



Modelling COSMOS in the Integrating Model and Data Infrastructure

- Overview
- Integrated Data Management (IDM)
- Source Code Management (SCM)
- Standard Compile Environment (SCE)
- Standard Running Environment (SRE) (next talk by Heiner)



- Overview
 - ◆ History
 - ◆ High level design
 - Modularity
 - Interoperability
 - ◆ The COSMOS model family in IMDI
- Integrated Data Management (IDM)
- Source Code Management (SCM)
- Standard Compile Environment (SCE)
- Standard Running Environment (SRE)



IMDI: History



EU FP5 Project for Integrated Earth System Modeling (PRISM)

Dec 2001-Nov 2004:

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

- **compile**
 - **run**
 - **set up**
 - **analyse coupled ES model experiments**
- and thereby establish a European ES research network**

Since 2005:

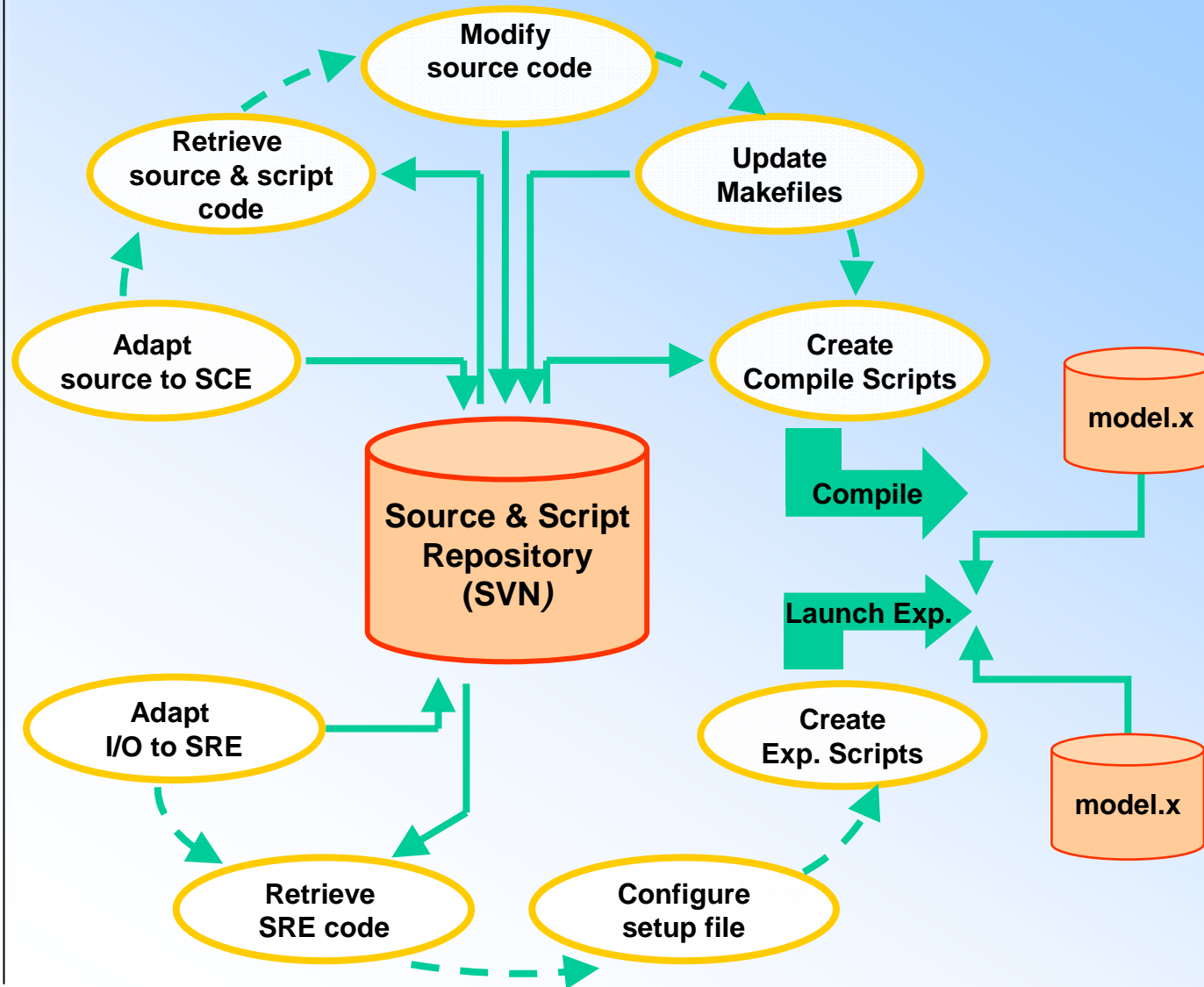
- **cooperation with PRISM Support Initiative (PSI)**
see <http://prism.enes.org>
- **continued development of SCE/SRE**
- **integration of data aspects**



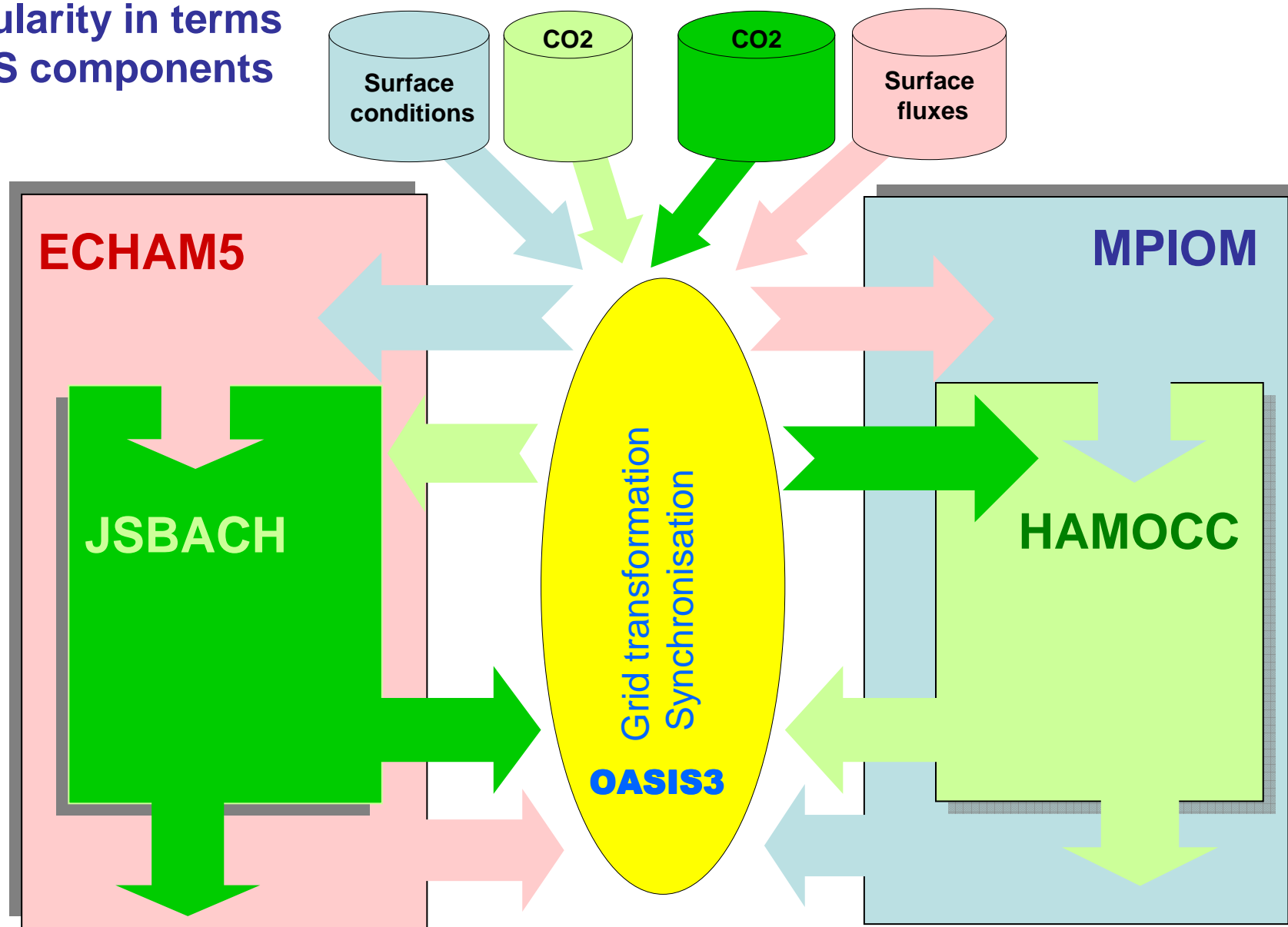
- Modularity of the IMDI toolkit
 - ◆ all tools of IMDI can be used independently
- Modularity in terms of ES components
 - ◆ plug in and out of components in coupled models
- Modularity in terms of experiment tasks
 - ◆ choose from execution, postprocessing, archiving, monitoring[, data base filling] job scripts
- Interoperability
 - ◆ all components of IMDI work together
- Common look&feel for all (coupled) models and platforms
- Integrate best practices for the target platform



