Development Plan: Formats for Data Transfer and Graphics for Simulation Codes

Igor Grešovnik, Božidar Šarler

June 2011

1 PLAN

1.1 Basic plan

We will implement a set of C# libraries and applications for saving/reading simulation input and output data and for graphical representation of simulation results.

Data formats should be extensible, such that additional things can be added when necessary. After the basic framework is set up, each interested developer can add things that are needed for specific simulation code.

Project is long term. It is lead by IG and implemented by TK, KM, QL, GKk and UH. Responsible for allocation of resources: BŠ. RV and UH will have the main advisory role with respect to arrangement of data in final formats.

1.2 Things to Be Done Shortly

Get the necessary office space - BŠ, help of others if necessary

Define plan of work – BŠ, IG

Prepare working environment – IG, KM, TK, RV

- KM & TK check for a task management system, implement different systems, and perform comparison (remark: BŠ has decided that we don’t go into this for now; the task may be approved later)
- IG performs the rest of SVN administration, supplements rules for use of servers, writes programmer’s guidelines, etc.

Carry out the last order of hardware (workstation for UNG) – IG, BŠ

See what is missing w.r. books, software, etc. – IG, others

Set up project structure for simulation tools in C# - IG

Learn or improve knowledge of C# - QL, GKk, KM, TK

1.3 Suggested Course of Work (will adapt to situation)

We arrange working conditions (offices, common services - task management & SVN)

We organize information meetings (BŠ) where people are informed about the plan and their role & tasks.

Organize some working meetings in order to establish a feasible mid term plan (and correct long term plan, if necessary). UH should express his opinion about this, propose a feasible timeline, especially with regard to interaction with his code.

With UH, BŠ and probably RV, we discuss feasibility of the plan and eventual corrections if needed.

Suggest graphic libraries to be looked at (MS WPF, Activiz.NET (VTK), Dislin?)

QL, KM, GK improve knowledge on C#.

IG & TK start to work on definition and implementation of I/O. Reference implementation in C# based on tools from Investigative Generic Library (similar approach is used in the neural networks module developed by IG & TK) will simultaneously act as format definition (de facto standard) that is evolving. Stages:

- Definition of basic framework
- Implementation of testing routines for implementation of this framework (KM, QL, UH and others begin observing the work)
- Collection of requests (from UH, KM, etc.) for addition of variables & fields
- QL & GKk (supervision of UH) begin implementation of UH's presentation library by using I/O representation of data
- Supplementation with missing graph types (QL, GK, KM, UH, TK?)
- Switch to or addition of other graphic libraries, if necessary (probably a single library can not cover everything, maybe the quality of some output is not good enough, or there are other limitations, etc.). This may involve redesign of graphical modules.

UH works on his representation functionality. His first priority is to prepare Ph.D. thesis. He puts his code on SVN and connects it with libraries.

Division of I/O and presentation code to multiple projects will be made, some migrate to library part.

QL (and others?) with help of UH adapt current graphic capabiities to standard I/O object formats.

2 TIME LOG

01 2011

Big plan defined in January-February 2011

02 – 06 2011

A number of changes announced since then, statements with respect to plans were changing, things are still not defined enough to start work.

06-06-2011

Meeting (IG & BŠ, discussing implementation, BŠ writes summary of variables contained in input and output files and the necessary graphs).

3 VARIABLES FOR I/O FILES, GRAPHS

3.1 Remarks on I/O formats

Names should be easily read, distinctive and descriptive as much as possible (e.g. Displacements, Deformation, Temperature, RollDistanceCenterCenter, etc.). Long compound names are allowed, with capitalization of words, exceptionally with underscores ('_' – to be considered), rules for variables in C# and C++ apply.

JSON format will be used. Format is standardized (many tools are available, which means easier validation; strictly defined syntax; human readable; data to object mapping generically implemented in IGLib; extensible).

Structure will be optimized with respect to I/O, not with respect to use (this would not be possible anyway, since we will deal with different codes). Object representation follows a DTO (data transfer object) paradigm: we will have standard I/O object structure, library of routines (graphics, I/O), and internal transcription algorithms within each code that uses the system.

Grouping of vector & tensor field data is by field and then by component (i.e. a single component is listed as a raw table for all points, rather than all components grouped together for a single point). In the cases where something is not defined on all points, there will be point indices or point flags. This idea has to be elaborated, discuss with involved parties!

