Coupling Telemac and Sisyphe internally

Jacek A. Jankowski

Bundesanstalt für Wasserbau, Karlsruhe

17th October 2002

Reasons for internal coupling

- Telemac and Sisyphe coupled by files
  → coupling treated as an exceptional feature
  → too inefficient for practical purposes

- new developments going on in Sisyphe
  (e.g. fractional transport, multilayer bottom models)
  → add computational effort
  → require direct coupling morphodynamics/hydrodynamics

Telemac System libraries

- Problems:
  - Telemac System programs use common libraries, but...
  - each program of the System based on own large public libraries *(Fortran-77-legacy)*, never designed to be joined, e.g.
    → names of variables and subroutines in libs not unique
    → input / output defined for ONE program
- Advantages:
  - exemplarily verified software
  - user programming straightforward

Performance

For new developments:

- enhanced preformance required:
  - → no slow-down due to the coupling method
  - → parallel execution

- possibly transparent for developers and users:
  - → easy programming for developers
  - → concentration on the physical problems instead
  - → user programming not affected

What to do?

- theory and good practice say:
  - → transform f77-libraries to f90-modules
  - → define interfaces between module routines
  - → make programs to short units using modules
  - → organise user programming as *plug-ins*
- modify Telemac libraries?
  - → module (re)organisation simple
  - → interfaces more problematic
  - → short programs - already available
  - → user programming: presently everything possible

Discarded developments

❶ two serial programs coupled by network using MPI (or PVM)
→ no advantages compared to serial execution
→ parallelising by domain decomposition too complex

❷ modules out of whole Telemac and Sisyphe libraries
→ user programming unnecessary complex

❸ coupling by files
→ remains for verification purposes
→ in standalone *Sisyphe* only

Last developments: a compromise

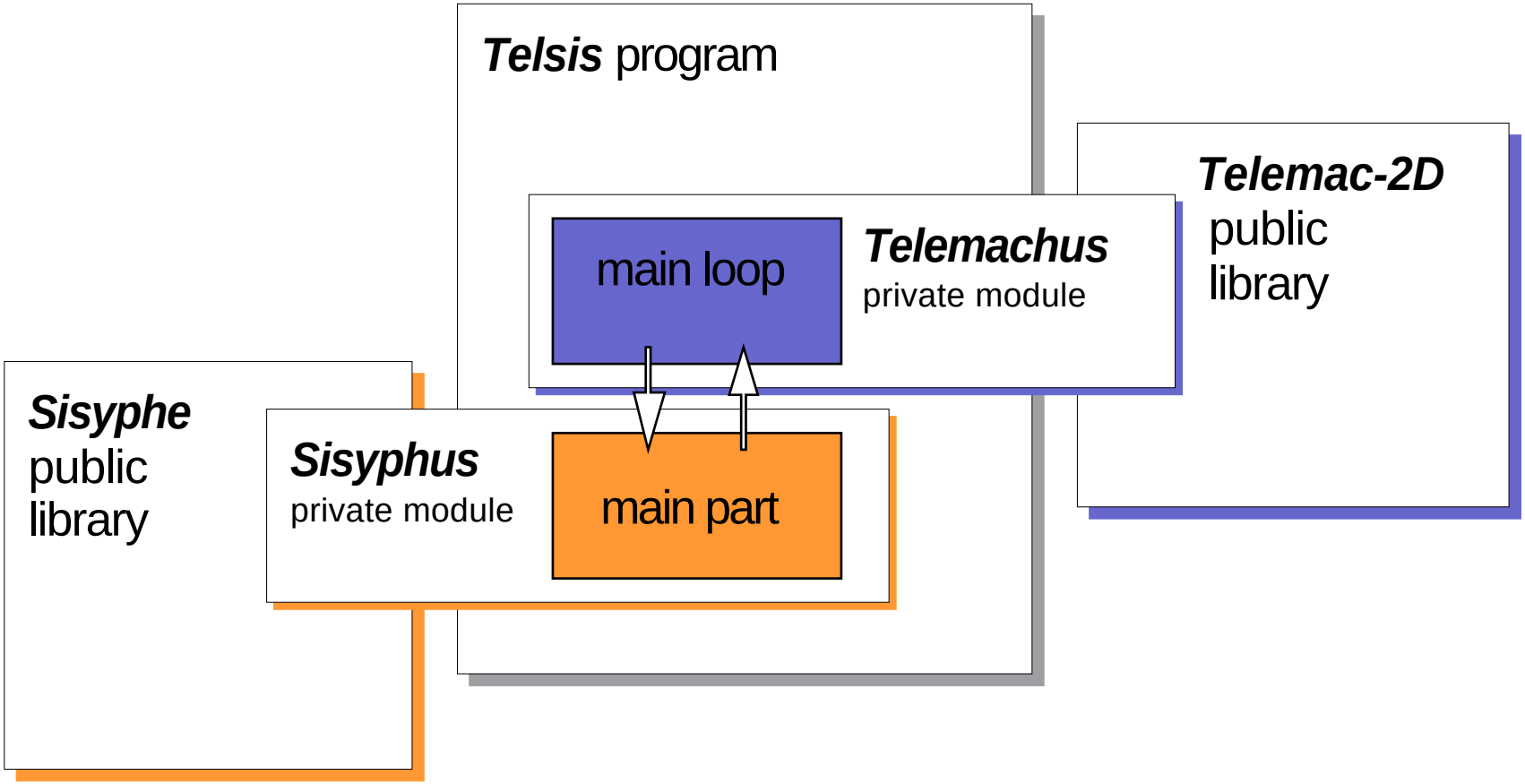
❶ modified Sisyphe public library *(Sisyphe)*
→ user programming as usual

