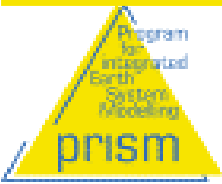




PRISM

Standard Compile Environment (SCE)

Stephanie Legutke, MPI-M M&D



Mission of PRISM (Dec 2001-Nov 2004):

Develop for the European Earth System (ES) modelling community a software infrastructure to

- **set up**
- **compile**
- **run**
- **analyse**

coupled ES model experiments

and thereby establish a European ES research network

M&D:

Develop SCE and SRE in the 'Modell Assembly' WP



... for the SCE infrastructure

- **Integrate 'any' ES climate research model ...**
- **... on 'any' platform**
- **Allow for easy replacement of components in ESM (component modularity)**
- **Interface to a GUI**



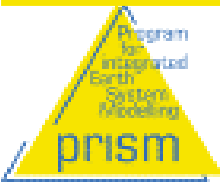
... raised by model developers and users

- **Minimum impact on component source code**
- **Give a common look&feel with all models and experiments**
- **Automate but allow for easy customizing**
- **Keep it simple**
- **[Low maintenance]**

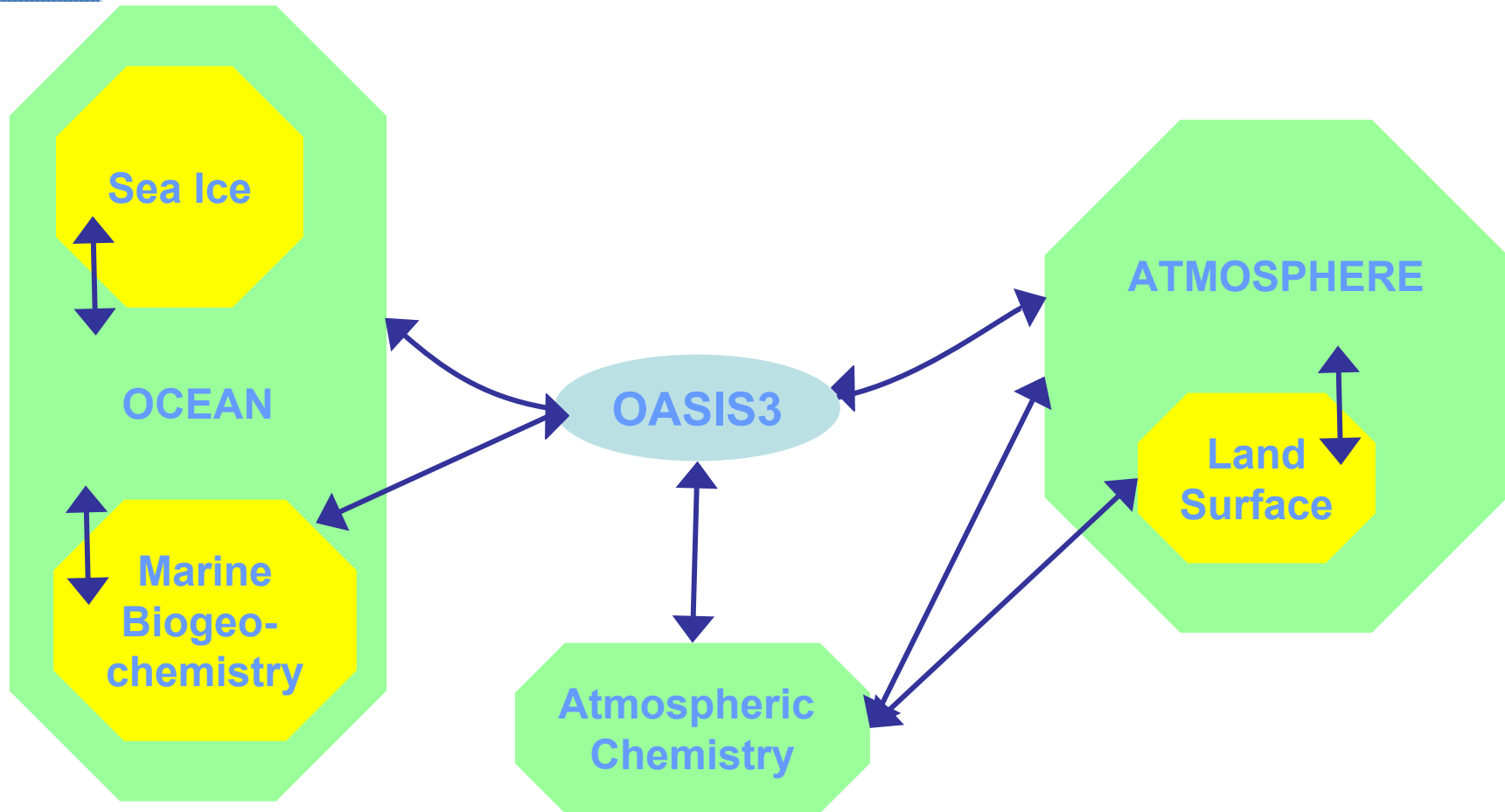


Components of the SCE

- **High level design for component modularity**
- **Component and coupled models, main and submodels, and libraries**
- **Standards and coding rules**
- **Compile scripts and their generation**
- **Tools: (g)make, perl, m4**
- **Planned developments**

