IDM UID
R463HW

VERSION CREATED ON / VERSION / STATUS
19 Mar 2015 / 1.1 / Approved

EXTERNAL REFERENCE

Technical Specifications (In-Cash Procurement)

Technical Specifications_Development of user-friendly interface for SOLPS-ITER

Technical Specifications

ITER_D_R463HW

Development of user-friendly interface for SOLPS-ITER

Technical Specifications

	Version 1	*Date: 11/03/2015*
	Name	*Affiliation*
Author	**X. Bonnin**	**POP**
Reviewer	**R. A. Pitts**	**POP**
Approver	**D. J. Campbell**	**POP**

Table of Contents

1 Abstract

The design of the ITER divertor and estimates of the required fuelling throughput have relied for many years on simulations performed by use of the SOLPS plasma edge modelling tool, more specifically its versions 4.0, 4.2 and 4.3. However, the SOLPS code base has been developed independently by other research groups among the ITER Members' institutions, most notably in IPP-Garching (EU) and St. Petersburg (RF), leading ultimately to versions SOLPS5.0/5.1 and SOLPS5.2, respectively.

While the developments at ITER on SOLPS4.x, in collaboration with FZ-Jülich (EU), have focused mainly on the physics of neutral transport, the work done in Garching has been devoted mostly to improving the physics capabilities of the plasma model and the St. Petersburg efforts have concentrated on improving the numerical solution of the drift terms and the electric potential equation. These various enhancements have now reached some maturity and it was decided at the IO in 2012 to devote resources to merging back these three code versions into a single package that would bring together all the advances from those branches of development. This began with a first Task Agreement between IO and F4E to couple the newest versions of the two main components of the SOLPS package (separate plasma fluid and neutral transport codes) to yield a new code version, named SOLPS-ITER. Subsequently, additional effort has been devoted to adding further refinements present in the various previous versions which were not all captured in the initial coupling exercise. The new code will be released to the R&D community at a workshop organized at the ITER Headquarters on April 13th-17th, 2015.

The ambition of the SOLPS-ITER effort is to make this last code version become the new standard used across the ITER Parties for modelling not only ITER, but any other tokamaks and linear plasma devices wherever applicable. In order to facilitate user adoption of SOLPS-ITER and migration from earlier versions, it has therefore been decided to include as part of the SOLPS-ITER package, a more user-friendly interface for some of the more tedious and error-prone tasks to ease the transition for users of older versions and provide additional added value and incentive for those switching to SOLPS-ITER. At the same time, SOLPS users, including those at the IO, have, over the years, expressed the desire for some run monitoring framework and more powerful graphical post-processing tools. This aim of this contract is to provide the software constituting this interface.

2 Background and Objectives

SOLPS-ITER is an important element of the Integrated Modeling (IM) strategy for ITER. It contains an input file generator (**DivGeo** [or **DG**]/**Uinp**), a grid generator suite (**CARRE**/**Tria**/**Triageom**), a solver for plasma fluid equations (**B2.5**) and for neutral kinetic transport (**Eirene**), often (but not always) run together in coupled mode in what is called **B2.5-Eirene**, and post-processing tools in the form of analysis scripts and a plotting program (**b2plot**). **B2.5-Eirene** is to become an actor in the Integrated Modelling Analysis Suite (IMAS) being developed at ITER, and the interfacing tools being envisioned in this document shall be designed with this transition in mind. As far as possible they should in fact facilitate it.

The objective of this contract is to add to the current SOLPS-ITER code suite a framework consisting of various tools aiming at improving the user's experience, to accelerate and simplify run input set-up, and to increase the scientific usability of the **B2.5-Eirene** simulation results.

3 Scope of the work

The aim of this contract is the design and implementation of a set of user-friendly interface tools, with the ultimate objective of forming a framework for orchestrating the work of SOLPS-ITER applications. These tools, described below and numbered as deliverables D1 to D4, can initially be considered as stand-alone applications, but would eventually merge into a single framework. In order of priority, the deliverables are:

D1. A 'dashboard' utility allowing a large set of runs to be scanned, identifying the state they are in, and providing a framework for in-line analysis and run re-launch, including input file editing beforehand;

D2. A graphical post-processor for plotting results selected from a collection of finished runs, so that they may be compared with each other, with output from other codes, and/or with experimental data;

D3. A graphical helper for building/editing the **B2.5** input files (*b2ag.dat*, *b2ah.dat*, *b2ai.dat*, *b2ar.dat*, *b2mn.dat*, *b2.*.parameters*, *b2.transport.inputfile*, *b2.sources.inputfile*), as well as providing a means to editing the **Eirene** *input.eir* file;

D4. A wrapper for the grid generation chain, to decrease the number of steps the users must perform to build a consistent set of input files necessary before launching a run or set of runs.