SCE/SRE: Modular but interoperable



Modularity in terms of ES components



cosmos-asob



Create_COMP_cpl_models.ksh *cosmos-x*

Create_TASKS.frm *cosmos-x*

cosmos-a = **echam5**

cosmos-as = **echam5+jsbach**

cosmos-ao = **echam5 & mpiom & oasis3**

cosmos-aob = **echam5 & mpiom/hamocc & oasis3**

cosmos-asob = **echam5+jsbach & mpiom/hamocc & oasis3**

cosmos-aso = **echam5+jsbach & mpiom & oasis3**

cosmos-ob = **mpiom/hamocc**

cosmos-o = **mpiom**

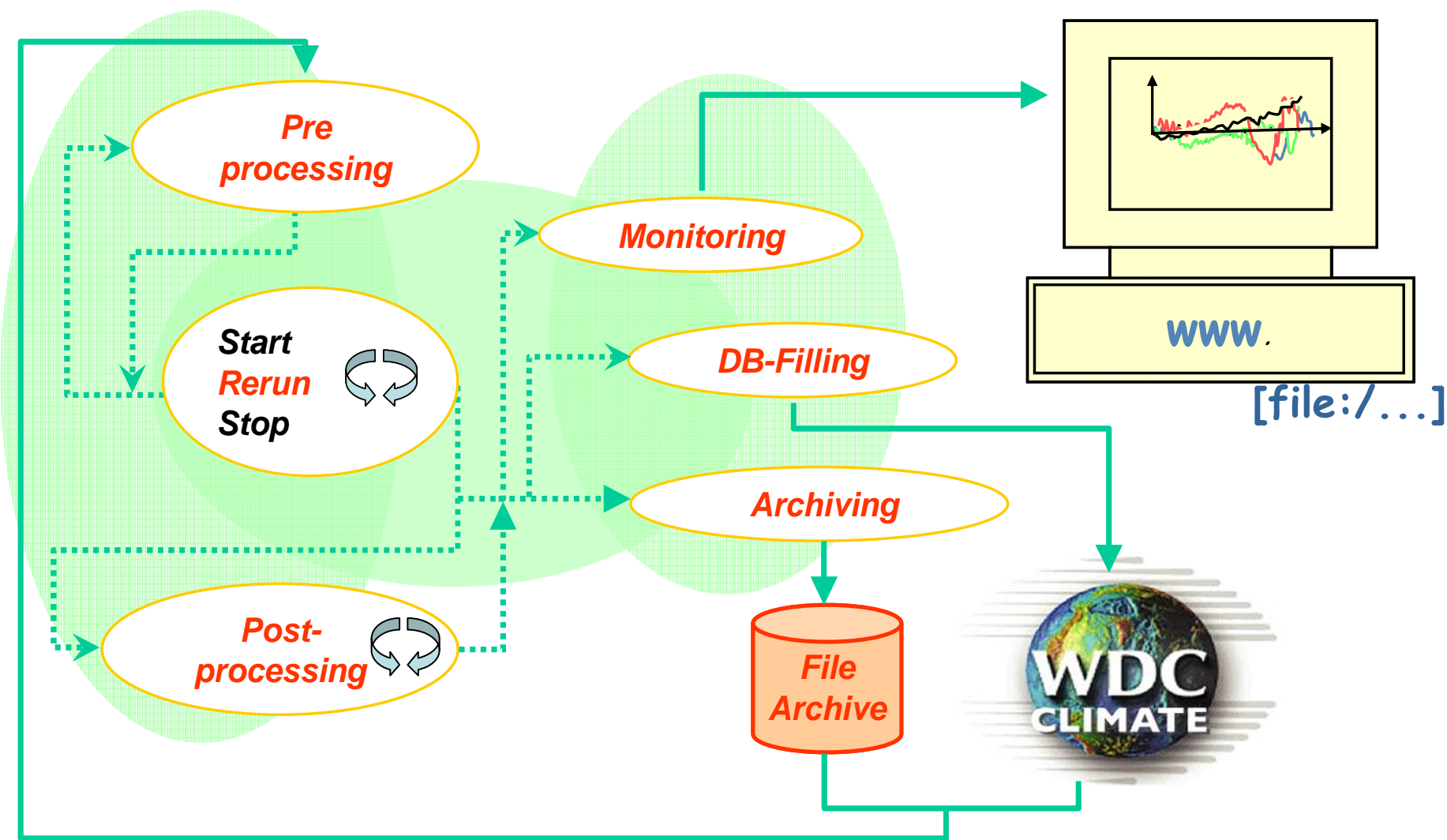
+: one code

/: one executable

&: extra executable



SRE: Modularity in terms of tasks

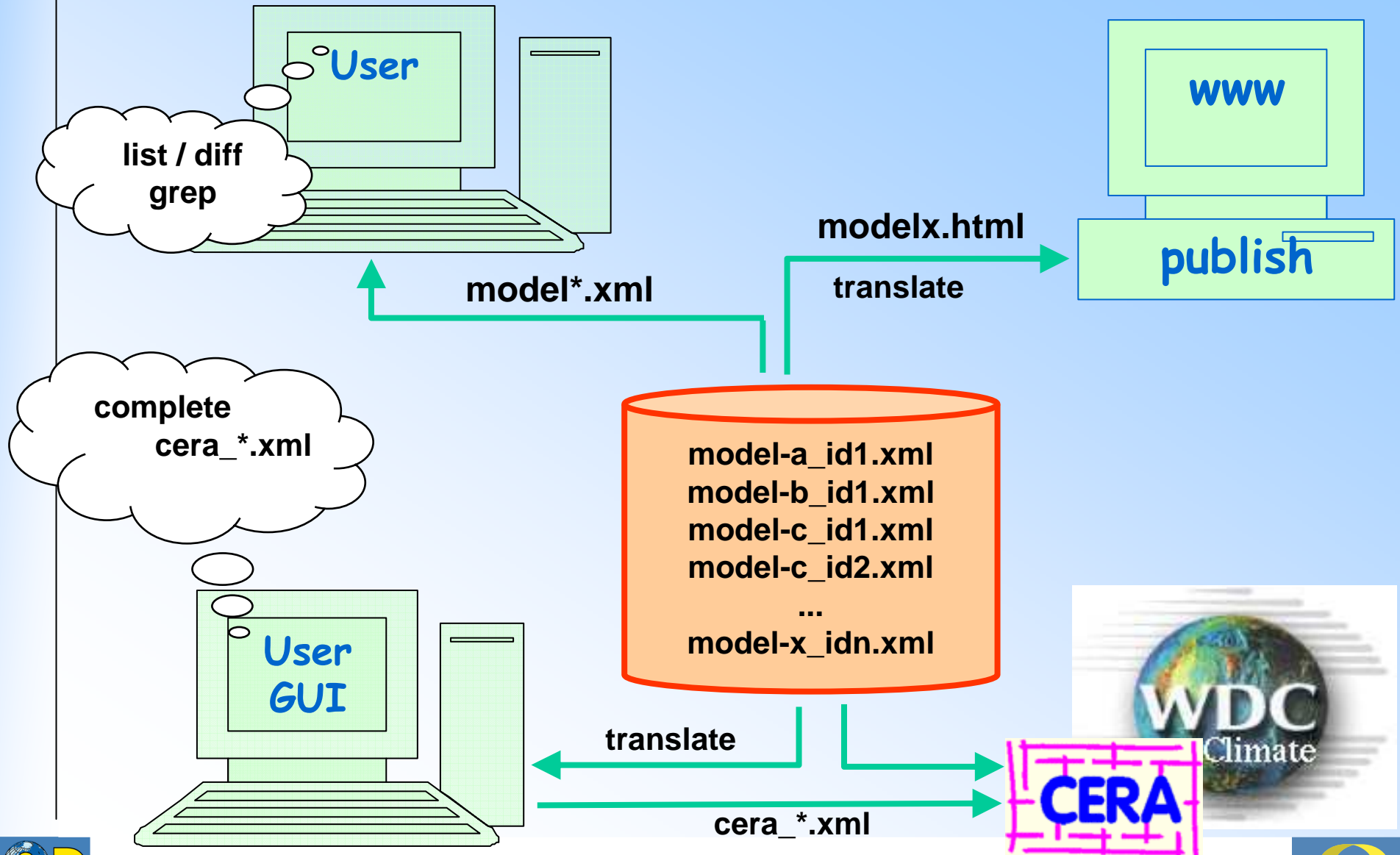




- Overview
- Integrated Data Management (IDM)
 - WDCC (tomorrow by Hans)
 - Processing and visualization (tomorrow by Jörg)
 - Meta Data by IMDI (outlook)
 - Researcher's note book (outlook)
- Source Code Management (SCM)
- Standard Compile Environment (SCE)
- Standard Running Environment (SRE)



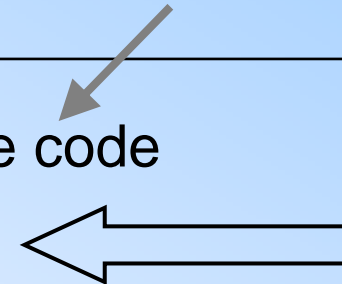