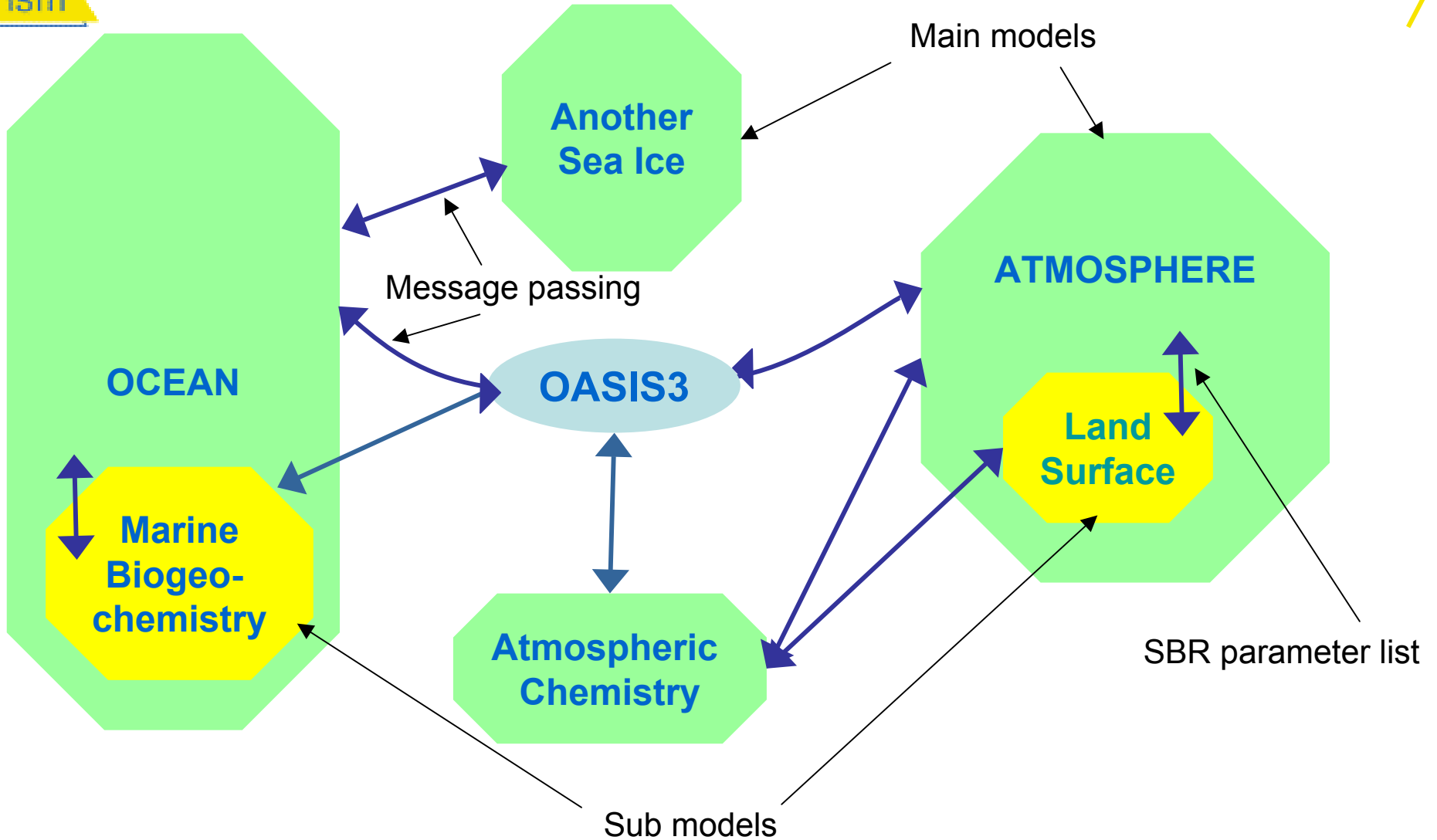


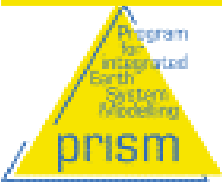
Coupled Earth System Models





Coupled Earth System Models



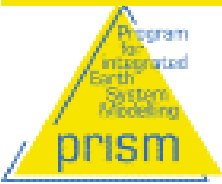


Libraries:

- Not configurable
- Can be precompiled
- Usable by all components

Models:

- Configurable
- Can not be precompiled
(until after make has learned to react on changing cpp flags)