4 Work description

The workflow of a SOLPS-ITER code run can be broken down into 6 separate steps, not all of which need to be done for every simulation. The steps consist of:

1) Geometry set-up
2) Choice of physics parameters for the run(s)
3) Choice of initial state
4) Launch of the run(s)
5) In-line analysis and continuation of the run(s) until convergence
6) Post-processing analysis.

The deliverables addressed by this contract affect these steps in the following way:

Deliverable 1.	Steps 4 and 5
Deliverable 2.	Step 6
Deliverable 3.	Steps 2 and 3
Deliverable 4.	Step 1

The remainder of this Section details what occurs in each step and then provides a description of the work to be done in each deliverable to facilitate them. In general, the user interface being provided must be written in a language and using a framework that will allow for easy maintenance by the SOLPS-ITER developers at the IO and in the ITER Member parties. In particular, the methodology for the introduction of additional input switches, control checks, or quantities to be plotted to follow on future developments of the SOLPS-ITER code base must be demonstrated and documented to the IO's satisfaction.

Step 1.

Geometry set-up: this is usually defined by an experimental device, so data is provided describing the positions of surfaces bounding the domain of interest together with the properties of these surfaces such as temperature, material properties, transparency, etc… This "device description" data changes only rarely, so this is something that is saved and retrievable from a local database. At ITER, this database will be housed within IMAS and an IDS format will be used. For each device,

simulations may be required for a variety of magnetic field configurations. The latter are described in a data file containing information on the magnetic equilibrium, often called an **.eqdsk* file. From this given device description and magnetic equilibrium, a series of input files need to be built, namely:

- the main **Eirene** input file *input.eir*
- the **B2.5** grid file: *b2fgmtry*
- the **Eirene** grid files: *fort.30*, *fort.33*, *fort.34*, *fort.35*.

The first tool at the user's disposal is **DivGeo**, or **DG**, a C-based graphical tool, with which the user can load up the device description in form of a "template", the magnetic equilibrium and the topology of the magnetic domain to be considered. The user then selects and labels which surfaces are to be considered as being in contact with the plasma and neutral computational domains, and indicates their properties. The user also chooses at this stage the number of grid cells required for the discretisation of the plasma domain, region by region (core plasma, SOL plasma, PFR plasma). Once this is done, **DivGeo** creates some output that is then passed to the mesh generator **CARRE**. During the **CARRE** mesh generation process, the user has a further opportunity to modify the grid parameters. Once the grid is created, it is loaded back into the **DivGeo** model for inspection.

Assuming the grid is of sufficient quality (in particular that it does not contain concave cells), one may proceed to the triangulation step of the vacuum vessel volume lying outside the main plasma computational grid. The **DivGeo** model output file, **.dgo*, is thus translated, by means of the **Uinp** input file converter program, into input files for **B2.5-Eirene** use. The **.geo* file produced by **CARRE** is then interpreted by **b2ag** to rewrite it as a *b2fgmtry* file for the **B2.5** code and a *fort.30* file intelligible by **Eirene**. The latter is then run in a special mode, using a special input file *triang.eir* prepared by **Uinp**, to create the file *fort.78* containing the boundary of the domain to be triangulated. This domain is then triangulated by means of the **Tria** program, and the plasma computational grid is also triangulated and merged with the **Tria** output by use of the **Triageom** program. This will then produce the three files *fort.33*, *fort.34*, and *fort.35*, which contain the vertex positions, connectivities and triangle neighbour relationships, respectively.

The objective of **Deliverable 4**, is to wrap this grid generation chain so as to minimize the number of steps the user must remember. It is expected that errors will sometimes occur somewhere along the chain, but the interface provided by **Deliverable 4** is not expected to be able to parse the error message and apply a corrective action, but rather simply to bring the error message to the user, and only in the simplest cases, suggest a solution.

Step 2.

