Why would a modeller take the trouble to comply with a shared modelling framework?

J. De Keyser, E. Gamby, M. Kruglanski

Belgian Institute for Space Aeronomy

S. Poedts, G. Lapenta

Katholieke Universiteit Leuven

A. Lani, H. Deconinck

Von Karman Institute for Fluid Dynamics

D. Heynderickx

DH Consulting

Johan.DeKeyser@aeronomie.be

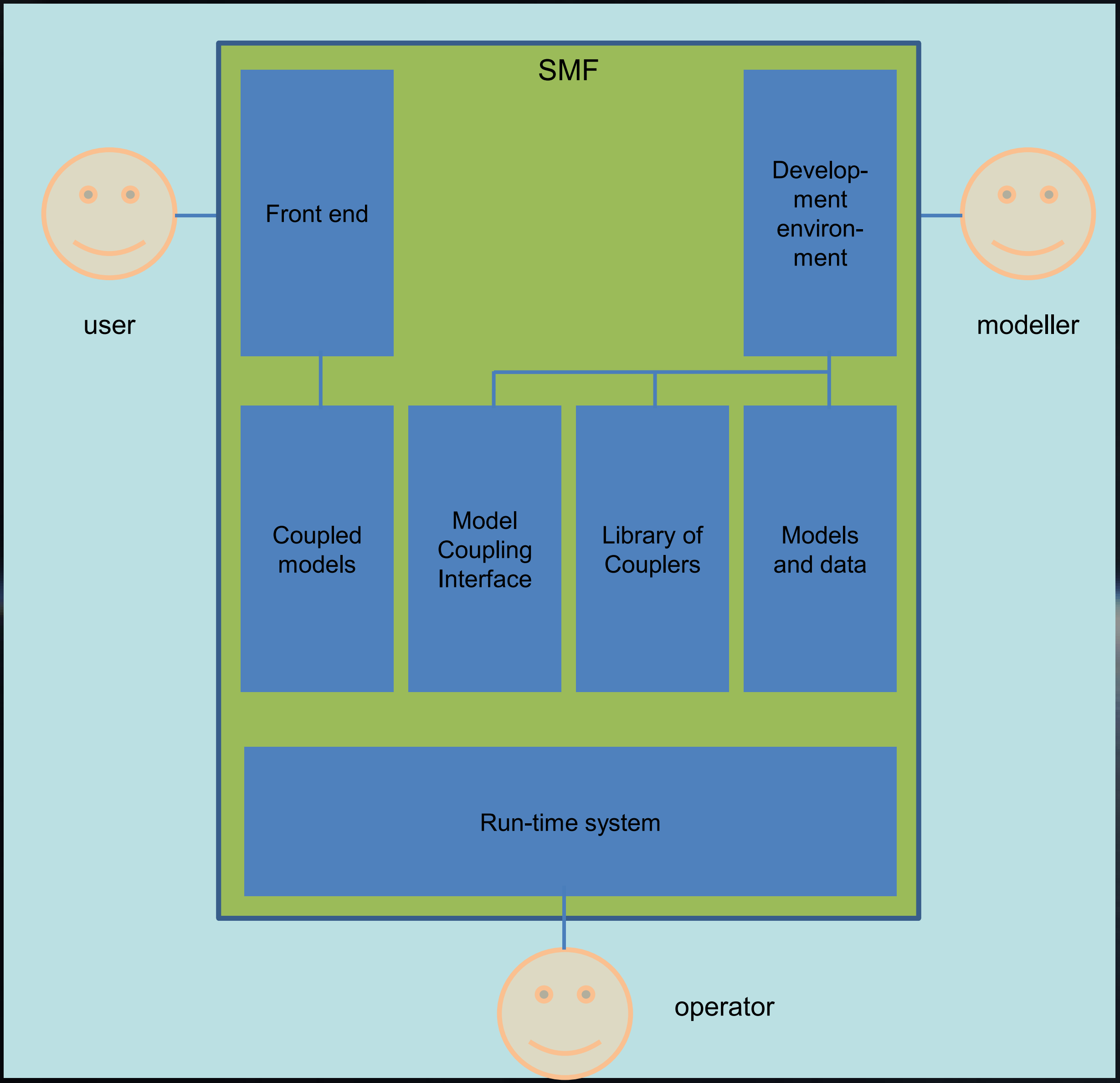
Introduction

- The goal of this presentation is twofold
  - To incite modellers to do some work to use a shared modelling framework (SMF).
  - To incite framework developers to make life easy for modellers
    - provide support for porting codes to the framework
    - foresee added-value tools
- Part of these considerations stem from the VSWMC Phase 1A study