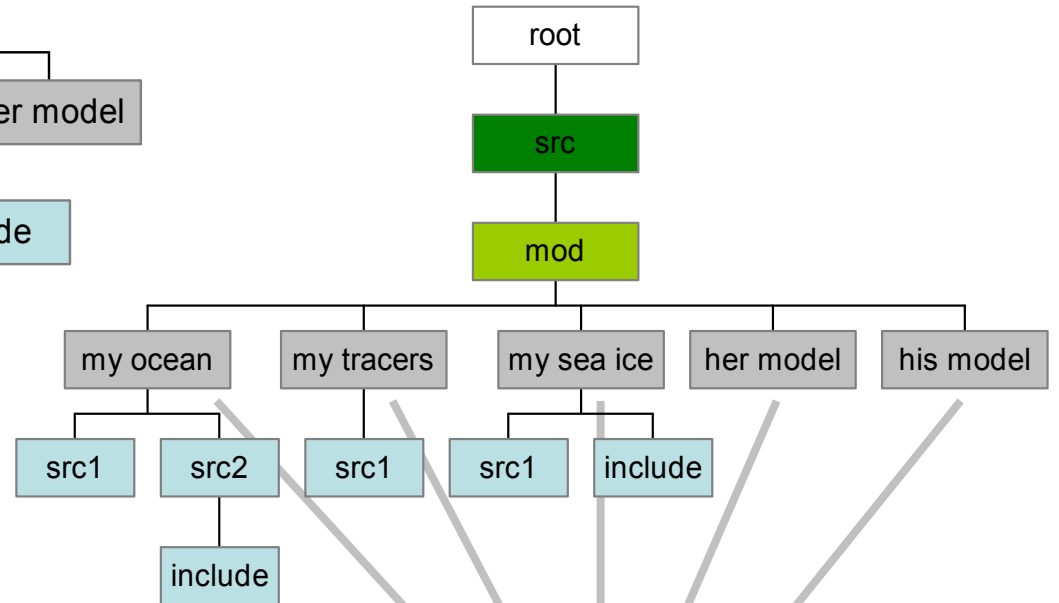
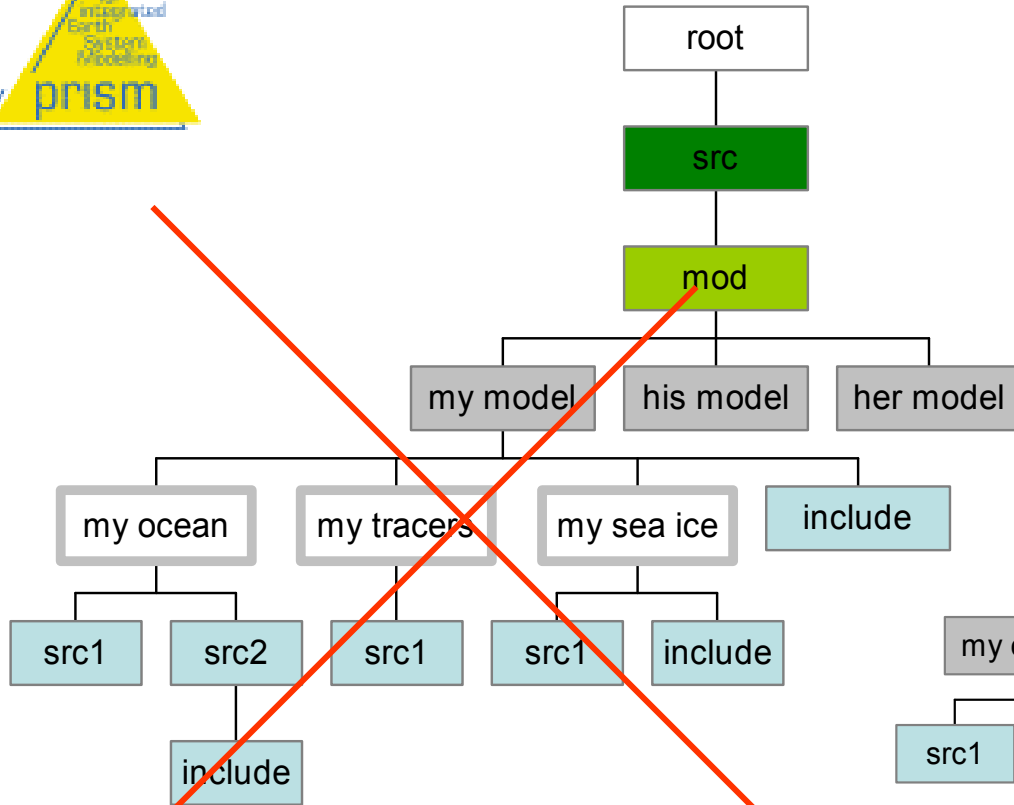
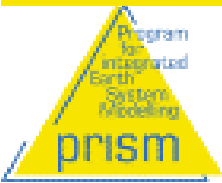
SCE standards for

... component model codes

- Components have to be independent from each other
- Standard directory structure