Choice of physics parameters: once the geometry has been set, the user must now choose physics parameters for the run, i.e. species to include, boundary conditions to apply (heat and particle input, mostly), transport coefficients and physics model switches to use, etc…, which are then written out in a second complement of input files (human-readable ASCII format, FORTRAN namelists and hash lists for the most part). These files are *input.eir* for **Eirene**, and *b2ah.dat*, *b2ai.dat*, *b2ar.dat*, *b2mn.dat*, *b2.*.parameters*, etc… for **B2.5**. At least for the **B2.5** input files, the syntax is straight-forward and there is ready documentation for describing the various input parameters available and their functions. The *input.eir* file is provided for the most part by **Uinp**, but the user usually still often needs to go in and modify some of its settings.

The job of the software constituting **Deliverable 3** is to provide a graphical interface for preparation of these input files to make it easier, faster and less tedious for the user. The interface could also perform some sanity and consistency checks of some of the input values, when appropriate, such checks being currently done at run-time. Sample versions of the files in question

and the list of parameters they can contain can be found in the SOLPS and Eirene manuals which will be provided to the Contractor. For ease of comprehension, the switches to set could, for example, be organized in contextual menus according to their function. The user interface shall also allow the import of a complete set of input files from an already existing run. For a few key parameters (input power, gas puff rate, separatrix midplane density, etc…), users are often interested in performing parameter scans. The tool delivered shall facilitate this task by providing a means to create a sequence of input files corresponding to the parameter scan being desired, and position each self-consistent input file set into a separate directory in preparation for launching all these runs simultaneously.

Step 3.
Choice of initial state: considering the complexity of the plasma model to be solved, it is almost always more efficient to use as the starting point to a new simulation the converged end-state of some previous run, even if it comes from a different geometry or with different physics parameters. Thus the user often renames an end state *b2fstate* file to be an initial state *b2fstati* in the new run directory being prepared. If, however, the initial state file being chosen is from a SOLPS4.x run, a converter (**b2sxdr**) must first be applied, since the two file formats are incompatible. In addition, if the *b2fstate* file chosen does not correspond to the same grid size or species list, a separate converter program (**b2yt**) is used to adapt the plasma state file to the new case at hand. Both of these converters also bring in and translate the accompanying input files that were used to compute the plasma state files, hence alleviating the need for Step 2 above in some cases, apart for possibly the adjustment of a few input parameters.

Again, the task of the software constituting **Deliverable 3** here would be to simplify and fluidify this process for the user, by allowing browsing of directories to search for an already converged end-state which could be used as the restart point. It shall also then identify from the file format and the input switches whether some conversion is necessary, and, if so, invoke the proper conversion program. Once the initial state has been imported/converted, along with its accompanying set of input files, the user shall be given the opportunity to modify this input as per Step 2 above, before run launch.

Step 4.
Launch: with all input parameters set, the user can then launch a code run or runs. Depending on the choice of physics parameters and geometry, these may take minutes (for simple toy problems) to months (for full-fledged real-size physics problems at the ITER scale). The code is typically run on a scientific computing cluster, and users will submit multiple runs simultaneously to explore physical dependencies by doing parameter scans. On these clusters, which operate on a job submission model, the jobs are timed to last for a day to a week of CPU time. The input file must therefore be mindful of this time limit, so that the SOLPS-ITER run ends in a clean state with a valid set of checkpoint files written before the CPU time allotment for the job is fully spent. Then, the job may be resubmitted if the user feels it has not yet fully converged.

The user thus needs a means to assess quickly (requiring the use of the tool to be provided as **Deliverable 1**) which jobs are currently running and which have finished. For the latter, following certain convergence criteria to be chosen by the user among some choices provided by the **Deliverable 1** software, automatic identification is required to assess:

- which runs are doing well but need continuing (and propose to the user to resubmit them);
- which runs have converged and can be stored for archival/analysis/post-processing;
- which runs have crashed;

- ambiguous runs which have not crashed, but do not show that they are on the path to convergence according to the automatic criteria and need some human assessment.

Of course, all of this shall be performed within a single occurrence of the software tool of **Deliverable 1**, not one instance per run to be checked. The checks need not be performed in a continuous fashion, but rather only updated at the user's request and on start-up. Because of the machine-specific nature of the job submission script, the user would be expected to provide a skeleton script in some standard location that would accomplish an orderly job submission on the local computer system. However, the bulk of the work done by the **Deliverable 1** software must not rely on any specifics on the cluster on which the jobs are being run. The software shall interrogate only the SOLPS-ITER log files (*run.log, b2ftrace, b2ftrack, b2time.nc,* …) being written during the runs, and keep track, by means of an internal log file, of which runs have been launched, in which directory they are running, and the result of their last assessment. A mechanism for inspection and clean-up of this internal log file shall also be provided.

