mesh format. The CAD service library could create simple meshes such as triangles in 2D-space or tetrahedron in 3D-space and writes the mesh as general grid description (GGD) in the database.

The CAD service library can serve codes such as ASCOT with the first wall as a limiting surface to accurately represent the wall structures.The surface is than meshed with triangles which are used by the code to calculate the collision detection [8].

Cross-section service of the CAD library is useful for codes which need 2D geometry of a particular axisymmetric section-cut with different level of details. The physics codes which use 2D section cut as an input are SOLPS code [9] , EFIT++ [10], etc.

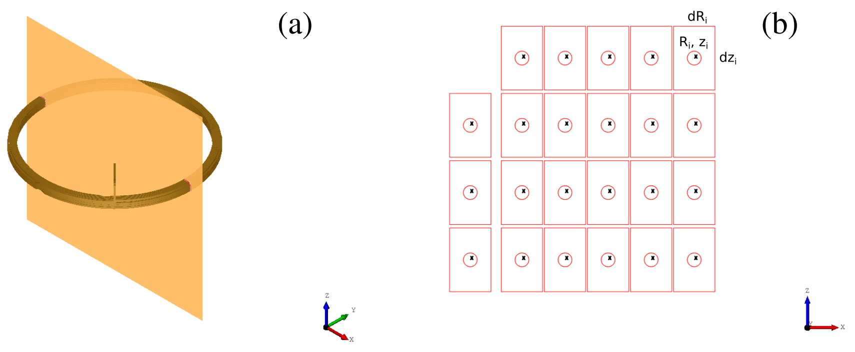

Figure 3: CAD model of one P-coil with axisymmetric section cut (a) 3D model intersected by a cutting plane with a given angle, (b) 2D section cut of one P coil with extruded centre coordinates (R, z) and side lengths (dR, dz) for each coil filament in the P coil.

In Fig. 3 the 3D model of one P-coil is shown, intersected by a cutting plane in order to get axisymmetric section. This cross-section is used as an input for, for example, EFIT++ code. The coordinates of the geometrical centres (R, z) of each coil filament together with the side lengths (dR, dz) are extruded and written in a CSV formatted file. This file is then read by a Python script that converts the data into XML format used by the EFIT++ code.

Conclusion and future work

A link between the CAD models and physics codes has been developed by creating a CAD service library. This library will serve the physics codes with appropriate data, such as 3D STEP files, first wall meshed with triangles, 2D axisymmetric sections cuts, basic mesh for storage and visualisation. The future work will be focused in expanding the CAD service library with other services as the list of physics codes which would benefit from this library grows.

Acknowledgement

This work has been carried out within the framework of the EUROfusion Consortium and has received funding from the Euratom research and training programme 2014-2018 under grant agreement No. 633053, task agreement AWP15-EEG-JSI/Telenta. The views and opinions expressed herein do not necessarily reflect those of the European Commission. The authors would like to acknowledge to many students from the Faculty of Mechanical Engineering at the University of Ljubljana that have contributed to this work.

References

[1] “OpenCASCADE – OpenCASCADE technology,” `http://opencascade.org` (2015).

[2] T. Kurki-Suonio, S. Äkäslompolo, K. Särkimäki, S. Sipilä, A. Snicker, E. Hirvijoki, O. Asunta, T. Koskela, and M. Gagliardi, in *41st EPS Conference on Plasma Physics* (Berlin, Germany, 2014) p. P5.017.

[3] M. Telenta and L. Kos, in *Distributed Computing and Cloud computing* (Opatija, Croatia, 2016) pp. 707.1 – 707.8.

[4] S. Äkäslompolo, O. Asunta, T. Bergmans, M. Gagliardi, J. Galabert, E. Hirvijoki, T. Kurki-Suonio, S. Sipilä, A. Snicker, and K. Särkimäki, Fusion Eng. Des. **98 - 99**, 1039 (2015).

[5] M. Benedetti, P. Gaudio, I. Lupelli, A. Malizia, M. T. Porfiri, and M. Richetta, Fusion Eng. Des. **88**, 2665 (2013).

[6] I. Lupelli, P. Gaudio, M. Gelfusa, A. Malizia, I. Belluzzo, and M. Richetta, Fusion Eng. Des. **89**, 2048 (2014).

[7] D. Muir, L. Appel, N. Conway, A. Kirk, R. Martin, H. Meyer, J. Storrs, D. Taylor, N. Thomas-Davies, and J. Waterhouse, Fus Eng. Des. **83**, 406 (2008).

[8] E. Hirvijoki, O. Asunta, T. Koskela, T. Kurki-Suonio, J. Miettunen, S. Sipilä, A. Snicker, and S. Ävkäslompolo, Computer Physics Communications **185**, 1310 (2014).

[9] A. Kukushkin, H. Pacher, V. Kotov, G. Pacher, and D. Reiter, Fusion Eng. Des. **86**, 2865 (2011).

[10] I. Lupelli, D. Muir, L. Appel, R. Akers, M. Carr, and P. Abreu, Fusion Engineering and Design **96-97**, 835 (2015), Proceedings of the 28th Symposium On Fusion Technology (SOFT-28).