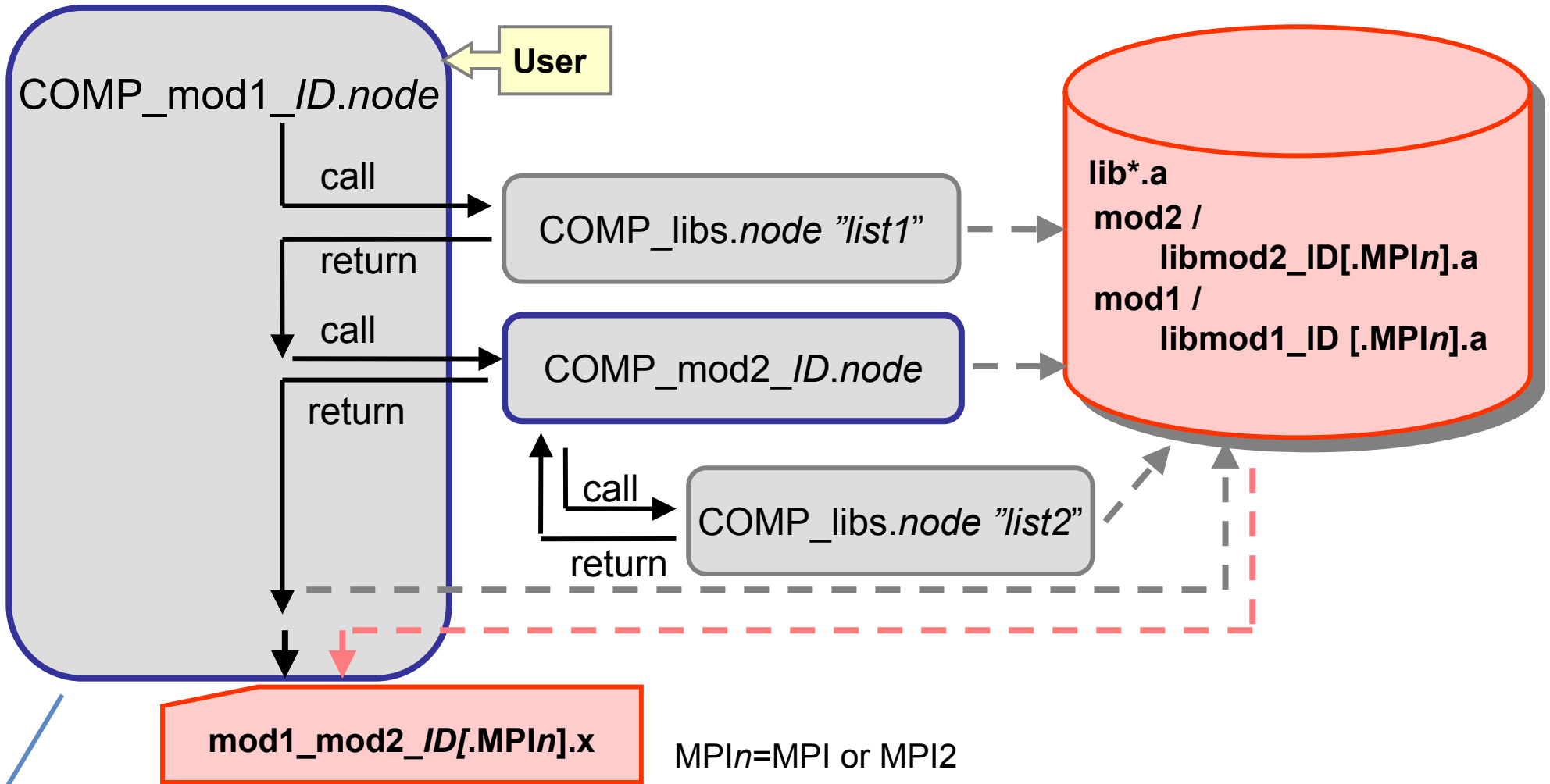
... source code

- One programming unit per file (Single F90 MODULE per file)
- File name = unit (MODULE, SBR) name
- Single occurrence of each basename
- Suffixes: F90, f90, F, f, c