Outlook: Use of IMDI MD





- Overview
- Integrated Data Management (IDM)
- Source Code Management (SCM)
 - ◆ SVN repository
 - ◆ Scripts to analyse and transform source code
 - ◆ Source/script code directory structure
- Standard Compile Environment (SCE)
- Standard Running Environment (SRE)

Reorganize directories
Rename files
Split files





Entities:

- **the OASIS3/4 coupler**
interpolation and synchronisation
- **component models**
source code packages describing sub systems of the ES
(e.g. sea ice, ocean, clouds, ...)
 - **main models:**
components which create an executable (e.g. MPIOM)
 - **submodels:**
component (optionally) linked to a main model
(e.g. HAMOCC in MPIOM)
- **libraries**
source code package not representing subsystems of the ES



Components:

- Configurable => Can not be precompiled

OASIS3/4

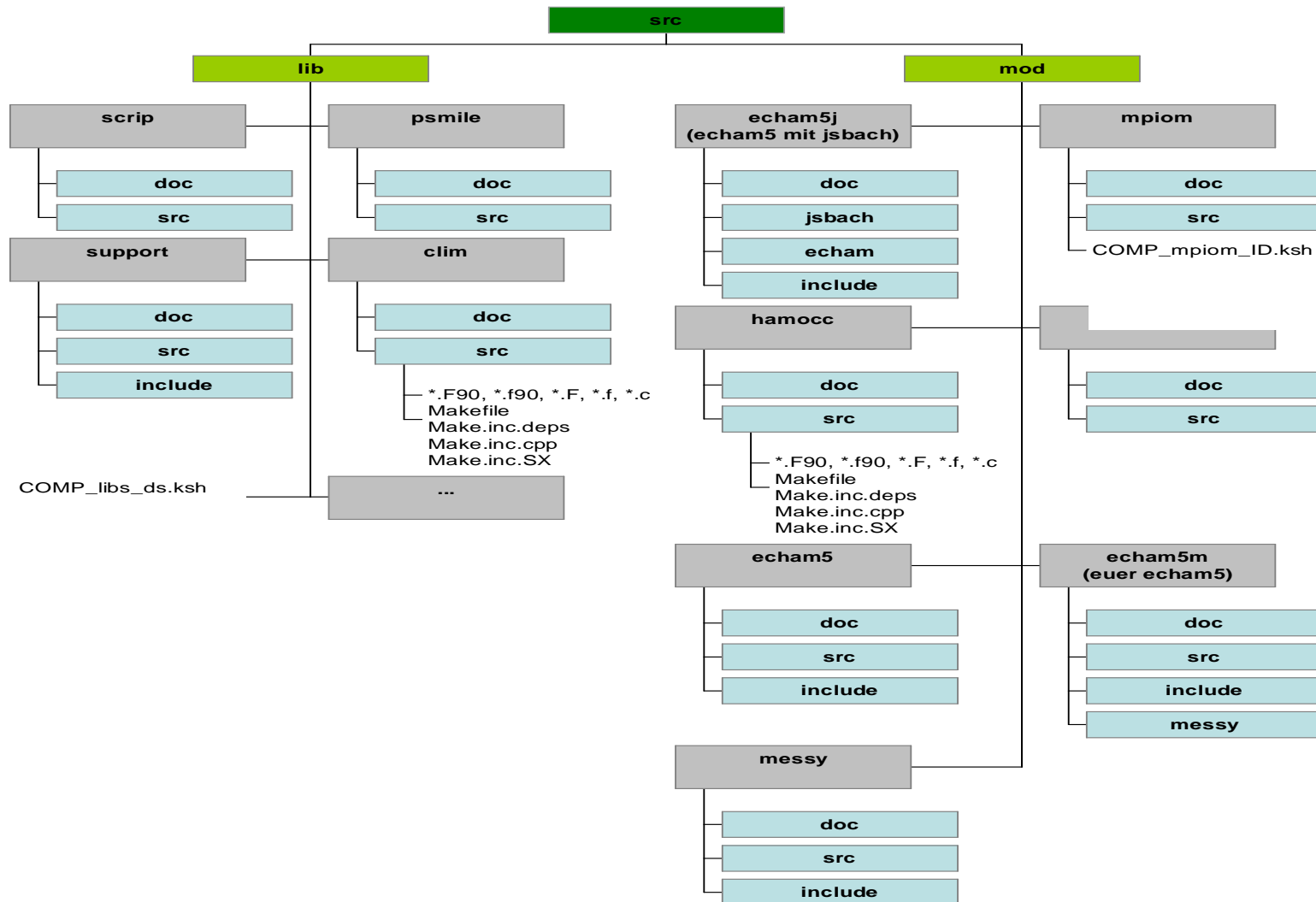
- Not configurable => Can be precompiled
- Not representing a subsystem of ES