Step 5.
In-line analysis and continuation: for runs currently executing, some simple in-line analysis to look at a few important physics parameters and assess if the run is doing well and going in the right direction is required. A rich library of individual command-line scripts already exists to perform such an analysis function.

The tool developed as **Deliverable 1**, once it has analysed the various runs under consideration as per Step 4 above, must therefore give the opportunity to the user to run these command-line scripts from within, with some graphical dialog boxes and contextual menu choices. The results could then be plotted directly inside one of the program's windows.

Step 6.
Post-processing: once runs have been identified as having converged and completed, post-processing and a means to compare results from different runs with each other is required. In the present SOLPS versions, only *ad hoc* solutions exist, often using proprietary software. For single-run analysis, the existing **b2plot** program can be employed, used to obtain ASCII and graphical output from a single run. It is written in Fortran and uses the NCAR and GKS libraries.

The goal of **Deliverable 2** is not to duplicate **b2plot**, but rather to provide a new complementary tool allowing the same quantity from a series of runs to be plotted, facilitating comparisons, and to enable the plotting of one quantity in the code output data against another. More detailed instructions regarding the functionality of this plotting algorithm will be provided by the IO-TRO at contract initiation. It is also required that the plots and the data contained within them be exported into some standardized format for comparison with output from other codes and/or experimental data. Details of this export format are to be discussed between the C-RO and IO-TRO.

5 Responsibilities

The Contractor appoints a responsible person (C-RO) who shall represent the Contractor in all matters related to the implementation of this Contract.

The contractor will be responsible for the work described in Section 4, providing results according to the scope of the work outlined above and fulfilling the implementation plan and conditions of the present contract.

The IO will provide access to the SOLPS-ITER GIT server to the contractor and create dedicated branches where the contractor can execute all necessary code development.

The contractor will be responsible for delivering the user interface code with compilation instructions and accompanying documentation, as needed, to the dedicated branches of the SOLPS-ITER GIT server.

The IO-TRO will then be responsible for additional testing of the SOLPS-ITER user interface work packages and incorporating them, once they meet the acceptance criteria, into the SOLPS-ITER master branch of the repository for general release and distribution. If the needs to be met by the user interfaces evolve during the effective duration of the contract, the IO-TRO will be responsible for drafting the required changes to the relevant interface's features.

6 Deliverables and Time Schedule

The 4 deliverables for this contract are as listed above in Section 3 and again below. Their ordering is to be understood as a priority ranking for meeting the current needs of SOLPS-ITER development at the IO and elsewhere, and they should be achieved and delivered sequentially.

Deliverable 1. 'Dashboard' utility, with the following features:

i. Keeps track of the jobs submitted via its interface
ii. Allows for the selection of convergence criteria
iii. Scans the log files from those jobs and flags the runs as:
    1. On-going
    2. Finished, not yet converged, but doing well, according to the criteria from ii
    3. Finished and converged, according to the criteria from ii
    4. Finished, not yet converged, and not doing well, according to the criteria from ii
    5. Finished, but crashed
iv. Provides a framework for in-line analysis of the jobs, both on-going and finished
v. Allows for continuation of a run by submitting of a new job, with possible simple input file editing beforehand (more complex edition task to be provided later via **Deliverable 3**).
vi. Allows for archival of a job

Initial delivery: no later than 4 months after contract start-up date
Final delivery: no later than 6 months after contract start-up date

Deliverable 2. Graphical post-processor, with the following features:

i. Ability to represent quantities for multiple **B2.5-Eirene** runs on the same plot
ii. Ability to plot parameter scans, showing the dependency of code results on run parameters
iii. Ability to produce publication-quality graphics
iv. Ability to output data from produced plots in a standardized ASCII format
v. Ability to import and plot experimental data and results from other codes in the same standard format

Initial delivery: no later than 8 months after contract start-up date
Final delivery: no later than 10 months after contract start-up date

Deliverable 3. 'Point-and-click' utility for input file build-up, with the following features:

i. Ability to create/build/edit the **B2.5** input files (*b2ag.dat*, *b2ah.dat*, *b2ai.dat*, *b2ar.dat*, *b2mn.dat*, *b2.*.parameters*, *b2.transport.inputfile*, *b2.sources.inputfile*)
ii. Ability to import the input files from an existing run