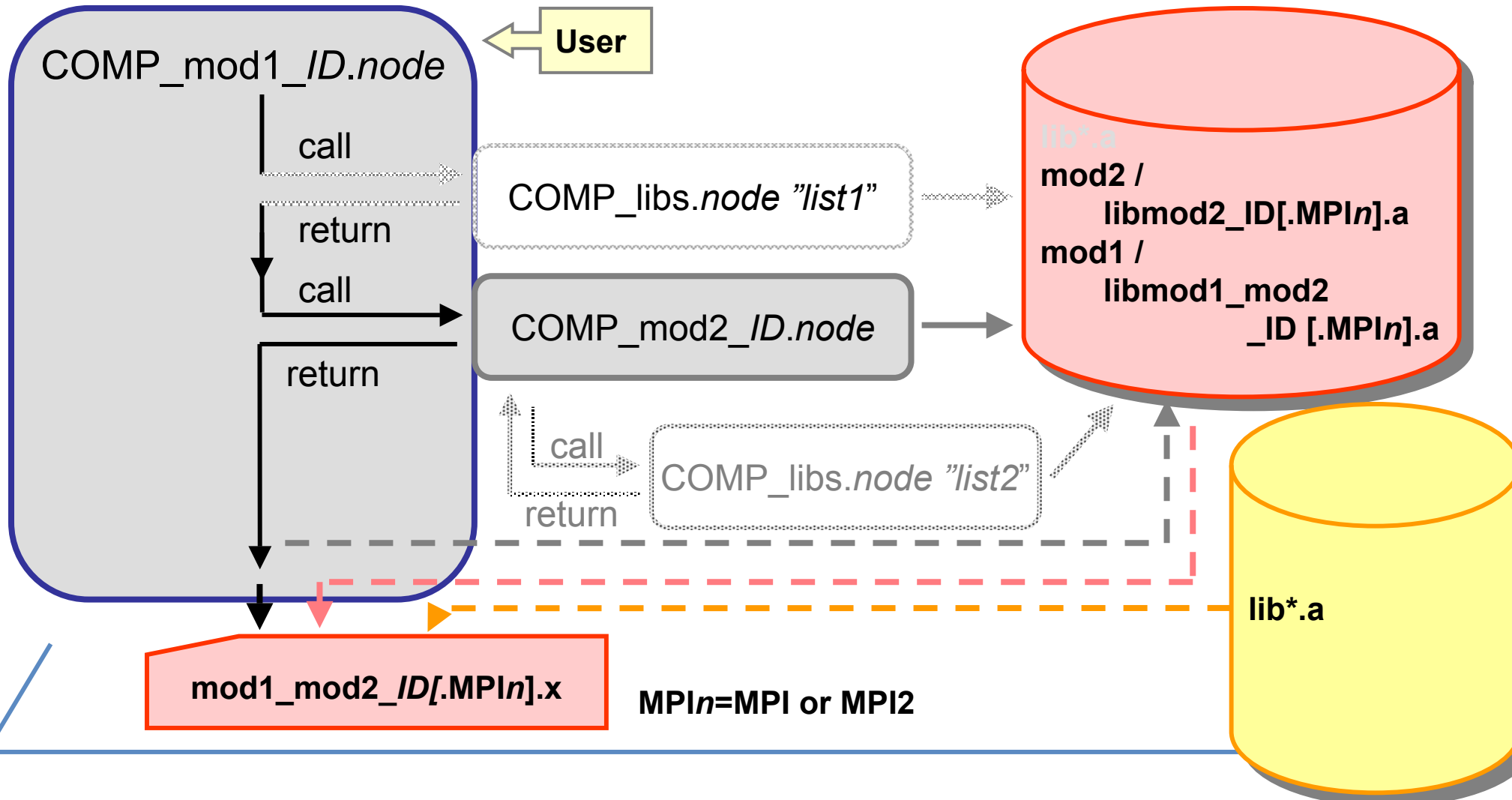


Libmy_ocean.a
libmy_tracers.a

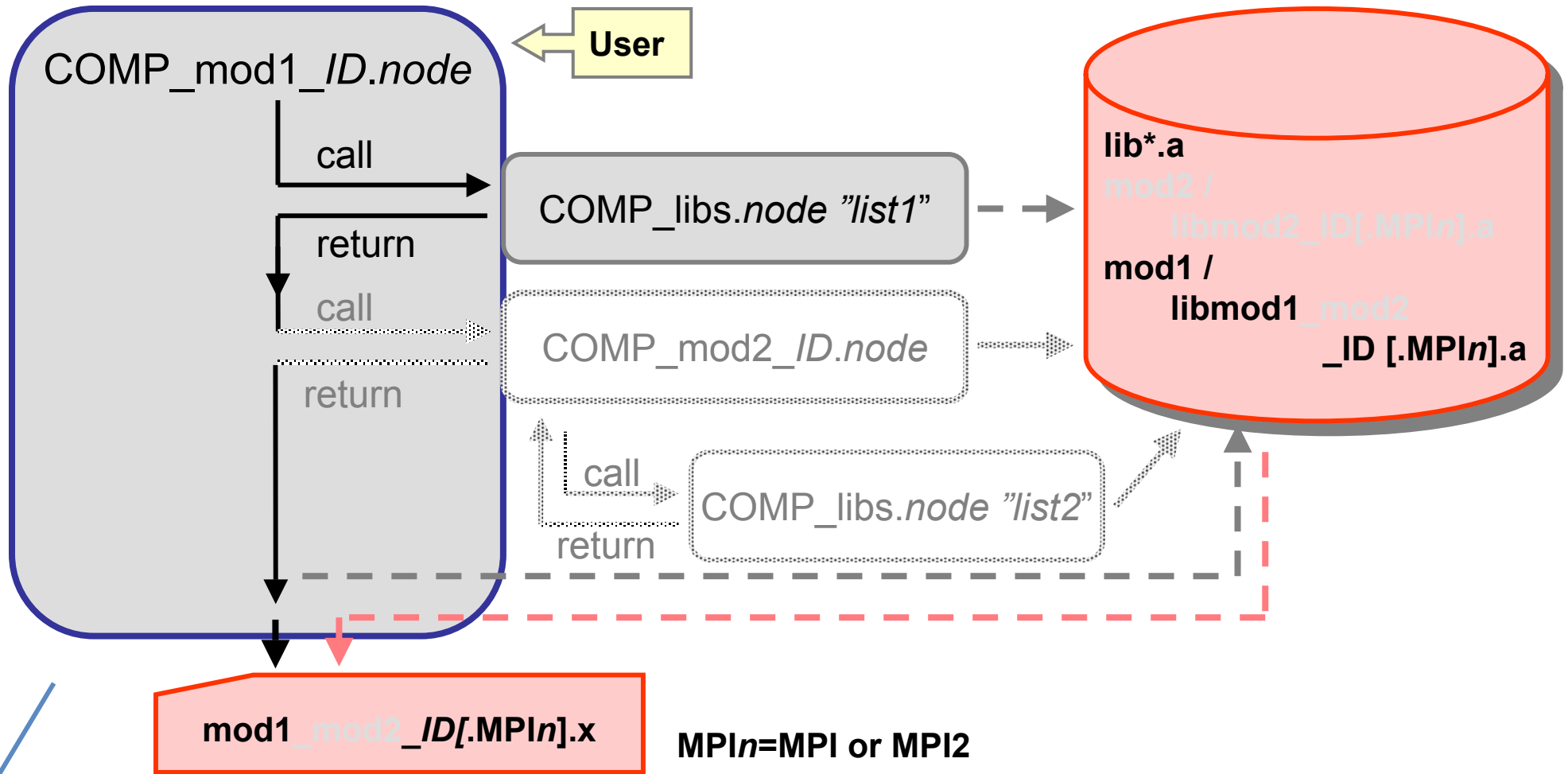
Model + submodel compilation flow

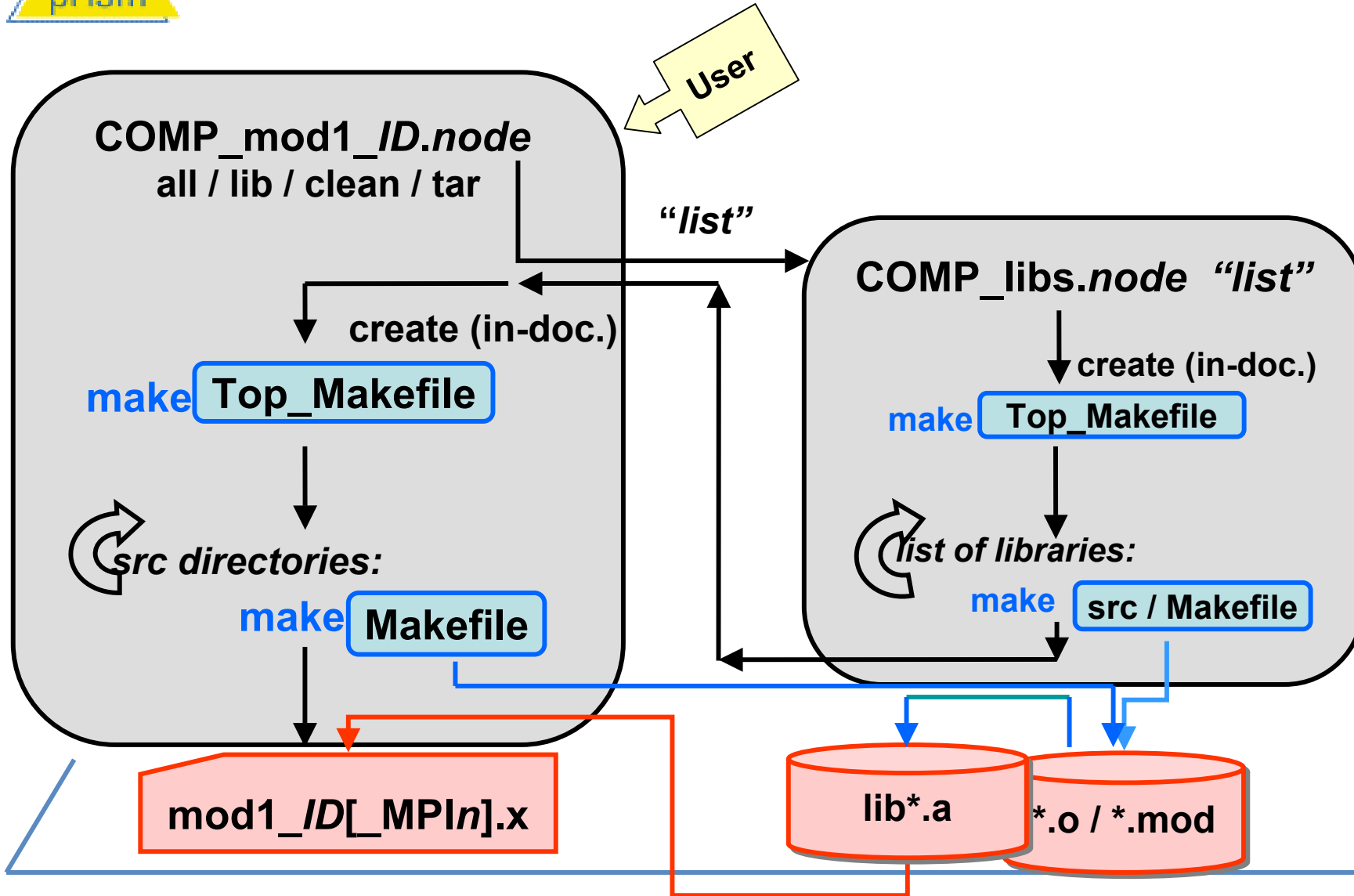


Compilation flow with central libraries