Libraries:

- Not configurable => Can be precompiled
- Potentially usable by all components

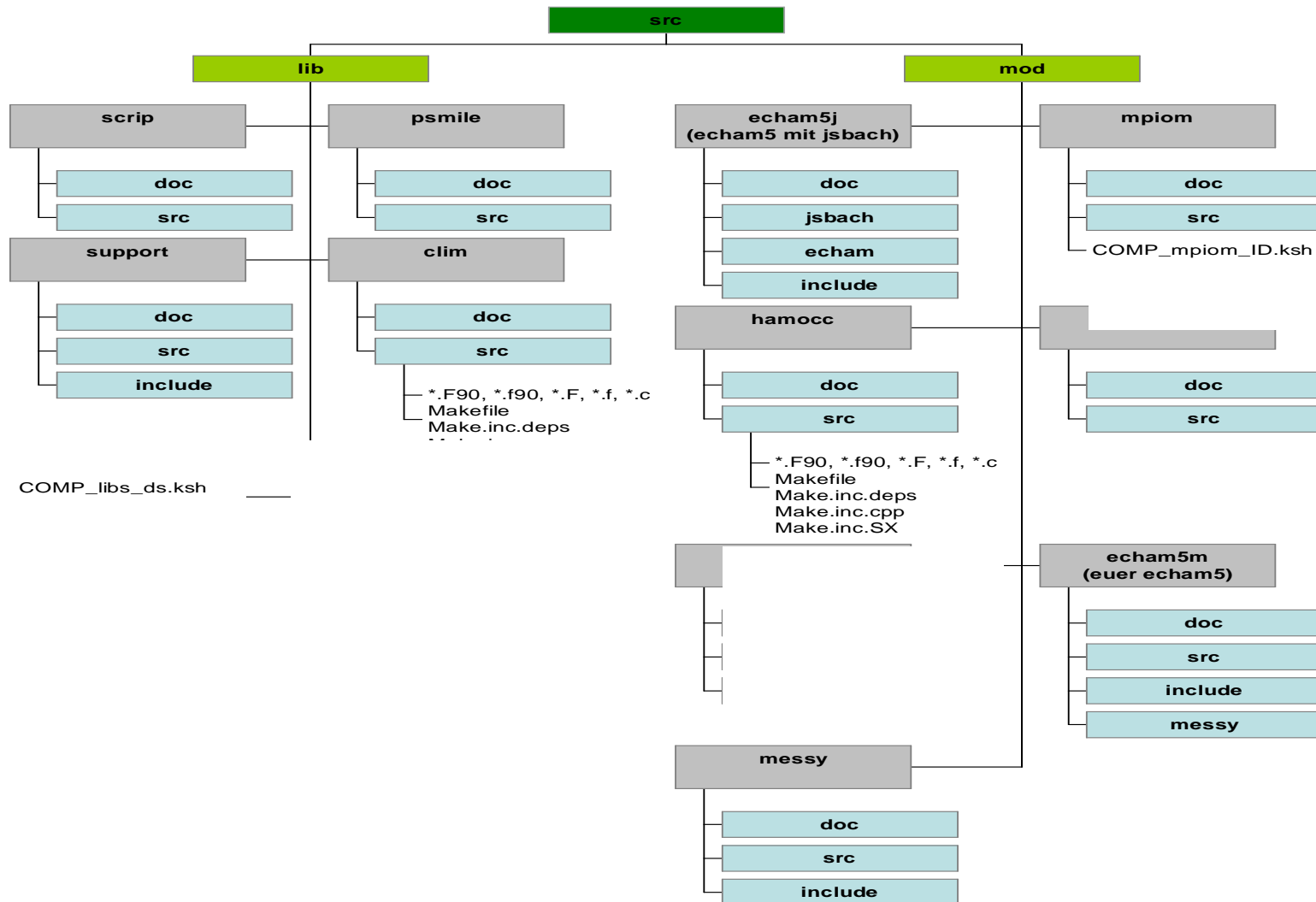


COSMOS Model source code





COSMOS Model source code





Stand-alone models

- Unix-ksh
- (g)make
- m4, perl

Coupled models

- Coupling software(<http://prism.enes.org>)
 - OASIS 3 with library clim (MPI-1/2 library)
 - Interpolation libraries (scrip, ...)
 - Model interface library psmile (MPI-1/2 library)
 - I/O library mpp_io (NetCDF library)



