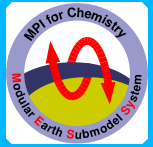




(joeckel@mpch-mainz.mpg.de)

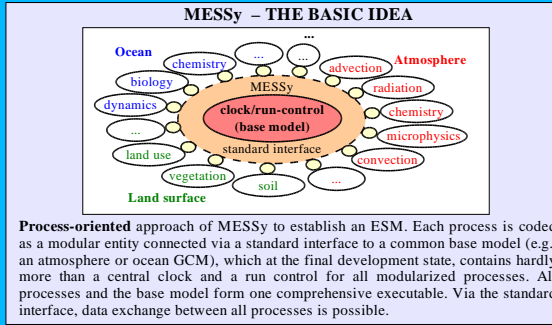
The Modular Earth Submodel System (MESSy): A technical overview

Patrick Jöckel¹, Rolf Sander¹, Andreas Rhodin², Jos Lelieveld¹
¹Max-Planck-Institute for Chemistry, Mainz, Germany, ²German Weather Service (DWD)



<http://www.messy-interface.org>

Abstract. The transition from General Circulation Models (GCMs) to Earth System Models (ESMs) requires the development of new software technologies, since the rapidly increasing model complexity needs a transparent control. The Modular Earth Submodel System (MESSy) provides a generalized interface structure which allows unique new possibilities to study feedback mechanisms between various bio-geo-chemical processes. Strict compliance with the ISO-Fortran95 standard makes it highly portable to different hardware platforms. The modularization allows for uncomplicated connection to various base models, as well as the co-existence of different algorithms and parameterizations for the same process, e.g. for testing purposes. The coding standard provides a multi-developer environment, and the flexibility enables a large number of applications with different foci in many research topics referring to a wide range of temporal and spatial scales. We present an overview of the basic interface structure and its technical implementation, and offer it to the scientific modeling community.



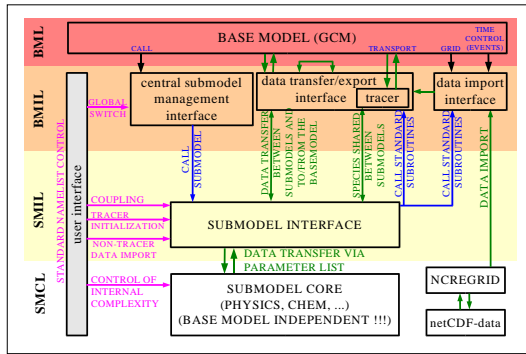
Process-oriented approach of MESSy to establish an ESM. Each process is coded as a modular entity connected via a standard interface to a common base model (e.g., an atmosphere or ocean GCM), which at the final development state, contains hardly more than a central clock and a run control for all modularized processes. All processes and the base model form one comprehensive executable. Via the standard interface, data exchange between all processes is possible.

MESSy - THE FUTURE

MESSy is an activity that is open to the scientific community following the "open-source" philosophy. We encourage collaborations with colleague modelers, and aim to efficiently achieve improvements. Additional submodels and other contributions from the modeling community will be highly appreciated. We encourage modelers to adapt their code according to the MESSy standard.



MESSy - THE 4-LAYER INTERFACE STRUCTURE



The user interface is implemented by using the Fortran95 namelist constructs, and is connected to the three layers BMIL, SMIL, and SMCL. The global switch to turn the submodel on/off is used in the BMIL. These switches for all submodels are set by the run script in the file `MESSy.nml`. Submodel-specific data initialization (e.g., initialization of chemical species (= tracers), `<SUBMODEL>_t_nml`), and import of data during the time integration (e.g., temporal changing boundary conditions, `<SUBMODEL>.nml`) using the data import interface are handled by the SMIL. Within the SMIL, also the coupling options (`&CPL`) from the user interface are evaluated and applied, which determine the way of coupling of the submodel to the base model and to other submodels. For instance, the user has here the choice to select input to the submodel from alternative sources, e.g., from results calculated online by other submodels, or from data provided offline. Therefore, this interface allows a straightforward implementation and management of feedback mechanisms between various processes. The control interface is located within the SMCL and manages the internal complexity (and with this also the output) of the submodel (`&CTRL`). It comprises, for instance, changeable parameters for the calculations, switches for the choice of different parameterizations, etc.

