New open computational resource for plasma processes modelling

A S Zhilkin$^{1,3}$, D Yu Sychugov$^1$, L I Vysotsky$^1$, I V Zotov$^1$, S Yu Soloviev$^1$ and A D Sadykov$^2$

$^1$Faculty of Computational Mathematics and Cybernetics (CMC), Lomonosov Moscow State University, Moscow, Russia
$^2$Institute of Atomic Energy, Kurchatov, Republic of Kazakhstan

E-mail: alkszh@yandex.ru

Abstract. The paper describes a new open access computing resource nfusion.cs.msu.ru. The resource includes modules for calculating equilibrium, vertical stability, plasma evolution, simulation systems for magnetic diagnostics, as well as a new algorithm for constructing three-dimensional tetrahedral meshes in areas of complex structure. The modules are integrated into a unified software environment designed for numerical support of experiments on tokamak installations. The possibility of accessing the calculation modules on the server via the Internet, data exchange and the issuance of calculation results in the form of files, figures, and tables has been implemented. The resource supports the simultaneous operation of several users and has an information support system in Russian and English.

1. Introduction
A number of new generation tokamak installations are currently under construction in the world. Optimizing their design and numerical support of experiments on them is impossible without mathematical modeling. Hundreds of computational codes have been created and are being used for purposes of simulation of all processes in plasma. These codes were implemented at different times by various scientific groups and, as a result, they use different input and output formats which are not well suited for integration into a single system. An important modern task is to create an integrated system based on these codes, which would make it possible to simulate the entire life cycle of Tokamak installations, including the design stages, physical start-up and numerical support of experiments. The authors of this article were involved in the development of such a system, and its current state is described here.

2. The principle of building a unified software environment
The starting point for developing a unified environment is the discharge scenario in the installation. The scenario itself consists of the following six stages: preparation of breakdown, breakdown itself, current rise, formation of a separatrix (in installations with a divertor), reaching a quasi-stationary state, and discharge decay (current shedding). To fully support experiments, it is necessary to be able to numerically simulate all of the above stages. In addition, for the development of projects for neutron sources and reactors, it is necessary to calculate the neutron fluxes and their impact, which requires

---

To whom any correspondence should be addressed.
three-dimensional calculations, at the beginning of which effective algorithms for constructing three-dimensional grids must be involved. Further, at all stages of the discharge, the occurrence of so-called disruptions is possible, which can lead both to a drop in the temperature and plasma density (small disruptions) and to the death of the entire discharge (large disruption). To prevent breakdowns, it is necessary to control the shape and position, as well as a number of internal parameters of the plasma column. This leads to the need to include simulating diagnostic and control systems in an integrated environment. Thus, the set of modules that provide full mathematical support for experiments should include: a code for calculating the MHD equilibrium and plasma evolution, a code for calculating its stability (otherwise the discharge scenario may turn out to be unrealizable), a code for calculating the energy and particle balance (transport calculations), a module for construction of three-dimensional grids, necessary for calculating neutron sources and reactors, as well as simulators of diagnostic systems, plasma cord control, and operation of the magnetic system of the installation. These computational codes, which were written by different research teams at different times, must be combined into a unified software environment with a common interface, a system for automated data exchange between modules, and a means of issuing results. It is also necessary to have a unified information system and instructions for working with modules. It should also be noted that the software and hardware must support multi-user mode.

3. General structure of a unified software environment

System requirements determined the set of modules included in it [1-10]. As the system continues to evolve, it turned out to be advisable to separate its information and reference part, located at plasma-fusion.ru, and its functional part, which is located at nfusion.cs.msu.ru. The functional part of the system is implemented according to the "client-server" principle.

The server is a high-performance workstation. The server part of the system is responsible for delimiting access to data and computing resources by different users, for launching computing modules and interacting with them, as well as for converting input and output data for transmission to the client. The server side of the current version of the system uses the Apache HTTP server, which refers to programs in Python, which, in turn, run computational modules written in various languages (C, C++, Fortran, Java, etc.) and compiled into executable files (or bytecode in the case of Java).

Client devices can be desktops, laptops and even smartphones. The client part of the system implements the display of the user interface, including graphs, tables and figures, as well as interactive interaction with them. The client side uses only web technologies: the graphical interface is implemented in HTML and CSS, and logical dependencies in JavaScript.

The adopted approach allows not to deploy a special software environment to work on the client device: a modern web browser is enough. The interaction between the client and the server is carried out via the HTTP protocol, the main data formats are XML and JSON.

The following modules are currently open to the public in his unified environment.