❷ modules with adapted Telemac and Sisyphe main subroutines
→ a special solution for coupling *(Telsis)*

❸ parallel execution of the standalone and coupled programs

❹ redesigned input/output, changed perl scripts
→ nothing special, but most effort

How it works


Telsis program
main loop
Telemachus
private module
Telemac-2D
public
library
Sisyphe
public
library
Sisyphus
private module
main part

```
module TELEMACHUS
  use DECLARATIONS_SISYPHE, only : zf_sis => zf
  use SISYPHUS, only : sis
  ...
  private :: M_TEL_MASTER
  interface TEL_MASTER
    module procedure M_TEL_MASTER
  end interface
contains
  subroutine M_TEL_MASTER
    ...
    if ( ...coupling...) then
      call OV ('X=Y     ', zf%r, zf_sis%r, zf_sis%r, 0.0d0, npoin)
      call SIS(1, lt, leoprd, lisprd, nit)
    endif
    ...
  end subroutine M_TEL_MASTER
end module TELEMACHUS
```

```
module SISYPHUS
  use MOD_SISYPHE
  use DECLARATIONS_TELEMAC2D, only :  &
     &  h_tel => h, u_tel => u, v_tel => v
    ...
  double precision, public, save :: at_sis=-1.0d0
  double precision, private, save :: at0, at
    ...
  private :: M_SIS
  interface SIS
    module procedure M_SIS
  end interface
contains
  subroutine M_SIS     &
     &    (part, loopcount, grafcount, listcount, telnit)
    use BIEF
    use DECLARATIONS_TELEMAC
    use DECLARATIONS_SISYPHE
     ...
  end subroutine M_SIS
end module SISYPHUS
```

Main Telsis program

❶ initialise Telemac

❷ initialise Sisyphe

❸ call modified Telemac main subroutine

→ Telemac is the master program

→ calling appropriate parts of Sisyphe main subroutine

→ Telemac dictates the output

❹ close all files and stop

Remaining problems

- any changes in main subroutines of Telemac or Sisyphe
  → changes in Telsis modules necessary as well
- integration of *standalone* and *coupled* codes (especially Sisyphe) would increase code complexity
- *standalone* and *coupled* parameter files for Sisyphe are interpreted differently

Conclusions

- small ”modern” changes in ”older” code →
  → large functionality improvements
- present solution acceptable for most users →
  → but still unsatisfactory for programmers
- re-engineering of Telemac libraries necessary →
  → to integrate presently designed software
  → to make coupling algorithms easier