SMF components

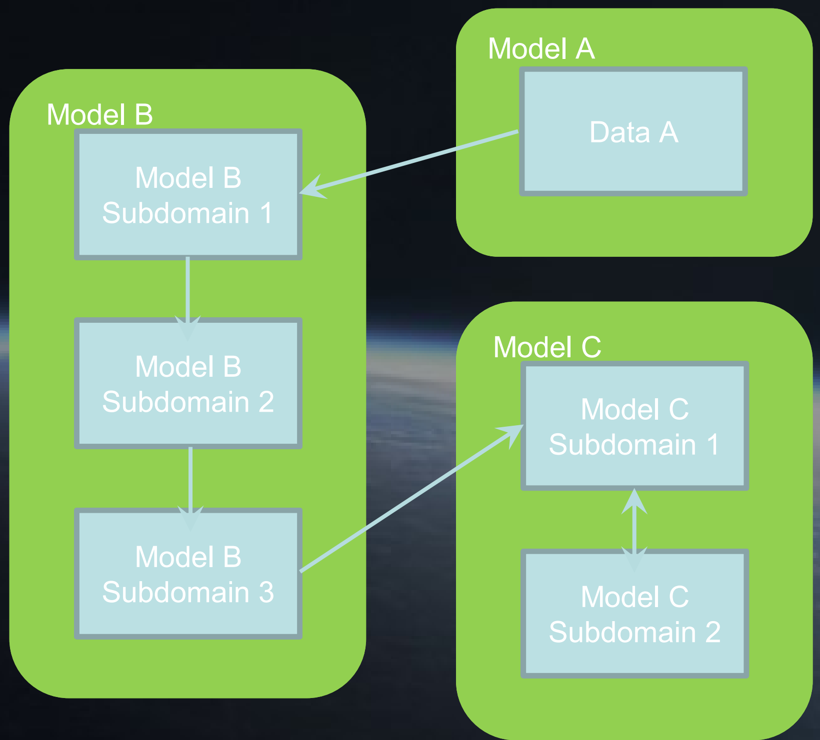
- A SMF is a software environment that allows different data sources and models to be coupled with each other and executed.

- A SMF may consist of
  - Data sources and Models
  - RTS: Run-Time System
  - MCI: Model Coupling Interface
  - LOC: Library of Couplers
  - FE: Front-end
  - DE: Design Environment


SMF
Front end
Develop-
ment
environ-
ment
user
modeller
Coupled
models
Model
Coupling
Interface
Library of
Couplers
Models
and data
Run-time system
operator

Coupling data and models

- Conceptually, there is no difference between coupling
  - real-time data sources
  - archived data sources
  - computational tasks within one model
  - computational tasks of different models
  - ...
- The MCI should be applicable to all these situations.
- The modeller only has to worry about the MCI and should no longer think about I/O.



Sharing data between models

- Communicating models must know the format and meaning of the data imported/exported: they must use the same "vocabulary”.
- Current SMFs use two approaches :
  - A *fast* one : limit models to data structure predefined by the SMF.
    - E.g. ESMF offers mesh data structures + mesh libraries
  - A *general* one : to allow any kind of data structure to be exchanged, the SMF must offer a tool to describe the data structure + a dictionary to describe the metadata
    - E.g. HLA uses an object-oriented data description
    - E.g. VOTable is a format based on XML, representing data as a set of tables, with big-data and grid computing features
- The modeller should
  - provide a description of the format and meaning of the data.
  - use the MCI to perform all import/export operations.

Writing data descriptions

- The SMF should facilitate writing data descriptions
  - by offering assistance in writing standardized data descriptions.
    - "Federation development and execution process" (FEDEP, IEEE 1526.3-2003, IEEE 1730-2010 standard), defined in HLA terminology, but generally applicable.
  - by giving the modeller a broad set of standard data representations to/from which each model should convert its shared data.

- Metadata can, e.g., use the SPASE Data Model to describe data
  - access to resource descriptions stored in web registries of products, so as to query them to retrieve the data a user needs.
  - Interoperability with Virtual Observatories.

Coupling

2014: fiftieth years anniversary

Key dates

- The data exchange format and its meaning in practice always need to be well-documented. The MCI-imposed format description therefore does not really constitute extra work (at least not for a new model; it may be some work for porting an existing model).

- In some cases, the modeller can benefit from functionality in the MCI and LOC, e.g. interpolation in block-structured meshes.