For UH's code (slice model) data will be arranged in slices, which are treated normally as 2D domains. If there should be additional components of something in the direction normal to a slice (e.g. heat flux), this should probably be solved by inheritance, by a concept that a 2D slice is a more specific 2D domain.

List from BŠ needs to be checked by the involved parties and supplemented. Feasibility of implementation steps needs to be discussed.

3.1.1 Suggested nesting

From outer-most to inner-most:

- Application
- SimulationGroup (e.g. Thermal, Mechanical, Micro)
- TimeStep
- Domain (part of geometry with same physics, belonging to the same body)
- Conditions / Variables (e.g. initial coordinates vs. Coordinates & displacements)
- Point defined / function defined (e.g. temperature in mesh points vs. initial temperature function)
- Field (e.g. Displacement, Temperature, Stress)

- Field component (e.g. [i,j] for tenzors, [i] for vectors, no nesting for scalars). This is simply a table of values in mesh points.

Each field will have, beside its subfields (or final values in the case of field components), a data section with fields such as Title, Description, ComponentNames, Ponts (optional - defining in which points values aree defined when not defined in all points, etc.

Maybe each field will have a pointer to its parent, which will not be written to files. This is to be considered because it may imply two level DTO transcription, which is bed, or one option is that methods are provided to set or clear this field and it is automatically cleared through object tree before write operations, and set after read operations.

3.1.2 Rough list of general variables (fields) to be included

Mechanical:

- *InitialCoordinate* – vector (should we use different name, or it would be just Coordinate stated in different block (belonging to initial conditions)?)
- *Coordinate* – vector
- *SurfaceNormal* – vector
- *Displacement* – vector
- *Deformation* – tensor
- *EffectiveDeformation* – scalar (dos it need to be specified separately, or should it be calculated in the handling code?)
- *Stress* – tensor
- *EffectiveStress* – scalar (does it need to be specified separately?)
- *Traction* – vector
- ForceDensity - *vector*

Thermal:

- Temperature - scalar
- HeatFlux - vector
- HeatSource - scalar

3.1.3 Supplementation of fields:

3.2 Remarks on Graphs

We have to look at which graphs are already implemented by UH, and which of the remaining can be implemented soon (approximate timeline!).

Then we need to see if it is feasible to adapt tools that UH has to common formats. Somebody (e.g. QL & GK) can implement these graphs such that standard structures for I/O are used.

Further, graphs that will be missing, are implemented by the same persons (and maybe help of others).

3.3 List of Variables & Graphs (updated from Umut)

1 PROCESS PARAMETERS

1.1 GENERAL PROCESS (SIMULATION CONTROL) DATA

V_ENTRY - initial entry speed towards the rolling direction in [mm/s]
T_ENTRY(N_MAX) – initial temperature values of the nodes on the first slice in [K].
Z_START - initial calculating coordinate
Z_END - final calculating coordinate
N_SLICE_MAX - maximum number of slices
Z_SLICE(N_SLICE) – coordinate z of slice N_SLICE
N_MAX(N_SLICE) - number of discretisation points in slice N_SLICE
NB_MAX(N_SLICE) - number of discretisation points on the boundary
ND_MAX(N_SLICE) - number of discretisation points in domain
P_X(N_MAX) - x coordinates of slice
P_Y(N_MAX) - y coordinates of slice

1.2 ROLLS (PRIMARY DIE) DEFINITION VARIABLES

N_ROL_O_MAX - number of horizontal rolls below the peace
N_ROL_I_MAX - number of horizontal rolls above the peace
N_ROL_L_MAX - number of vertical rolls left to the peace
N_ROL_R_MAX - number of vertical rolls right to the peace

R_ROL_O(N_ROL) - radius of roll
R_ROL_I(N_ROL) - radius of roll
R_ROL_L(N_ROL) - radius of roll
R_ROL_R(N_ROL) – radius of roll

Z_ROL_O(N_ROL) – position of Z coordinate of the roll O
Z_ROL_I(N_ROL) – position of Z coordinate of the roll I

Z_ROL_L(N_ROL) – position of Z coordinate of the roll L
Z_ROL_R(N_ROL) – position of Z coordinate of the roll R

Y_ROL_O(N_ROL) - distance of roll center from the center in y dir.
Y_ROL_I(N_ROL) - distance of roll center from the center in y dir.
X_ROL_L(N_ROL) - distance of roll center from the center in x dir.
X_ROL_R(N_ROL) - distance of roll center from the center in x dir.