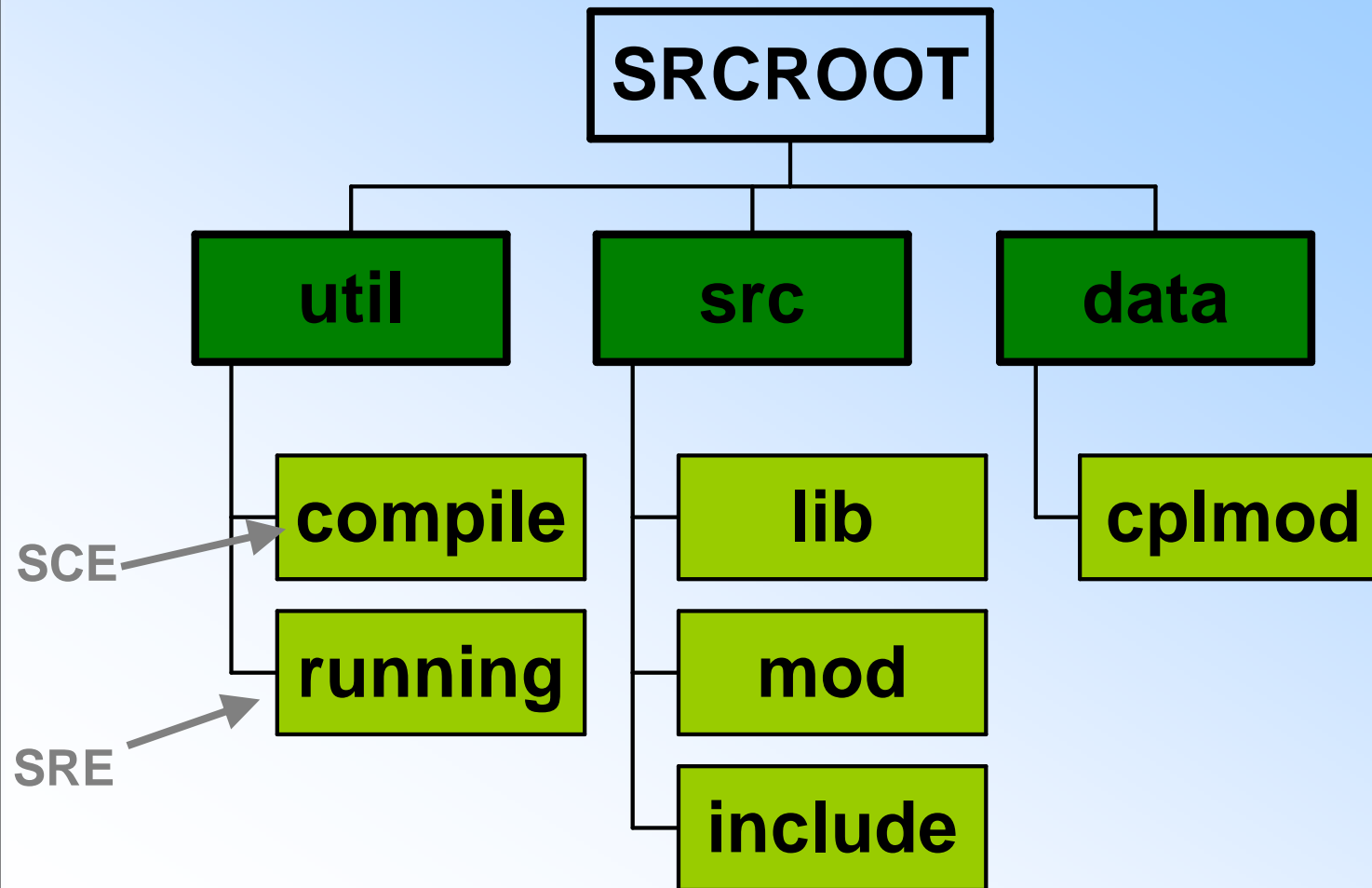
- Overview
- Integrated Data Management (IDM)
- Source Code Management (SCM)
- **Standard Compile Environment (SCE)**
 - ◆ SCE toolkit:
 - Creating compile scripts
 - Support for Makefile writing
 - Maintaining Makefiles (prerequisites)
- Standard Running Environment (SRE)

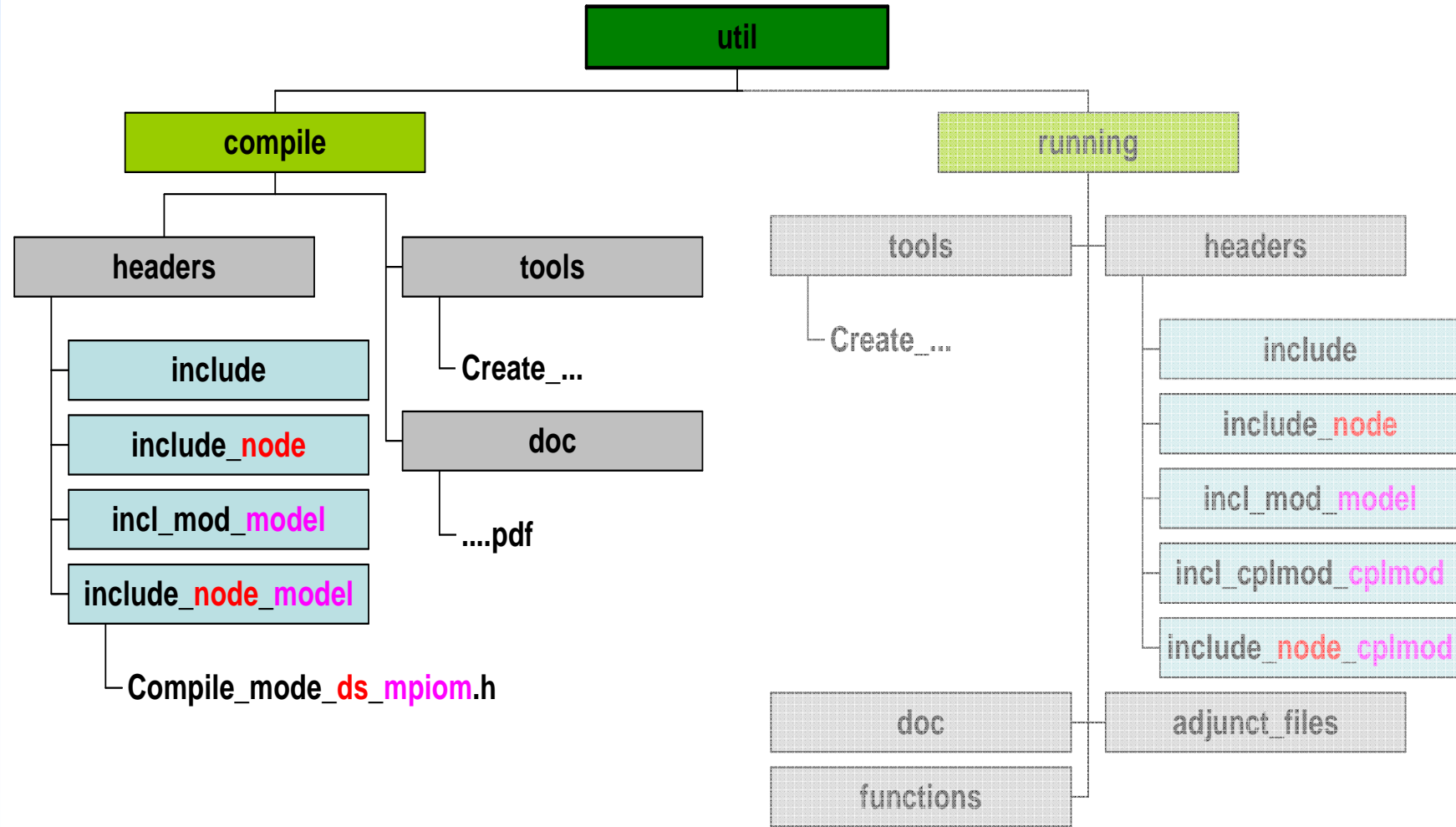


**No compile scripts are provided ...
... but tools that create them ...**

- ... for the specific platform
- ... for the specific component model or library
- ... configured for the coupled combination of models

→common look&feel







- **Create_prerequisites**
create a file containing USE or include based prerequisites
- **Create_prerequisites_cpp**
create a files needed to control recompilation if cpp flags have changed
- **Create_COMP_cpl_models.ksh**
create consistent compile scripts for a coupled model (coupler, components, libraries)
- **Create_COMP_models.frm**
create compile scripts for a component model or the coupler
- **Create_COMP_libs.frm**
create a compile script for the libraries



SCE: Use the help functions !