1. Identify the origin of the older run and run appropriate converter if necessary
2. Propose adaptation of older run input to adjust grid size and/or species list

iii. Organization of the input switches according to role in code
iv. Description of the input variables directly available as per the existing SOLPS-ITER documentation files
v. Performing sanity and consistency checks of the input parameters
vi. Ability to edit the **Eirene** *input.eir* file

Initial delivery: no later than 12 months after contract start-up date
Final delivery: no later than 14 months after contract start-up date

Deliverable 4. Wrapper to the grid generation chain, with the following features:

i. Overarching framework able to call the various tools of the grid generation chain (**DivGeo**/**Carre**/**Uinp**/**Tria**/**Triageom**).
ii. Automate the intermediate steps for transitioning between the various tools
iii. Create a corresponding self-consistent set of input files
iv. Link back to **Deliverable 3** for further editing of these input files

Initial delivery: no later than 16 months after contract start-up date
Final delivery: no later than 18 months after contract start-up date

The work under this contract will need to be performed with frequent interactions between IO and the user interface developers. These interactions will take the form of either videoconference calls with screen sharing capability to see the prototype code in action, or by in person meetings on site at the IO. The break-up of the deliverables into 4 work packages, each providing a particular capability, is meant to allow the IO to obtain a valuable product it can use and meets the needs of the SOLPS user community as the contracted work progresses. It also provides for earlier release of the interface tools to the wider SOLPS-ITER user community as they are being implemented. It is important to note that these various interface tools are destined in the end to be part of the same unified framework, so shall be built with cross-compatibility in mind. It is essential that they also avoid making use of proprietary software for which restrictive licenses and/or financial investment are necessary, since this would then impair the portability and dissemination of SOLPS-ITER to the wider plasma physics R&D community.

Starting date: Signing of contract.
Completion date: at the latest 18 months from the date of signature.

7 Acceptance Criteria

The Contractor shall demonstrate that personnel assigned to the contract have the experience and technical capability to perform the work described in Section 4.

The Contractor shall quote a fixed amount for each of the work packages described in Section 4.

The initial delivery of each of the deliverable work packages will consist of:

- commitment of the corresponding piece of code and accompanying compilation instructions to a clearly identified branch on the SOLPS-ITER GIT repository maintained by the IO, as well as appropriate documentation, in either PDF or LaTeX format, the latter being preferred so that it may easily be incorporated into the body of the general SOLPS user manual;

- successful compilation of the code on the ITER HPC cluster within the SOLPS-ITER run environment;
- demonstration of the software with the execution of a pre-defined sample of use cases.

The accompanying documentation to the software will present the interface tool being delivered, describe its functionalities, and demonstrate its proper use with illustrative examples. Each work package being delivered should be self-enclosed, readily usable and demonstrate integration into the overall SOLPS-ITER framework scheme.

During a two working week period following receipt of each Deliverable, the IO-TRO will examine the delivered software tool, and test it to check the following:

- Compilation and execution of the SOLPS-ITER user interface on the ITER HPC cluster, within its Linux environment, and assess potential portability issues to other systems where SOLPS-ITER will be in use,
- Comprehensiveness of the tool,
- Usability of the features being delivered with representative use cases,
- Compatibility with existing other SOLPS-ITER components, including previous deliverables from this contract, when applicable,
- Accuracy, comprehensiveness, and readability of the accompanying documentation,
- Ease of future code maintenance.

After the initial delivery of each work package, if the software succeeds in passing these tests, it will be incorporated into the master branch of the SOLPS-ITER GIT server. If not, the contractor will need to address any identified issues and propose a remedy to them at the next scheduled interim progress meeting. Such assessments will take place as part of the proposed fortnightly progress meetings through the contract duration. The contractor will then have up to an additional six (6) weeks to deliver a new updated version of the software with the identified problems fixed. A delivery will be considered final and accepted only once the corresponding code and accompanying documentation have been committed to the SOLPS-ITER GIT server master branch and tagged as a new code version ready for release and dissemination to the SOLPS-ITER user community.

The contract will be considered complete once ITER has accepted the final deliverable.

8 Work Monitoring

At the beginning the contract, the Contractor shall send one or more representatives to the ITER Headquarters in Cadarache, for a period of at least two weeks, to become familiar with the current working environment used for SOLPS-ITER and learn how to use the various code suite components. This will also be the opportunity for the Contractor to have extensive discussions with the IO-TRO to ensure that the Contractor has a clear understanding of the long-term vision held at the IO for the SOLPS-ITER code suite and the functions to be performed within it by the various Deliverables.