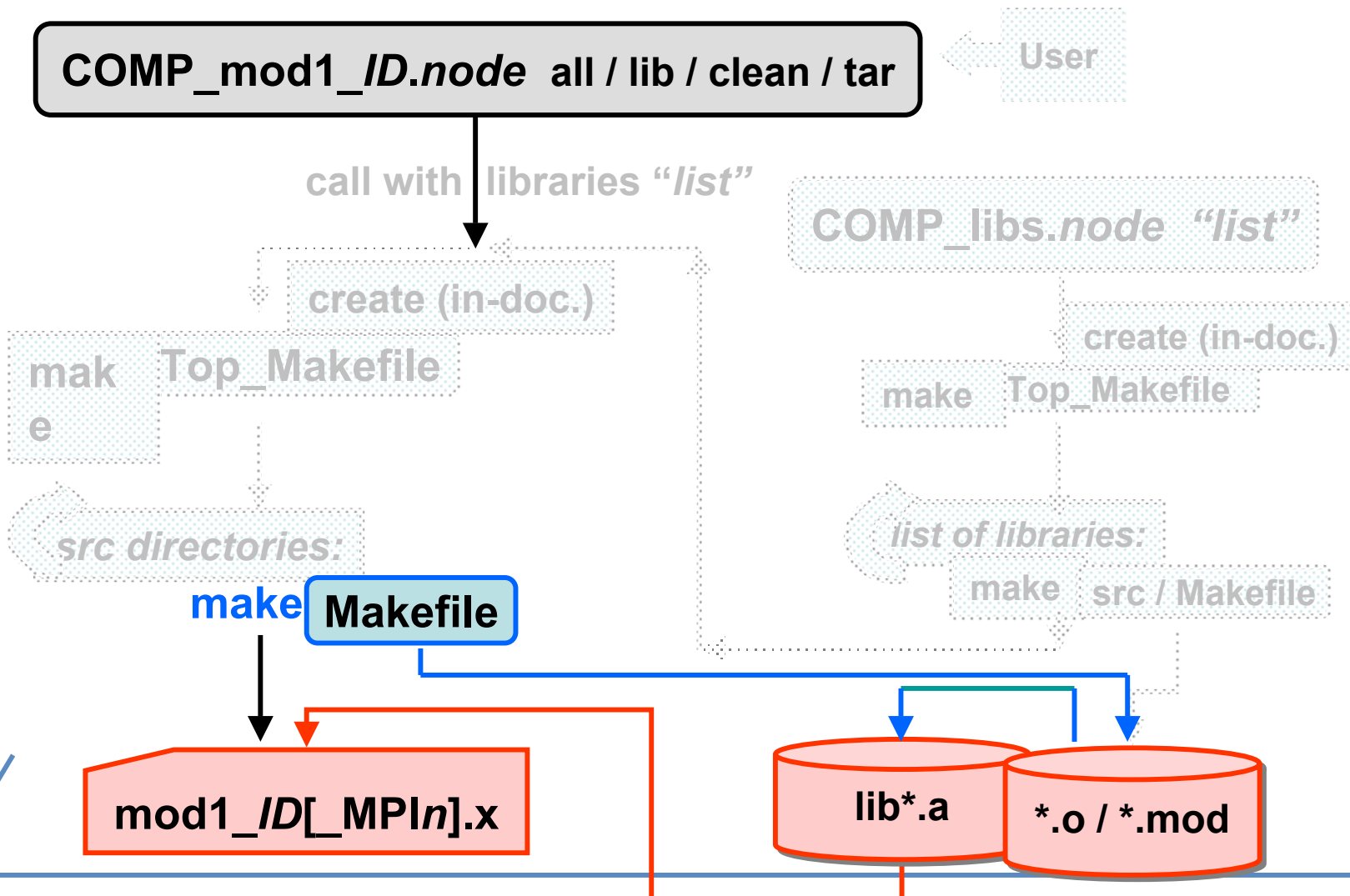
Component model compilation flow

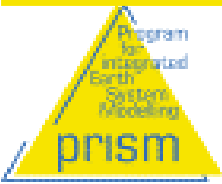






SCE : Model compilation





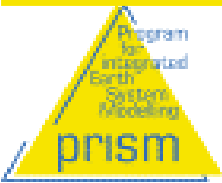
All compilation is based on the (g)make software:

Make 'targets' having 'prerequisites' with well defined 'rules' while avoiding redundant actions:

Make -f Makefile

Targets, rules & prerequisites are defined in Makefile:

```
...  
Target: prerequisite1, prerequisite2 ....  
        echo prerequisite1 and prerequisite2 are ready  
...
```



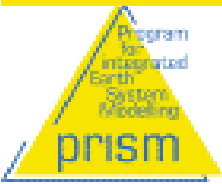
(GNU) Make targets

- **Executables:** OASIS3.x, ECHAM5.x, *.x
- **Libraries:** *.a
- **Model libraries:** *_ID.a
- **Binary object code:** *.o
- **FORTRAN90 Modules:** *.mod
- **Clean build directories**
- **Tar files**
- **... easily extended**



(GNU) Portable Makefiles

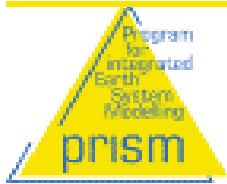
- All non portable code exported from the compile scripts
- 1 Makefile for each source directory with compiler input
- All CM Makefiles look the same
- All library Makefiles look the same
- Exception: prerequisites
perl-Tool for generation of *.o prerequisites



Generating .o prerequisites

```
mod_unitncdf.o: mod_kinds_oasis.o
alloc_src.o: mod_anais.o mod_analysis.o mod_coast.o mod_experiment.o \
    mod_extrapol.o mod_kinds_oasis.o mod_memory.o mod_nproc.o \
    mod_parallel.o mod_parameter.o mod_pipe.o mod_rainbow.o mod_sipc.o \
    mod_string.o mod_timestep.o mod_unitncdf.o
dealloc_src.o: mod_anais.o mod_analysis.o mod_coast.o mod_experiment.o \
    mod_extrapol.o mod_kinds_oasis.o mod_memory.o mod_nproc.o \
    mod_parallel.o mod_parameter.o mod_pipe.o mod_rainbow.o mod_sipc.o \
    mod_string.o mod_timestep.o mod_unitncdf.o
extrap.o: mod_extrapol.o mod_kinds_oasis.o mod_parameter.o mod_printing.o \
    mod_unit.o
getfld.o: mod_analysis.o mod_clim.o mod_experiment.o mod_hardware.o \
    mod_kinds_oasis.o mod_label.o mod_memory.o mod_parameter.o \
    mod_printing.o mod_sipc.o mod_string.o mod_timestep.o mod_unit.o \
    mod_unitncdf.o netcdf.inc
```

Append_Dependencies *model_name* *src_directory*

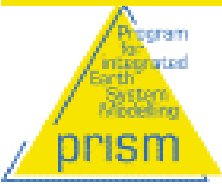


No compile scripts are provided ...

... but tools that generate them ...

- **... for the specific platform**
- **... for the specific component model**
- **... configured for the coupled configuration**

... Example how to do that =>



Creating compile scripts (local system)

bold: to be typed by user (scripting)
additional configuring by editing

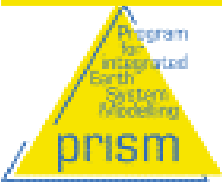
ESM name = **cplmod** = **mod1**+**mod2** & **mod3** & OASIS3:

Create_COMP_cpl_mod.ksh **cplmod** [MPI1 [ID [node]]]

```
Create_COMP_models.frm mod1 MPI[1,2] - - node ID "mod1 mod2 mod3"  
Create_COMP_models.frm mod2 MPI[1,2] - - node ID "mod1 mod2 mod3"  
Create_COMP_models.frm mod3 MPI[1,2] - - node ID "mod1 mod2 mod3"  
Create_COMP_models.frm oasis3 MPI[1,2] - - node  
Create_COMP_libs.frm MPI[1,2] - - node
```

```
COMP_mod1_ID.node  
COMP_mod2_ID.node  
COMP_mod3_ID.node  
COMP_oasis3.node  
COMP_libs.node
```

```
mod1_mod2_ID.MPI[1,2].x  
mod2_ID.MPI[1,2].a  
mod3_ID.MPI[1,2].x  
oasis3.MPI[1,2].x  
.... , psmile.MPI[1,2].a
```



Creating compile scripts (central system)

ESM name = **cplmod** = **mod1+mod2** & **mod3** & OASIS3:

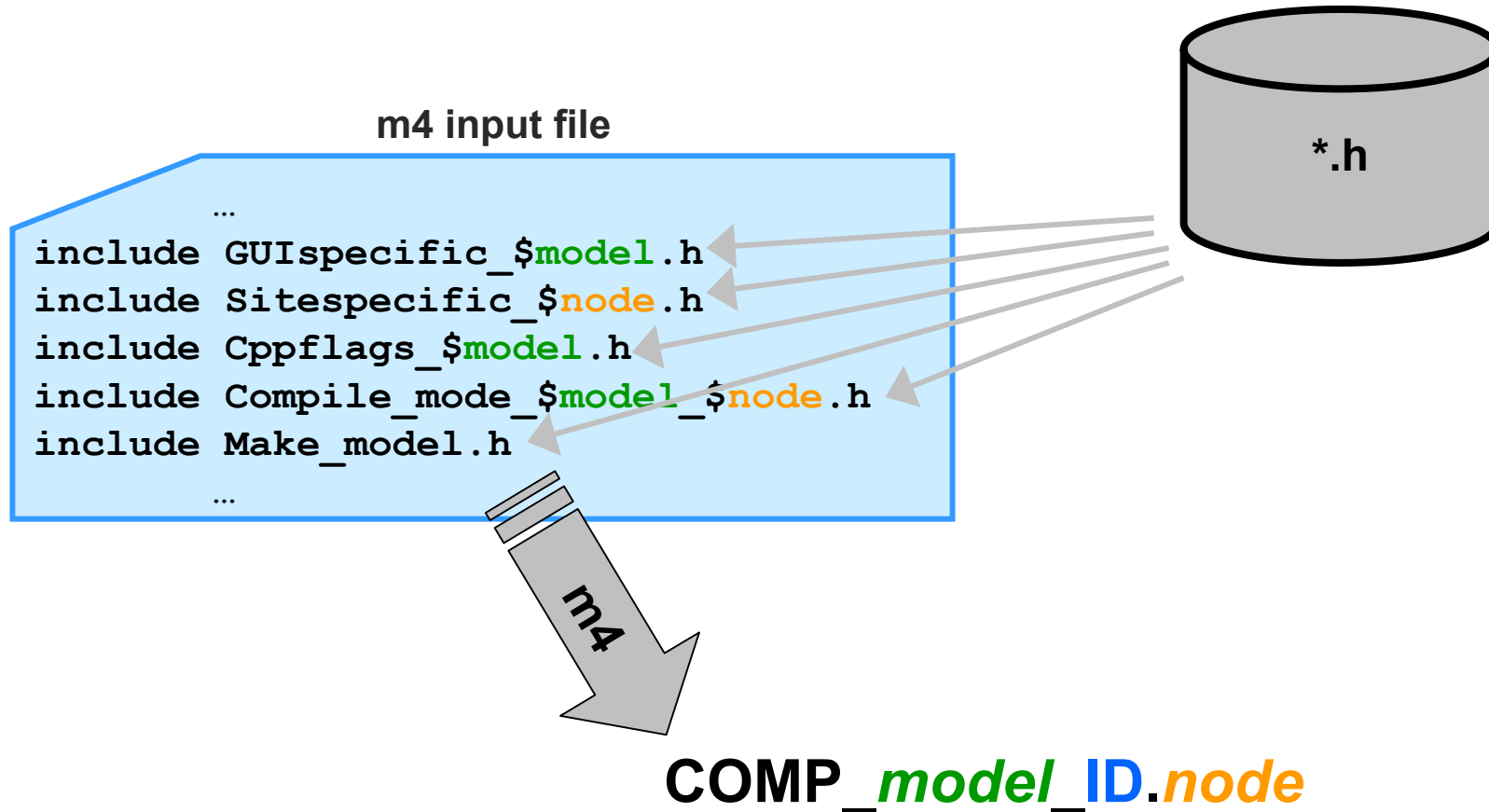
Create_COMP_cpl_mod.ksh **cplmod** [MPI1 [ID [node]]]

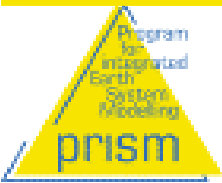
```
Create_COMP_models.frm mod1 MPI[1,2] - - node ID "mod1 mod2 mod3"  
Create_COMP_models.frm mod2 MPI[1,2] - - node ID "mod1 mod2 mod3"  
Create_COMP_models.frm mod3 MPI[1,2] - - node ID "mod1 mod2 mod3"  
Create_COMP_models.frm oasis3 MPI[1,2] - - node  
Create_COMP_libs.frm MPI[1,2] - - node
```

```
COMP_mod1_ID.node  
COMP_mod2_ID.node  
COMP_mod3_ID.node  
COMP_oasis3.node  
COMP_libs.node
```

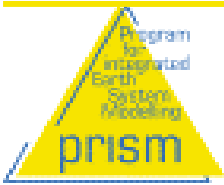
```
mod1_mod2_ID.MPI[1,2].x  
mod2_ID.MPI[1,2].a  
mod3_ID.MPI[1,2].x  
oasis3.MPI[1,2].x  
.... , psmile.MPI[1,2].a
```

Create `COMP_models.frm` *model* “” – “” *node ID* “*partner models*”





- Adapted Models
 - ECHAM5, Arpege, LMDZ, TOYATM, TOY4OPA
 - MOZART2, TM5
 - ORCHIDEE
 - MPI-OM, OPA, TOYOCE, TOY4Arpege, NEMO
 - HAMOCC, PISCES
 - LIM
- Coupled Models
 - ECHAM5, ECHAM5+MPI-OM, ECHAM5+MPI-OM+HAMOCC, ECHAM5+MPI-OM+PISCES
 - MPI-OM, MPI-OM+HAMOCC, MPI-OM+PISCES
 - LMDZ+ORCHIDEE+Lim+OPA
 - Others done by model owners ...



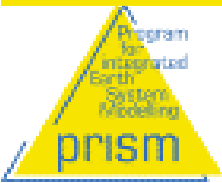
- **PRISM web site**
<http://prism.enes.org>
- **Handbook**
The PRISM Standard Compilation Environment
Stephanie Legutke and Veronika Gayler
PRISM Report Series, No. 4, 1. Edition, 15 Jan 2005
<http://prism.enes.org/Publications/Reports>
- **WP3i (Model Assembly) M&D PRISM web site**
<http://prism.dkrz.de/Workpackages/WP3i/Documentations>



**“Instead of trying to foresee the future,
we have added features as required
and will continue to do so.”**

J. Gregory (2003)

- ... if it fits into the system and philosophy
- ... if it is really needed
- ... If we have the time to do so



- **Enable compilation of (ordered) subset of code base files**
- **Teach *make* to react on cpp flags to allow for precompiled model code**
- **Clearer distinction of local / central installations**
- **Revise the user interface to the scripting system (interface with GUI?)**
- **Improve documentation**
- **Include OASIS4 coupled models**
- **Allow for more complex directory structure**
- **Full generation of Makefiles (libraries and components)**



- The End -