Create_COMP_cpl_models.ksh --help




SCE: Creating cosmos-asob

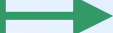





comos-asob = echam5+jsbach & mpiom/hamocc & oasis3

Create_COMP_cpl_models.ksh **cosmos-asob** \
--id= sus00?? [--stdout=- --stderr=-]



```
Create_COMP_models.frm echam5j -s=MPI1 -nds - -id=sus00?? "mpiom hamocc"  
Create_COMP_models.frm mpiom -s=MPI1 -nds - -id=sus00?? "echam5j hamocc"  
Create_COMP_models.frm hamocc -s=MPI1 -nds - -id=sus00?? "echam5j mpiom"  
Create_COMP_models.frm oasis3 -s=MPI1 -nds  
Create_COMP_libs.frm -s=MPI1 -nds
```



```
COMP_mpiom_sus00?? .ksh  
COMP_hamocc_sus00?? .ksh  
COMP_echam5j_sus00?? .ksh  
COMP_oasis3_MPI1.ksh  
COMP_libs_node.ksh
```

SX/bin/

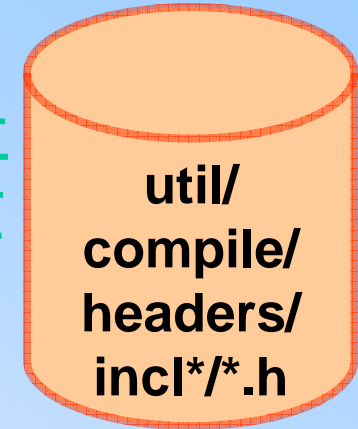
```
mpiom_hamocc_sus00?? .MPI1.x  
hamocc_sus00?? .MPI1.a  
echam5j_sus00?? .MPI1.x  
oasis3_MPI1.x  
.... , psmile.MPI1.a
```



SCE: Creating a component compile script

Create `_COMP_models.frm echm5j[-node] --id=3Ds00??`

```
...  
include incl_mod_$model/GUISpecif_$model.h  
include include_$node/Sitespecific_$node.h  
include incl_mod_$model/Cppflags_$model.h  
include include_$node_$model/Compile_mode_$model_$node.h  
include Make_model.h  
...
```



```
COMP_echm5j_sus00??.ksh
```



SCE: Creating compile scripts for ...



Linux ix86 workstations: e.g. **cosmos-o** = **mpiom**

```
Create_COMP_cpl_models.ksh cosmos-o  
    --node=linux86  
    --compiler=[sun,pgi,intel,nag,lf]  
    [--id=ID]
```

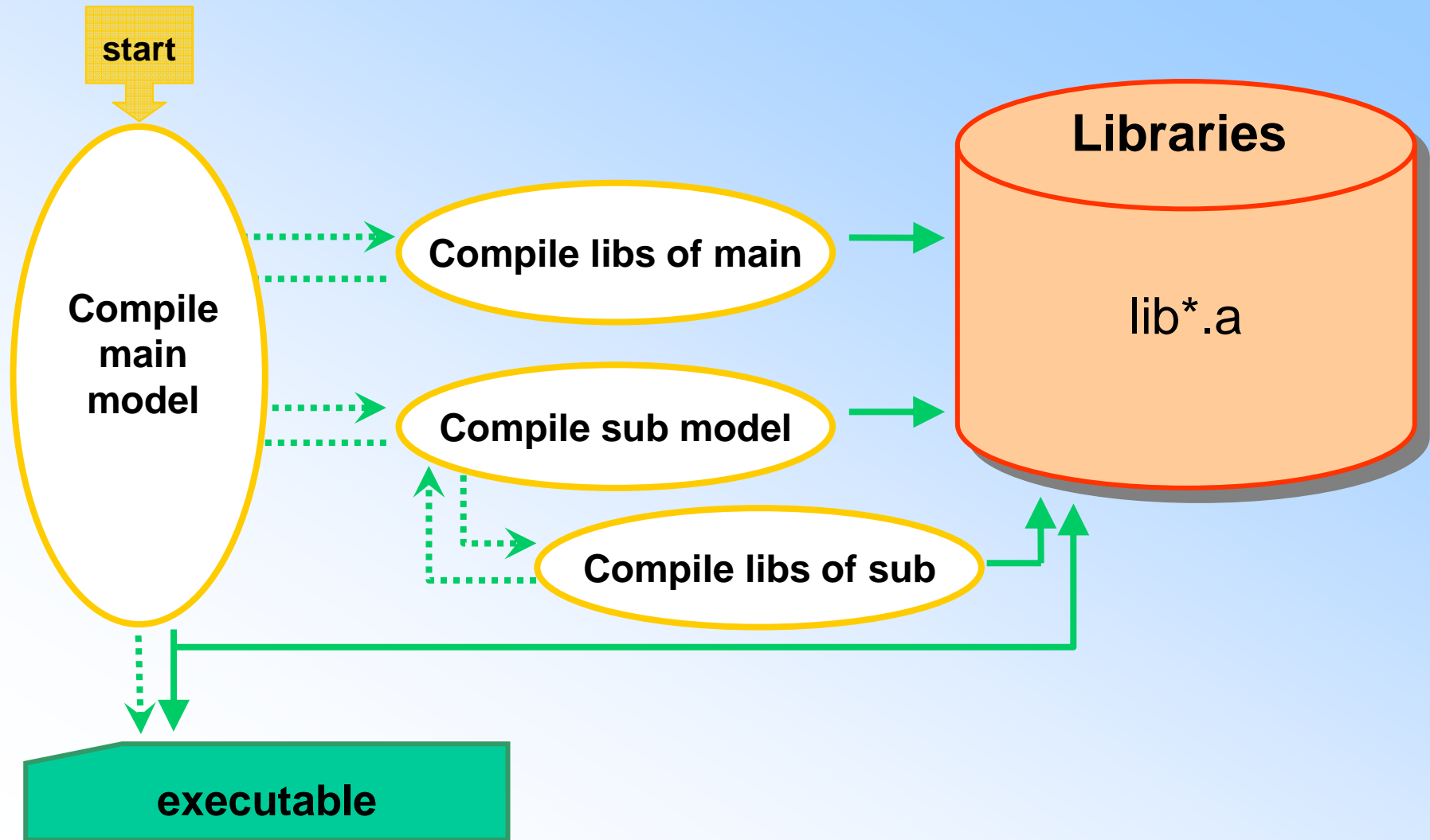
Give paths in environment parameters:

```
SCE_MPIROOT  
SCE_NETCDFROOT  
SCE_COMPILER
```

Use help function with -h -v !