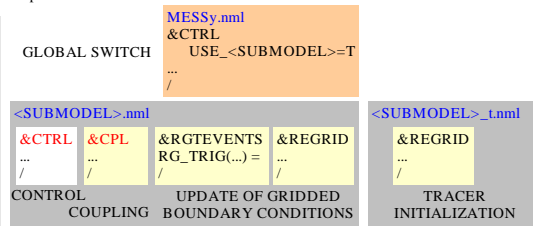
The Base Model Layer (BML): At the final development state, this layer comprises only a central time integration management and a run control facility for the involved processes. At the transition state (as currently) the BML is the domain specific model with all modularized parts removed. For instance, in case of an atmospheric model it is usually a GCM.

The Base Model Interface Layer (BMIL) comprises basically three functionalities:

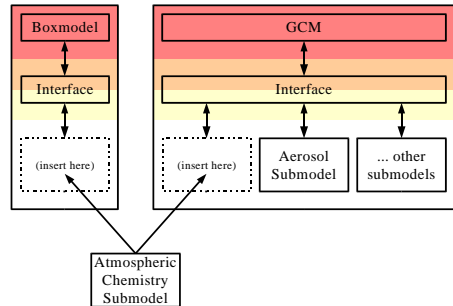
- The central submodel management interface allows the base model to control (i.e., to switch and call) the submodels.
- The data transfer/export interface organizes the data transfer between the submodels and the base model and between different submodels. It is further responsible for the output of results (export). According to the requirements of the model setup, the data can be classified according to their usage, e.g., as physical constants, as time varying physical fields, and as tracers (i.e., chemical compounds).
- The data import interface is used for flexible (i.e., grid independent) import of gridded initial and time dependent boundary conditions. The BMIL therefore comprises the whole MESSy infrastructure which is organized in so called generic submodels.

The Submodel Interface Layer (SMIL): This layer is a submodel-specific interface, which gathers all relevant information/data from the BMIL, transfers them via parameter lists to the SMCL (see below), calls the SMCL routines, and distributes the calculated results from the parameter lists back to the BMIL. Since this layer performs the data exchange for the submodel, also the coupling/feedback between different submodels is managed within this layer.

The Submodel Core Layer (SMCL): This layer comprises the self-consistent core routines of a submodel (e.g., chemical integrations, physics, parameterizations, diagnostic calculations, etc.), which are completely independent of the implementation of the base model. Information exchange is solely performed via parameter lists of the subroutines. The output is completely determined by the input.



MESSy - PLUG & PLAY



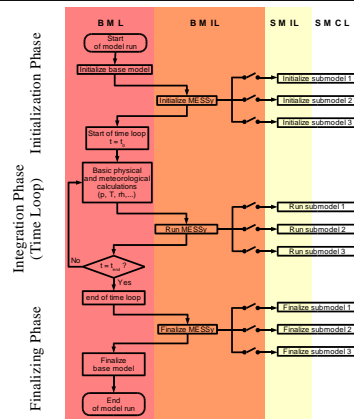
The 4-layer interface structure of MESSy, in particular the basemodel independent coding of the submodel core layer (SMCL) allows code sharing between different models. Here, for instance, a module for atmospheric chemistry integrations is shared between a box model and a GCM.