- A combined approach limiting the effort is possible :
  - For new models, use MCI for all communication.
  - For rapid inclusion of an existing model, continue with existing internal communication structure, and use MCI only for the data exchange with other models.

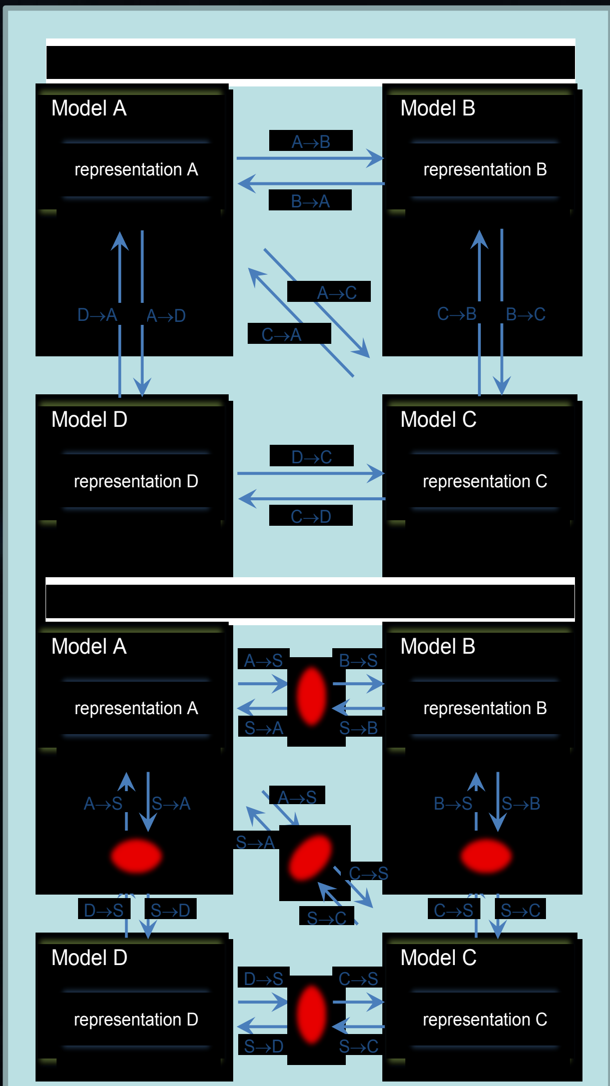
Model couplers

- The MCI must provide an open mechanism to include future models and their data structures.
- To allow communication, there must be couplers that translate the data between different formats and meanings.
- This determines what attributes must be associated with data.
  - Attribute = “units”
  - Coupler = “unit conversion”
- The modeller will benefit from building in as much functionality into the LOC as possible. This promotes code reuse; LOC routines can be optimized once and for all.

Coupling operations

- conversion of the data into common units;
- conversion of state quantities (e.g. temperature ↔ thermal velocity);
- interpolation of data over an n-dimensional regular/irregular mesh;
- convert a plasma representation defined in terms of a 3D VDF $f(v_x, v_y, v_z)$ into a moments-based representation (density, velocity, temperature, …);
- construct a 3D VDF $f(v_x, v_y, v_z)$ from given density, bulk velocity, temperature + an assumption;
- convert a 3D VDF $f(v_x, v_y, v_z)$ into a 2D VDF $f(v_\perp, v_\parallel)$ or $f(\mu, E)$;
- convert a set of particles $\{\vec{x}_i, \vec{v}_i\}$ into a a 3D VDF $f(v_x, v_y, v_z)$
- …
- The LOC should provide the most common of these convertor operations, ready for use for the modeller, but must be open to accept new ones.

Direct or indirect coupling



- Direct coupling:
  - more efficient
  - # couplers ~ # model pairs
  - know data structures in both models
  - who will write the couplers?
- Indirect coupling:
  - via an intermediate standard format
  - # couplers ~ # models
  - know data structures of only one model
  - minimum number of couplers to be written by modeller
  - loss of efficiency : 2 conversions needed
  - detect null operation A →S + S→A, typical for intra-model coupling

Configuration

- The modeller has to set up configurations using the DE, specifying the topology according to which
  - there is intra-model data exchange for his model;
  - there is data exchange with other models;
  - data are to be read for his model;
  - output has to be produced by his model.
- Once a configuration is given, it may be run as a simulation with support of the RTS.

- The DE can help in setting up the configuration. Based on knowledge of the metadata, the DE can suggest alternatives.

- The RTS can optimize physical inter-process communication by exploiting information about the volume of the data exchanges.

Promoting modularity

- With support from the DE, the modeller can easily swap models in a configuration – at least if the required couplers are available.