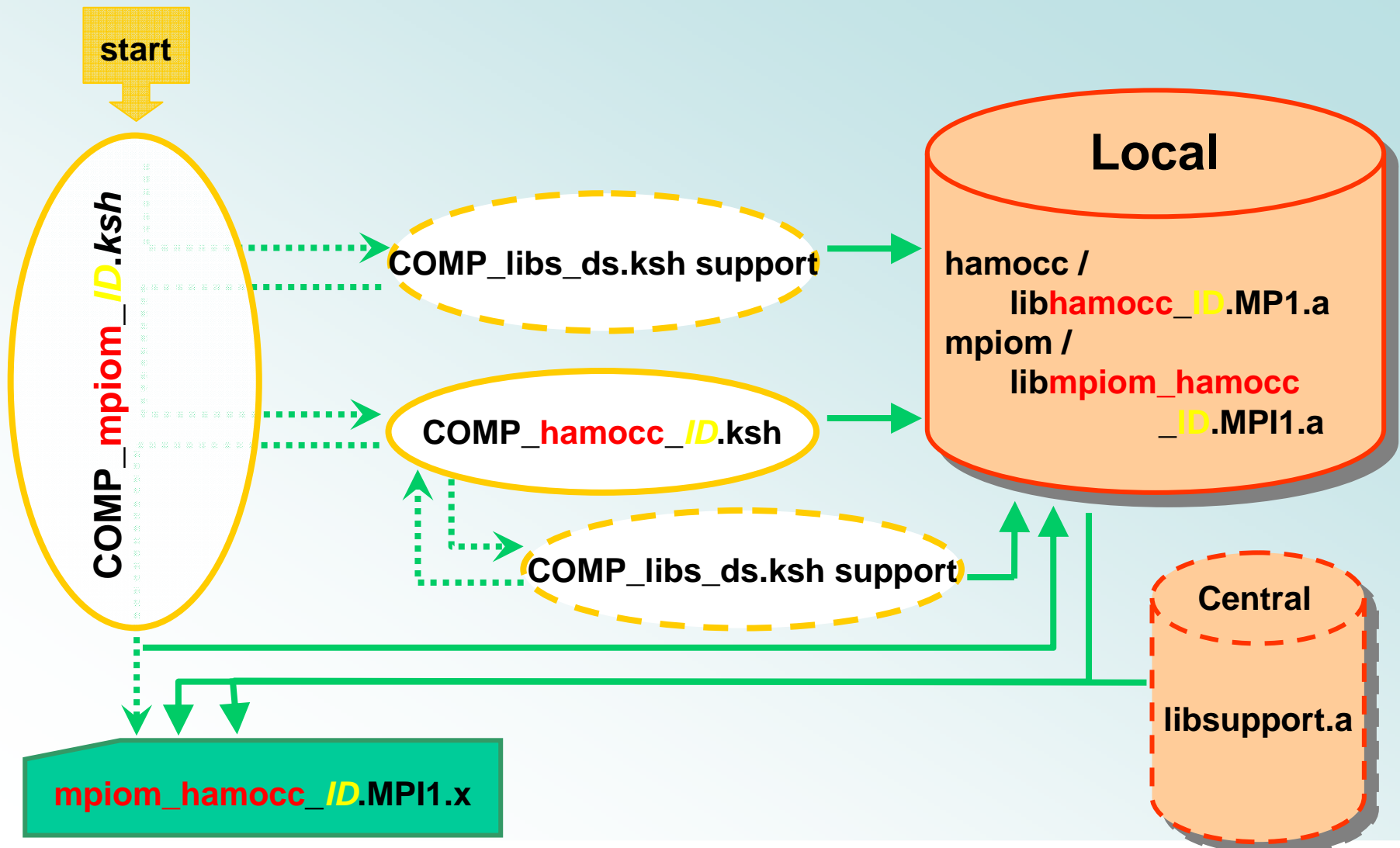


Compilation: Main+Sub Model





Compilation: Main+Sub Model with central libraries





Libraries are compiled with COMP_libs_'node'.ksh

The SCE checks all libraries required by a model
e.g. MPIOM and HAMMOCC use the 'support' library
Any local library can be declared to be 'central' in:

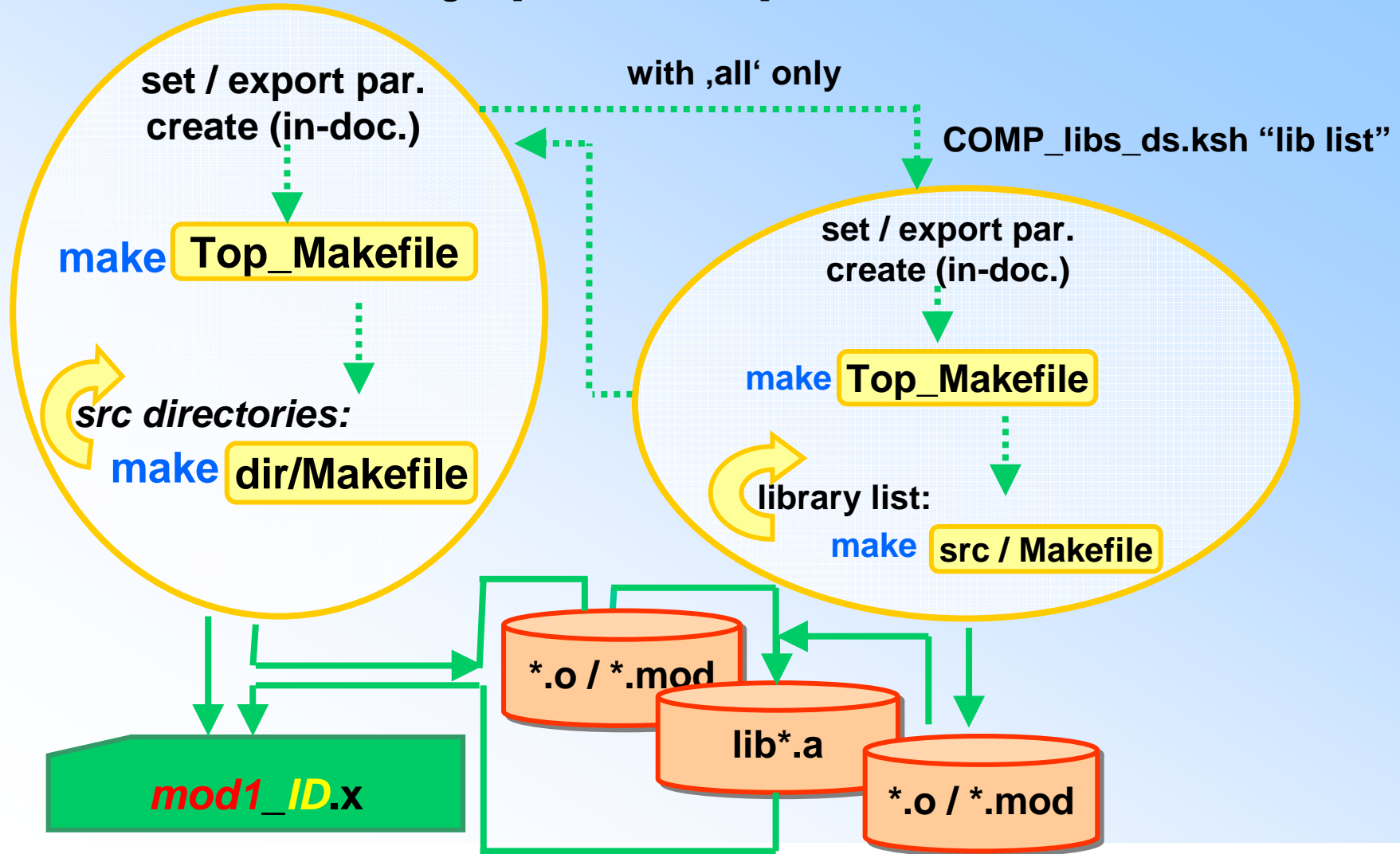
Sitespecific_'node'[_'compiler'].h

```
...  
central_libs='support'  
...
```