N_ROL_GRV_O_MAX(N_ROL) – number of points for roll groove definition
N_ROL_GRV_I_MAX(N_ROL) – number of points for roll groove definition
N_ROL_GRV_L_MAX(N_ROL) – number of points for roll groove definition
N_ROL_GRV_R_MAX(N_ROL) – number of points for roll groove definition

X_ROL_GVR_O(N) – x cordinate of roll from left to right for definition of roll groove
Y_ROL_GVR_O(N)- roll groove
X_ROL_GVR_I(N) -
Y_ROL_GVR_I(N)
X_ROL_GVR_L(N)
Y_ROL_GVR_L(N)
X_ROL_GVR_R(N)
Y_ROL_GVR_R(N)

1.3 MECHANICAL SLICE (WORKPIECE) VARIABLES

DIS_X(N_MAX) - displacement in direction X in point N
DIS_Y(N_MAX) - displacement in direction Y in point N
DIS(N_MAX) – displacement in point N
DIS_X_DOT (N_MAX) - strain rate in direction x in point N
DIS_Y_DOT (N_MAX) - strain rate in direction x in point N
DIS_DOT (N_MAX) - strain rate in point N
SIG_XY(N_MAX) - stress in point N
SIG_XX(N_MAX) - stress tensor in point N
SIG_YY(N_MAX) - stress tensor in point N

1.4 THERMAL SLICE VARIABLES

TEM(N_NAM) - temperature of point [C]
HFL(NB_MAX) – heat flux at the boundary [W/m**2]
HSC(N_MAX) – heat source of point [W/m**3]

1.4.1 Thermal Definition Functions

FT0(N_MAX) – initial temperature generation function
FHFL(NB_MAX) – heat flux forcing function
FHH(NB_MAX) – heat transfer coefficient forcing function

FTR(NB_MAX) - heat transfer coefficient forcing function

2 MATERIAL PROPERTIES

MAT(N) - referred index of a specific material used.
E(MAT(N)) - Young's modulus of a material N.
v(MAT(N)) - Poisson's ratio of a material N.
Hp(MAT(N), N_MAX, N_SLICE) – Instant (tangental) plastic modulus of a material
GEO(N) – Geometrical assumption index, plane stress or plane strain.

3 PLOTS

OUTPUT CONTOUR PLOTS FOR EACH SLICE

Temperature field

OUTPUT PLOTS FOR BOUNDARY VARIABLES
Temperature
Heat flux
Heat transfer coefficient
Reference temperature
Displacement in X - direction
Displacement in Y – direction
Strain rate in X - direction
Strain rate in Y – direction
Absolute value of displacement

CROSS-SECTIONAL PLOTS
X direction at Y=0: temperature
Y direction at X=0: temperature
X direction at Y=0: displacement in X direction
Y direction at X=0: displacement in X direction
X direction at Y=0: displacement in Y direction
Y direction at X=0: displacement in Y direction

LONGITUDINAL ROLLING MILL PLOTS
Positions of the rolls with roll radius I-O direction
Positions of the rolls with roll radius L-R direction
Number of total discretisation points of the slice [N_MAX]
Number of total boundary points of the slice [NB_MAX]
Number of total domain points of the slice [NB_MAX]
Remeshing indicator [1: remeshing done, 0: remeshing not done]
Average slice temperature [C]
Total slice heat generation [W]
Total slice heat removal [W]

Average slice crossection
Average slice velocity in z-direction

ROLL PLOTS
Roll groove as a function of the roll wide in slice coordinate system (vertical and horisontal grooves on one plot)