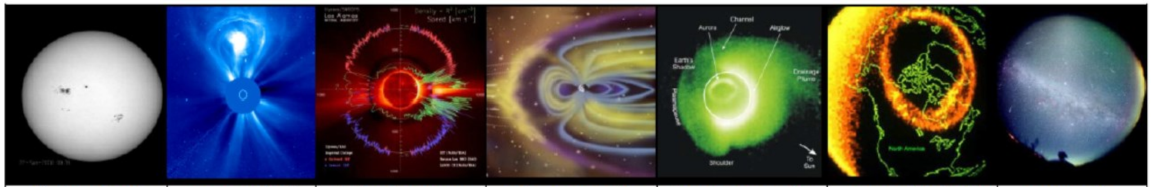
- This has multiple uses :
  - It provides easy debugging by mounting of a “virtual model” : replace a model component by another one that outputs pre-specified data.
  - It allows modellers to verify the quality of each model in the configuration and to check the correctness of the coupling.
  - The library of models should go from very simple ones to possibly very elaborate ones to allow modellers to find a compromise between quality and speed.
  - If no measurements are available for the boundary condition of a model, an empirical model can be used to supply the missing boundary condition, or a model that infers the info from other data.

Modelling uncertainty

- The SMF can itself generate alternate configurations and/or automatically generate simulation runs. Modellers would be able to use the following options:
  - performing ensemble simulations with perturbed inputs. Pro: evaluates nonlinear effects. Contra: extremely compute-intensive, verifies only the effect of errors on the data.
  - performing ensemble simulations with different numerical precision (e.g. different mesh resolution, different number of particles in a particle simulation) to evaluate the numerical technique.
  - performing simulations with different models so as to evaluate the overall effect of the particular physical and mathematical approximations.

Model availability

- The modeller benefits from the possibility to couple with a multitude of already existing models.



photospheric field Fromage	*coronal field* Fromage, AMRVAC, Flip3D-MHD	*interplanetary magnetic field* AMRVAC, Flip3D-MHD	*outer magneto-sphere fields* GUMICS-4, iPIC, COOLFluiD	*inner magneto-sphere magnetic and electric fields* GUMICS-4, iPIC	*magnetic field in the ionosphere* GUMICS-4	*magnetic field in the upper atmosphere*
	coronal plasma AMRVAC, Flip3D-MHD	*solar wind* AMRVAC, Flip3D-MHD	*boundary layer plasma, plasma sheet, lobe plasma* GUMICS-4, iPIC, COOLFluid	*plasmaspheric plasma, ionospheric outflow, auroral precipitation* GUMICS-4, iPIC, CTIP, PS-BISA	*ionospheric ions and electrons, aurora, conductivity* GUMICS-4, CTIM, TRANSCAR, IRI	
					thermosphere CTIM, DTM, TRANSCAR, IRI	*neutral atmosphere composition & winds* CMAT, DTM
				radiation belts BAS, Salammbô, RB-BISA		
				cosmic rays *		
			solar energetic particles SOLPENCO			
			meteorites and interplanetary dust Grün model			
			applications DICTAT			

Platforms

- A SMF should at some point be implemented as a distributed system that is providing full functionality, with a full complement of models and data sources, including real-time data feeds. The modellers should be allowed to work in this operational environment to validate their model and configurations using it, but under strict rules.

- The SMF should also be available on small platforms, then with only a subset of models in its repository, to allow the modeller all convenience of a local model development environment.

Centralized or distributed system?

2014: fiftieth years anniversary

Key dates

- A SMF may rely on a centralized or distributed communication approach.
  - *Centralized*: All inter-process communication goes through the central node. Advantage : couplers need to be available only on this central node. Disadvantage : not scaleable, may become a bottleneck
    - E.g. ViSpaNeT and HLA use a central server for the communication, but the federated tasks may be parallel applications based on distributed communications.
  - *Distributed*: All compute nodes are peers and may communicate directly with each other.
    - E.g. SWMF offers distributed communications based on MPI.

- It doesn’t really matter to the modeller.

SMF performance

- In reality, there will be a trade-off between parallel efficiency and the wall clock execution time of a simulation, i.e. the timeliness of the result.
  - *Scientific applications*: emphasis on an efficient use of the hardware.
  - *Operational forecasts*: minimize wall clock execution time: assign more computational resources to make the forecast faster, but machine use is less efficient.

- The two might be implemented in 2 separate systems:
  - an on-request system (batch) and
  - an offered-service system (real-time).

- The modellers benefit from the provided systems.