Meetings between the C-RO and IO-TRO should occur, where possible, every two weeks following contract kick-off. More frequent meetings can be called by either party as the need arises. These meetings may take the form of either videoconference calls with screen sharing capability to see the prototype code in action, or in person meetings on site at the IO. It is to be expected that an in-person visit by the Contractor team or representative will be required at contract kick-off.

Monitoring shall proceed through regular progress meetings and via more informal discussions which can be conducted at any time between the IO and the Contractor as required. At least the following meetings should be foreseen:

Scope of meeting	Point of check/Deliverable	Place of meeting
Kick-off contract	Work programme	ITER site
Interim progress meetings	Expected approximately every 2 working weeks to be agreed between the IO-TRO and the C-RO	Videoconference or ITER site
Individual work package initial delivery	Commitment of work package to GIT server and demonstration of work package use	Videoconference or ITER site
Individual work package final delivery	Approx. 2 months after the initial work package delivery or 6 weeks after IO-TRO request for modifications Merging of work package into master SOLPS-ITER GIT branch	Videoconference or ITER site
Closing contract meeting Contract completion	Full documentation report of the set of all work packages and final code commit to the SOLPS-ITER GIT server master branch	Videoconference or ITER site

9 Payment schedule

Payment made following invoice submitted by the Contractor with final delivery of each work package.

10 Quality Assurance (QA) requirement, specific requirements and conditions

Prior to commencement of any work, a Quality Plan must be provided to IO for approval. This is a separate document which comprises:

1) a workplan with proposed time schedule and agreed preliminary dates for progress meetings,

2) a statement of those involved in the activity and their approximate role and contribution in time,

3) a statement of what work (if any) will be subcontracted and who will responsible for checking this.

It is noted that Contractor's personnel visiting the ITER project will be bound by the rules and regulations governing safety and security. The Contractor shall have and maintain the necessary equipment and licences to run the software tools required to carry out the engineering analyses and produce the deliverables in accordance with the tools adopted by the IO.

The official language of the ITER project is English. Therefore all input and output documentation relevant to this Contract shall be in English. The Contractor shall ensure that all the professionals in charge of the Contract have an adequate knowledge of English, to allow easy communication and adequate drafting of technical documentation. This requirement also applies to the Contractor's staff working at the ITER site or participating in meetings with the IO.

Documentation developed shall be retained by the Contractor for a minimum of 5 years and then may be discarded at the direction of the IO. The use of computer software to perform a safety basis task activity such as analysis and/or modelling, etc shall be reviewed and approved by the IO prior to its use, it shall fulfil IO document on calculation code for safety analysis. The work may require the presence of the Contractor's personnel at the IO site, Route de Vinon sur Verdon, 13115 St Paul Lez Durance, France, for the purpose of meetings and data gathering.

11 References/terminology and acronyms

Relevant reference documents :
SOLPS-ITER Documentation: https://user.iter.org/?uid=Q92BAQ
SOLPS-ITER GIT repository: https://git.iter.org/projects/BND/repos/solps-iter/
SOLPS-ITER flowchart: https://user.iter.org/?uid=QHYQZT
SOLPS-ITER manual: https://user.iter.org/?uid=Q992NG
Eirene documentation: http://www.eirene.de/
The documents above will be made available to prospective bidders at their request if needed for preparation of their offer.

List of abbreviations and definitions used in this document:
EU: European Union
F4E: Fusion for Energy
GGD: Generalized Grid Description
GUI: Graphical User Interface
HPC: High Performance Computing
IDS: Interface Data Structure
IO: ITER Organization
IM: Integrated Modeling
IMAS: Integrated Modeling Analysis Suite
PFR: Private Flux Region (the region of plasma that is located on the other side of the magnetic null point from the confined core plasma)
RF: Russian Federation
RO: Responsible Officer
SOL: Scrape-Off Layer (the region of plasma that is magnetically connected to the walls and next to the confined core plasma)
SOLPS: Scrape-Off Layer Plasma Simulator

Notation: Program executables from within the SOLPS-ITER suite are designated in **bold**, while the filenames used are *italicized*. The SOLPS workflow chart and manuals should be referred to for further details.