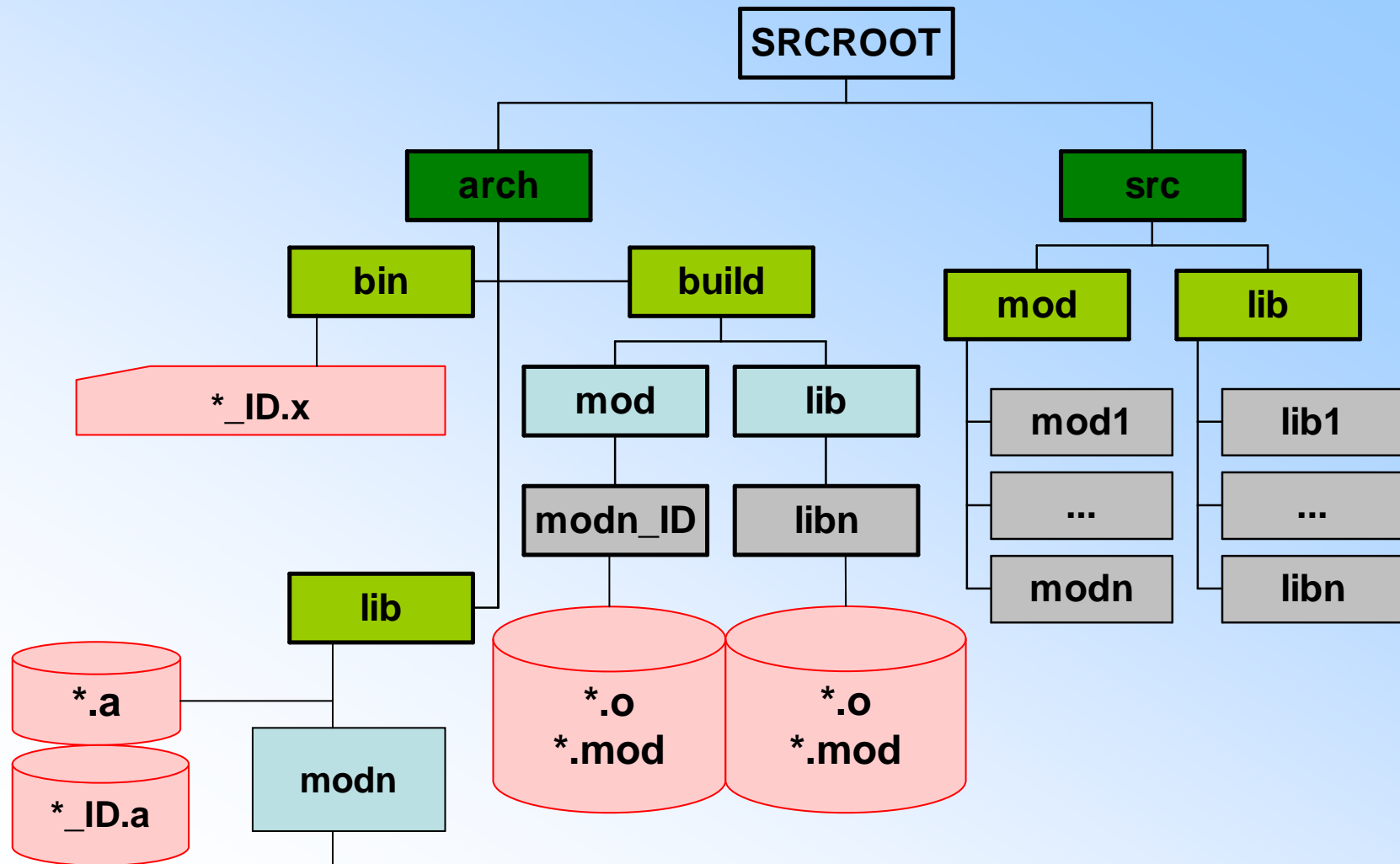
SCE: Component model compilation

COMP_mod1_ID.ksh --target=[all,lib,clean,tar]





SCE: Architecture Dependent Directories





SCE: (GNU) Make

- All compilation is based on the (g)make software:
Make 'targets' having 'prerequisites' with well defined 'rules' and avoid redundant actions
- Targets, rules & prerequisites are defined in file Makefile:

```
...  
target: prerequisite1, prerequisite2 ....  
    rules
```

```
...
```

```
...  
'what to make': 'what has to be made before'  
    'how to make'
```

```
...
```



- Executable: **OASIS3**_MPI1/2.x, **ECHAM5**_ID_MPI1.x, *.x
- Libraries: libblas.a, libsupport.a, libpsmile.MPI1/2.a, lib*.a
- Model libraries: **hamocc**_ID_MPI1.a, *_ID.a
- Binary object code: *.o
- Empty build directory
- tar file of model code



- **Executable: Libraries and model libraries**
 - echam5_ID.x: libsupport.a**
 - mpiom_ID.x: libmpiom_ID.a**
 - **Library: object binaries**
 - libsupport.a: *.o**
 - **Binary objects: F90 MODULES and include files**
 - file.o: module.mod (aliased to module.o)**
 - file.o: include.h**
 - file.o: file.F90**
- **Note:**
with multiple target lines for the same target:
the join of all prerequisite sets is valid



SCE make rules



- **\$(PROG): \$(LIBRARY)**
\$(F90) <flags> master.o echam5_ID.a support.a -o echam5_ID.x
- **\$(LIBRARY): *.o**
\$(AR) libsupport.a *.o
- **%.o: %.f90**
\$(f90) <f90flags> -c \$<
- **file.o: file.f90**
\$(f90) <special flags> -c \$<
 - **Note:**
with multiple target lines for the same target:
the last rule is valid



Example Makefile: Targets & Rules



```
clean:
    rm -f $(LIBRARY)
    ls | xargs rm -f
lib:    $(LIBRARY)
all:    $(PROG)

$(PROG): $(DEPLIBS) $(LIBRARY)
        $(F90) $(LDFLAGS) -o $@ $(MAINPRG).o $(LIBS)

$(LIBRARY): $(OBJS1) $(OBJS2) $(OBJS3) $(OBJS4) $(OBJS5)
        $(AR) $(ARFLAGS) $(LIBRARY) *.o

.SUFFIXES:
.SUFFIXES: .o .c .f .F .f90 .F90

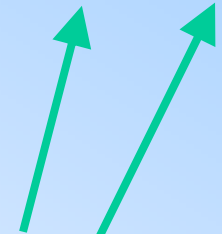
%.o: %.f90
    $(f90) $(f90FLAGS) $(INCLS) -c $<
...

%.o: %.c
    $(CC) $(CCFLAGS) $(INCLSC) -c $<

include ../../../../../../src/mod/$(MODEL_DIR)$(strip $(SRC))/Make.inc.deps
-include ../../../../../../src/mod/$(MODEL_DIR)$(strip $(SRC))/Make.inc.cpp

-include ../../../../../../src/mod/$(MODEL_DIR)$(strip $(SRC))/Make.inc.special
-include ../../../../../../src/mod/$(MODEL_DIR)$(strip $(SRC))/Make.inc.$(ARCH)
```

Create_prerequisites[_cpp]





On the web:

- **M&D homepage:**
<http://mad.zmaw.de>

- **Handbook:**
<http://mad.zmaw.de/imdi> (bottom of page):
 - - pdf-version
 - - html-version