TOKAMEQ (TOKAMak EQuilibrium) - calculation of MHD plasma equilibrium in a tokamak. It is based on the algorithm of the numerical Grad-Shafranov equation. A detailed description of the algorithm and instructions for working with it are given in [2], as well as in the informational part of the resource. Examples of using the code can be found in [11-12].
TOKSCEN (TOKAMak SCENario) - calculation of plasma evolution in a tokamak. It is based on an algorithm for the numerical solution of the equations of plasma evolution in a tokamak together with the Kirchhoff equations for conductive elements of the installation structure (coils of an external magnetic field, passive and active feedback loops, etc.). A detailed description of the algorithm and instructions for working with it are given in [1] and in the information part of the resource.
RPB (Reconstruction of Plasma Boundary). RPB code is intended for magnetic diagnostics of the plasma boundary. Its mathematical basis is the solution of a system of integral equations of the first kind by the regularization method. The RPB code was used to analyze and optimize the magnetic diagnostics system of the T-15M facility (Russia, NRC Kurchatov Institute) [11-12]. Currently, a high-performance parallel version of the code [13] based on graphic processors (GPU) has been developed, implemented in the Cuda language using the CuBlas linear algebra library. This version of the code is capable of reconstructing the plasma boundary in the online experiment mode (in a time of the order of a millisecond).
Figure 3. Results of reconstructing the plasma boundary in the T-15M facility (NRC Kurchatov Institute, Russia) using the RPB code included in the unified software environment. Solution of the direct problem of MHD equilibrium.

Figure 4. Results of reconstructing the plasma boundary in the T-15M facility (NRC Kurchatov Institute, Russia) using the RPB code included in the unified software environment. Solution of the inverse problem of MHD equilibrium.

The MESHER code is a new algorithm for constructing tetrahedral meshes in areas with a complex structure and shape [10]. The algorithm is highly efficient: for example, to build a grid consisting of about 1 million cells, it takes about 10 seconds for a standard personal computer to operate. It’s remote implementation is currently limited by available tools for visualization in browser – JavaScript is a single thread language at its core and can’t support practical visualization of this much data so visualization of initial upload and resulting mesh is limited to static images at this time.
4. Technologies for integrating modules into a unified software environment

Nowadays, adding new modules to the system is a creative process, sometimes requiring considerable effort. The fact is that, as mentioned above, computational codes are written in different languages and use different data formats and libraries for rendering graphical user interfaces. At the same time, the accumulated experience allows to formulate some general stages of the inclusion process.

First of all, you need to change the original program, transforming it into a console application with a unified format of input and output data. Typically, data is read from and written to files specified in command line arguments. Secondly, the server side is adapted. As a rule, here you need to implement an HTTP interface with commands like “save input data”, “run code” and “download output data”. In the third stage, the client part is implemented. This includes parts of the graphical interface specific to a particular computational code, displaying and providing the ability to download graphs, figures, tables and other files. It also implements the logic of interaction with the server and reactions to user actions.

As the system was designed, it became clear that a significant part of the program code could be moved into separate libraries for reuse. After the creation of the libraries, the development of the server and client parts of the new modules became much easier, as did the support of the system as a whole. The most difficult task is to adapt the computational code itself to a unified software environment. It is also quite troublesome that the "development paths" of the original and adapted codes inevitably diverge, which leads to the need to manually update the adapted version when updating the original code. A possible solution to this problem is to divide the program into the actual computing part and the part responsible for displaying the results. In this case, at the adaptation stage, it is possible to replace the part of the code responsible for visualization and development of the
computing part can go on as usual. Of course, to implement this idea, the authors of the code must either comply with a certain (currently not formalized or even formulated) standard or work closely with the developers of a unified software environment.

5. Automated data exchange technology
One of the central problems in the development of such integrated systems is the organization of data exchange. The point is that computational modules have different origins and, as a result, have different input data file formats. Thus, the task of transmitting data arises. A typical manifestation of the problem is the need to transfer the computational results from one module to another, in which they are used as input data. If a unified software environment consists of a small number of modules, which are implemented according to the same standard, then the problem seems far-fetched. However, as the number of modules grows, the number of pipes for the data flow between them grows almost exponentially. So a specific approach is required to tackle this problem. Let's list the standard ways to solve the problem.

1) A combination of automatic and manual exchange methods. This method is suitable for a small number of calculations and is unacceptable for mass calculations.

2) Introduction of a rigid standard for input and output data files. This method is justified for engineering calculations with computational codes which are finalized and perfected. But it falls short in case of scientific calculations, especially when it is necessary to modify the models embedded in the computational algorithms.

3) Implementation of specialized adapter programs for data exchange. The method is justified when the global exchange problem is reduced to the simpler task of setting up data exchange between two modules. The disadvantages of adapter programs in many respects repeat the disadvantages of standardization of exchanges, and, in addition, the number of such programs grows quadratically along with the number of modules.