Programs included within the SOLPS-ITER suite:

DivGeo/DG: Graphically based pre-processor for building input files for later SOLPS-ITER stages (in C).

CARRE: Grid generator program to create the magnetically aligned quadrangular grid used for computing the plasma solution (in Fortran)

Uinp: Input file generator that takes the output from **DivGeo** and extracts from it the information necessary to build the **Eirene** input file (in Fortran)

TRIA: Triangulation program for producing the unstructured triangular grid in the vacuum regions in the far corners of the vacuum vessel where no plasma is expected but neutral pressure may be significant (in Fortran)

TRIAGEOM: Program that merges the grids produced by **CARRE** and **TRIA** into a single triangular semi-structured grid (in Fortran)

B2.5: Program that solves the plasma fluid equations for the charged species (in Fortran)

Eirene: Program that solves for the neutral transport on an arbitrary plasma background and in an arbitrary geometry (in Fortran)

B2.5-Eirene: Name given to the **B2.5** and **Eirene** codes when run together in coupled mode

b2plot: Post-processor used to obtain graphical representations of the results of a single **B2.5-Eirene** run (in Fortran, based on the NCAR library)

Main SOLPS-ITER filenames referred to by or relevant to this document:

Files noted as "human-readable" are available for user modification. Others are not meant to be manipulated internally.

b2ag.dat: input file to the **b2ag B2.5** pre-processor that builds the geometry file *b2fgmtry* in the **B2.5** format from the **.geo* **CARRE** grid file (ASCII format, human-readable)

b2ah.dat: input file to the **b2ah B2.5** pre-processor that builds the parameters file *b2fpardf* containing the default set of transport coefficients and boundary conditions to be used by the plasma solver (ASCII format, human-readable)

b2ai.dat: input file to the **b2ai B2.5** pre-processor that builds the initial state file *b2fstati* containing the initial plasma state from which to start the **B2.5** or **B2-Eirene** run (ASCII format, human-readable)

b2ar.dat: input file to the **b2ar B2.5** pre-processor that builds the atomic physics rates file *b2frates* containing the look-up tables to be used by **B2.5** (ASCII format, human-readable)

b2fgmtry: **B2.5** geometry file (ASCII format)

b2fpardf: **B2.5** parameters file (ASCII format)

b2frates: **B2.5** atomic rates file (ASCII format)

b2fstati: **B2.5** initial state file (ASCII format)

b2mn.dat: input file to the **b2mn** main **B2.5** program containing the run switches and eventually overriding parameters to those specified in *b2ah.dat* (ASCII format, human-readable)

b2..parameters*, *b2.transport.inputfile*, *b2.sources.inputfile*: input files containing the namelists used by **b2mn**, which complete the input provided by the *b2??.dat* files (ASCII format, human-readable)

**.eqdsk:* file containing the information about the magnetic field configuration (ASCII format)

.geo*: geometry file format obtained after the translation step from the **carre script (ASCII format)

input.dat, input.eir: **Eirene** input file names, which contain the description of the bounding surfaces, a list of species to be followed and their reactions, and particle source definitions (ASCII format, human-readable)

fort.30: file containing the details of the **B2.5** computational grid being passed to **Eirene** (ASCII format)

fort.31: file containing the plasma background computed by **B2.5** upon which **Eirene** will follow neutral particle trajectories (ASCII format)

fort.33, fort.34, fort.35: files containing a description of the triangular unstructured grid used for the **Eirene** computation (ASCII format)

fort.44: file containing the particle, momentum and energy sources deduced from the **Eirene** trajectories to be passed back to **B2.5** (ASCII format)

*plasmastate.**: **B2.5-Eirene** checkpoint files that allow for restart (identical format to *b2fstati*)

run.log: standard name given to the **B2.5(-Eirene)** output log file produced during a run (ASCII format)

Denomination	**Definition**	**Acronym**
ITER Organization	For this Contract the ITER Organization	IO-
ITER Organization Technical Responsible Officer	Person appointed by the ITER Organization with responsibility to manage all the technical aspects of this contract	IO-TRO
Contractor	Firm or group of firms organized in a legal entity to provide the scope of supply.	C-
Contractor Responsible	The person appointed (in writing) by the legally authorised representative of the Contractor, empowered to act on behalf of the Contractor for all technical, administrative legal and financial matters relative to the performance of this contract	C-R
Contractor Responsible Officer	Equivalent to the IO-TRO in the Contractors team.	C-RO