- **Flexibility:** Several alternative implementations for the same process can coexist in the same model system, e.g., different parameterizations for sensitivity studies, different approaches for process studies, "frozen" and developer versions can be directly compared.
- **Plug & play:** The implemented processes can be easily exchanged between different model systems.
- **Test facility:** The implemented processes can be tested without running the entire model system, i.e., for instance coupled to a simple box model.
- **Security:** Parameterizations terminate the model system if they are used outside their valid range. Unwanted or uncontrolled extrapolations are avoided.
- **Coupled system:** Feedback mechanisms can be easily implemented, controlled, and quantified.
- **Multi-purpose:** The model system can be applied to a wide range of scientific tasks, especially with respect to the spatial and temporal scales, the processes involved, and the domains covered.
- **Portability:** The model system is highly portable and runs on various different computer architectures.
- **Expandability:** The model system structure allows the straightforward adaption of additional processes and is prepared for future contributions.
- **Multi-developer:** The model system can be further developed by more than one person at the same time without interference.

- **Consistency:** The model system is highly consistent, all implemented processes share the same fundamental data sources.
- **Efficiency:** The model system code is highly efficient regarding usage of computer time.
- **Reproducibility:** Re-compilation of the code is avoided whenever possible, at least within one model simulation including sensitivity studies. Especially the choice of process specific parameters, the coupling of different processes, and the choice of available alternatives must not require a recompilation of the code. Note: In coupled complex (non-linear) systems, re-compilation bears the risk of losing reproducibility due to uncontrollable compiler issues.
- **Variable complexity:** The internal complexity of each process can be changed according to its relevance in different applications.
- **Synergy:** Implementations relevant for different processes are shared.
- **User friendly:** The model system comprehends a unified, transparent user interface for the control of the model system.
- **Modularity:** Each specific process is coded as a separate, independent entity, i.e., as a submodel, which can be switched on/off individually.
- **Self-consistency:** Each submodel is completely self-consistent, the submodel output is completely defined by its numerical input.
- **Portability:** All submodels are coded according to the language standard of Fortran95 (ISO/IEC-1539-1). The submodel code is free of hardware vendor specific language extensions. In the rare cases where hardware specific code is unavoidable (e.g., to circumvent compiler deficiencies), it is encapsulated in preprocessor directives.

MESSy - ADVANTAGES

MESSy - FLOW CHART



A model simulation of a typical base model / MESSy setup can be subdivided into three phases. The main control (time integration and run control) is hosted by the base model, and therefore the base model is also responsible for the flow of the three phases:

- **Initialization phase:**
 - initialization of the base model
 - initialization of the MESSy infrastructure (generic submodels); activation of submodels
 - sequential initialization of activated submodels (internal control)
 - sequential initialization of activated submodels (coupling to base model and other submodels)
- **Time integration phase:**
 - integration of the base model
 - sequential integration of the activated submodels (operator splitting)
- **finalizing phase:**
 - finalizing of the submodels
 - finalizing of the MESSy infrastructure (generic submodels)
 - finalizing of the base model
 - termination of the run

- **Standard interface:** A so-called base model provides the framework to which all submodels are connected. At the final state of the development the base model must not contain more than a central clock for the time control (time integration loop) and a flow control of the involved processes (= submodels). This ultimate aim can be reached step by step. For instance one could start from an existing GCM (as in our example) and connect new processes to it via the standard interface. At the same time, it is possible to modularize processes which are already part of the GCM, and reconnect them via the standard interface. In many cases this requires only a slightly modified reimplementation based on the existing code.
- **Resolution independent:** The submodel code is completely independent of the spatial (grid) and temporal resolution (time step) of the base model. If applicable and possible, the submodels are also independent of the dimensionality (0D (box), 1D (column), 2D, 3D) and the horizontal (regional, global) and vertical domain of the base model. Each process is coded for the smallest applicable entity (box, column, column-vector, ...).
- **Data flow:** Exchange of data between the submodels and also between a submodel and the base model is standardized.
- **Soft-coding:** The model code does not contain any "hard-coded" specifications which require a change of the code and recompilation, if the model domain or the temporal or spatial resolution is changed. A prominent example is to use height or pressure for parameterizations of vertical profiles, instead of level indices, because the latter have to be changed, if the vertical resolution of the base model is altered.