3.4 *List of Variables & Graphs (original from Božidar Šarler)*

```
                    ROLLING SIMULATOR VARIABLES

Z_START - initial calculating coordinate
Z_END - final calculating coordinate
N_SLICE_MAX - maximum number of slices
N_MAX(N_SLICE) - number of discretisation points in slice N_SLICE
NB_MAX(N_SLICE) - number of discretisation points on the boundary
ND_MAX(N_SLICE) - number of discretisation points in domain
P_X(N_MAX) - x coordinates of slice
P_Y(N_MAX) - y coordinates of slice

ROLLS DEFINITION VARIABLES

N_ROL_O_MAX - number of horisontal rolls below the peace
N_ROL_I_MAX - number of horisontal rolls above the peace
N_ROL_L_MAX - number of vertical rolls left to the peace
N_ROL_R_MAX - number of vertical rolls right to the peace

R_O(N_ROL) - radius of roll
R_I(N_ROL) - radius of roll
R_L(N_ROL) - radius of roll
R_R(N_ROL) – radius of roll

Y_R_O(N_ROL) - distance of roll center from the center in y dir.
Y_R_I(N_ROL) - distance of roll center from the center in y dir.
X_R_L(N_ROL) - distance of roll center from the center in x dir.
X_R_R(N_ROL) - distance of roll center from the center in x dir.

N_R_O_MAX(N_ROL) – number of points for roll groove definition
N_R_I_MAX(N_ROL) – number of points for roll groove definition
N_R_L_MAX(N_ROL) – number of points for roll groove definition
N_R_R_MAX(N_ROL) – number of points for roll groove definition
```

X_GVR_ROL(N) – x cordinate of roll from left to right or from bottom to up (center of the roll X_ROL = 0.0) for definition of roll groove
Y_GRV_ROL(N) – y coordinate of roll groove (values only)

MECHANICAL SLICE VARIABLES

DIS_X(N_MAX) - displacement indirection X in point N
DIS_Y(N_MAX) - displacement indirection Y in point N
DIS(N_MAX) – displacement in point N
DIS_X_DOT (N_MAX) - displacement rate in direction x in point N
DIS_Y_DOT (N_MAX) - displacement rate in direction x in point N
DIS_DOT (N_MAX) - displacement rate in point N
SIG_XY(N_MAX) - stress in point N
SIG_XX(N_MAX) - stress tensor in point N
SIG_YY(N_MAX) - stress tensor in point N

THERMAL SLICE VARIABLES

TEM(N_NAM) - temperature of point [C]
HFL(NB_MAX) – heat flux at the boundary [W/m**2]
HSC(N_MAX) – heat source of point [W/m**3]

THERMAL DEFINITION FUNCTIONS
FT0(N_MAX) – initial temperature generation function
FHFL(NB_MAX) – heat flux forcing function
FHH(NB_MAX) – heat transfer coefficient forcing function
FTR(NB_MAX) - heat transfer coefficient forcing function

PLOTS

OUTPUT CONTOUR PLOTS FOR EACH SLICE

Temperature field

OUTPUT PLOTS FOR BOUNDARY VARIABLES
Temperature
Heat flux
Heat transfer coefficient
Reference temperature
Displacement in X - direction
Displacement in Y – direction
Absolute value of displacement

CROSS-SECTIONAL PLOTS
X direction at Y=0: temperature
Y direction at X=0: temperature

```
X direction at Y=0: displacement in X direction
Y direction at X=0: displacement in X direction
X direction at Y=0: displacement in Y direction
Y direction at X=0: displacement in Y direction

LONGITUDINAL ROLLING MILL PLOTS
Positions of the rolls with roll radius I-O direction
Positions of the rolls with roll radius L-R direction
Number of total discretisation points of the slice [N_MAX]
Number of total boundary points of the slice [NB_MAX]
Number of total domain points of the slice [NB_MAX]
Remeshing indicator [1: remeshing done, 0: remeshing not done]
Average slice temperature [C]
Total slice heat generation [W]
Total slice heat removal [W]
Average slice crossection
Average slice velocity in z-direction

ROLL PLOTS
Roll groove as a function of the roll wide in slice coordinate
system (vertical and horisontal grooves on one plot)
```

References:

[1] Igor Grešovnik: Coordination of software development in COBIK and Laboratory for Multiphase Processes. Treatiese, COBIK, 2011.

[2] Igor Grešovnik: *Programmers' guidelines for Development of Software within COBIK & Laboratory for Multiphase Processes*. Treatise, COBIK, 2012.

[3] Igor Grešovnik: *IGLib.NET*, electronic document at http://www2.arnes.si/~ljc3m2/igor/iglib/index.html.

[4] Igor Grešovnik: *IGLib Documentation*, electronic document at http://dl.dropbox.com/u/12702901/code_documentation/generated/iglib/index.html .

[5] Igor Grešovnik: *IGLib.NET Code Documentation*, electronic document at http://dl.dropbox.com/u/12702901/code_documentation/generated/iglib/html/index.html .