Debugging and performance

- The MCI, LOC, and model implementations can be instrumented so that the RTS can collect run-time information about the system.
  - Allows monitoring the system for fault detection.
  - Helps in debugging.
    - Example: XPVM for PVM environment.

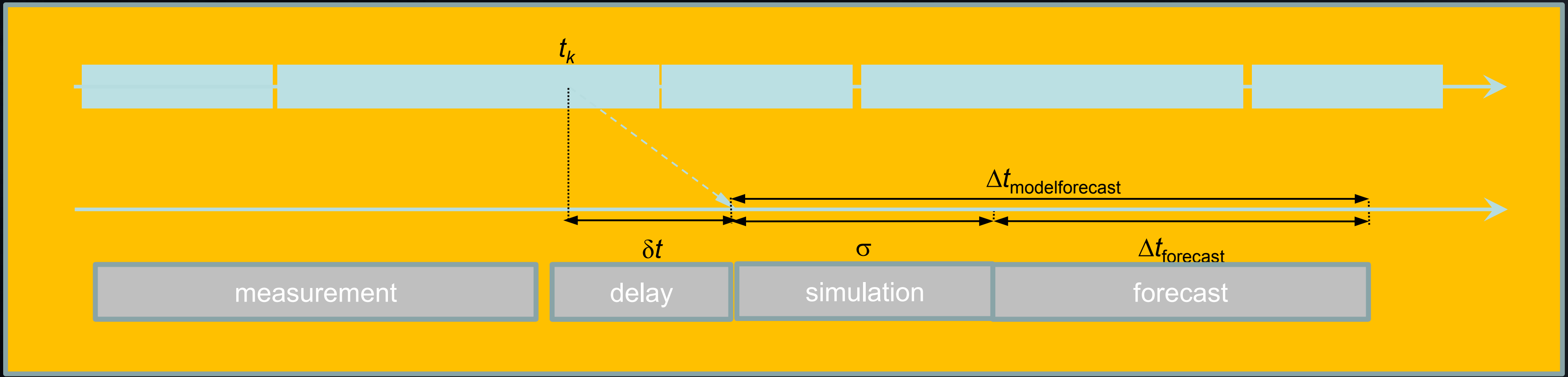
- Principle
  - The model can be linked to an alternative version of the MCI, LOC, and model libraries in which the MCI primitives and LOC couplers, and even models, collect timing and numerical info.
  - This info is logged in a trace file, which can be read by a GUI-based analysis tool.
  - The modeller optionally can contribute logging information.
  - The modeller can use the RTS debugging/monitoring facilities.

Usage statistics

- The RTS performance monitoring tool could be coupled to the DE to relate the performance metrics to the source code :
  - What parts of the code have been used most often?
  - What options have been given to the code?
  - What domain sizes, spatial or time resolutions have been used?

- The SMF system should give model developers access to usage statistics and collected performance information for their model.

Forecasting quality



- A particular performance quantity that can be monitored in a real-time SMF is the forecast quality.
- Forecast quality is related to forecast lead time; lead time improves
  - if the forecasting horizon of the model is longer,
  - if the delay with which the data are received is shorter, and
  - if the simulation is run faster.
- An automatic prediction chain can be set up so as to continuously compare a predicted quantity with its true value as measured later. Forecasting quality statistics can thus be provided to the modeller.

Dealing with uncertainty

- Forecast quality can only be evaluated if the forecast is accompanied by an uncertainty margin. The quality of a forecast depends on
  - precision of the data,
  - sensitivity of the model system state to the data,
  - precision of the model with its limitations due to the physical description, the chosen mathematical representation, the numerical approximations, and arithmetical round-off.
- Modellers and data sources should therefore provide error estimates on their data.
- If hard to do and/or computationally intensive, suitable estimates can be used.

Visualization & post-processing

- Since the MCI provides standardized data formats, it must furnish the modellers with the associated visualization and analysis tools. For complex multi-domain, multi-physics simulations, data interpretation is not very straightforward any more …

- The metadata further allow the visualization tools to adapt their behaviour to the specific data at hand.

Conclusions

- Model developers will be the first users of a SMF. They will have to integrate their models into the system, and they will have to test them. The SMF can offer testing facilities, performance reporting tools, to facilitate the effort. Give 7 – Take 27

- Modellers should be interested in a SMF as it could help them:
  - To compare their model to others;
  - To obtain the relevant input to run their models, either from data sources or from a coupled model;
  - To demonstrate the usefulness of their model by providing the output of their model as input to another model, possibly a model of a specific space weather effect.

- An open shared modelling framework may avoid reinventing the wheel each time.