In a unified software environment we are developing an alternative way of data exchange based on the existence of a significant number of "global" concepts, which are used by the creators of different modules. For tokamak installations, such concepts are the number of coils of the poloidal field, their location, size, number of loops in the coils, geometry and material of the first wall, types of sensors, their location and measurement accuracy, etc. The same "global" characteristics are the total current in the plasma, the value of the beta parameter, the inductance of the plasma cord and the like. Therefore, when creating data files, it is enough to provide each variable, group of variables or array with a corresponding “meaning indicator”. Thus, data reading for each calculation module can be reduced to a set of actions: “read the geometry of the magnetic coils”; “read geometry and material of the first wall”, etc. Implementation of such an idea can fully automate the exchange of data between codes.

6. Conclusions
The described software environment combines: (a) calculation modules of the processes occurring in plasma; (b) means allowing to work with them in the remote access mode; (c) means of data exchange between modules; (d) and information block.

The developed software is an open resource that allows you to fully study the designs of tokamak installations and numerically simulate experiments on these installations. Although this system has a narrow focus on tokamak installations, the principles embedded in it can be applied in the development of other virtual analogues of complex technical devices.

Acknowledgements
This work was supported by the RFBR grant no. 20-07-00391.

References
[1] A.D. Sadykov, D.Yu. Sychugov, G.V. Shapovalov, B.Zh. Chektybaev, M.K. Skakov and N.A. Gasilov. The numerical code TOKSCEN for modelling plasma evolution in tokamaks.
Nuclear Fusion, v. 55, Number 4, 55 043017, doi:10.1088/0029-5515/55/4/043017.

[2] Sychugov D.Yu. 2008 Probl. Atomic Sci. Technol. Ser. Thermonuclear Fusion, v. 31, No.1, pp.85-89.

[3] Sychugov D.Yu., Amelin V.V., Gasilov N.A. 2010 Probl. Atomic Sci. Technol. Ser. Thermonuclear Fusion, v. 33, No.3, pp.46-49.

[4] Sadykov A.D., G.V. Shapovalov G.V., Chektybaev B.Zh., Sychugov D.Yu., Gasilov N.A. 2013 Probl. Atomic Sci. Technol. Ser. Thermonuclear Fusion, v. 36, No.4, pp.94-101.

[5] Khayrutdinov R.R., Lukash V.E. 2009 Probl. Atomic Sci. Technol. Ser. Thermonuclear Fusion, v. 32, No.3, pp.57-59.

[6] Khayrutdinov R.R., Lukash V.E. 2010 Probl. Atomic Sci. Technol. Ser. Thermonuclear Fusion, v. 33, No.3, pp.50-54.

[7] Khayrutdinov R.R., Lukash V.E. 2011 Probl. Atomic Sci. Technol. Ser. Thermonuclear Fusion, v. 34, No.3, pp.88-92.

[8] Khayrutdinov R.R., Lukash V.E. 2012 Probl. Atomic Sci. Technol. Ser. Thermonuclear Fusion, v. 35, No.3, pp.93-96.

[9] Zotov I.V., Belov A.G. 2014 Probl. Atomic Sci. Technol. Ser. Thermonuclear Fusion, v. 37, No.1, pp.97-102.

[10] Zhilkin A.S., Sychugov D.Yu. A New Algorithm for Generating Tetrahedral Grids. Moscow University Computational Mathematics and Cybernetics, Allerton Press Inc. (United States), v. 42, n. 2, pp. 63-68.

[11] Sychugov D.Yu, Zotov I.V., Kasyanova N.V., Melnikov A.V., Dokuka V.N., Lukash V.E., Khayrutdinov R.R., Tsaun S.V., Sadykov A.D. Analysis of initial stage of the discharge in the T-15 tokamak – 44rd EPS Conference on Plasma Physics, Belfast, Northern Ireland, June 26-30, 2017, Europhysics Conference Abstracts (ECA), European Physical Society (Switzerland), v.41F, p.P1.142.

[12] Sychugov D.Yu, Zotov I.V., Kasyanova N.V., Melnikov A.V., Sushkov A.V., Sadykov A.D., Shapovalov G.V.Analysis of discharge scenarios in the T-15 tokamak – Journal of Physics: Conference Series, Institute of Physics (United Kingdom), 2017, v. 907, p.012011.

[13] Zotov I.V., Vysotsky L.I. High performance version of RPB code for determination of the plasma boundary in tokamak based on graphic processors – Journal of Physics: Conference Series, Institute of Physics (United Kingdom), 2019, v. 1383, p.012010.