# SURFEX
# Suppression des variables globales


Gabriel Jonville, Andrea Piacentini, Isabelle d'Ast

5 juillet 2013

## Objectif : supprimer les instructions THREADPRIVATE du parallélisme OpenMP

## Etat des lieux du code :

Nombre d'instructions OMP THREADPRIVATE : 2789
Nombre de subroutines : 2255
Nombre de fichiers : 1542

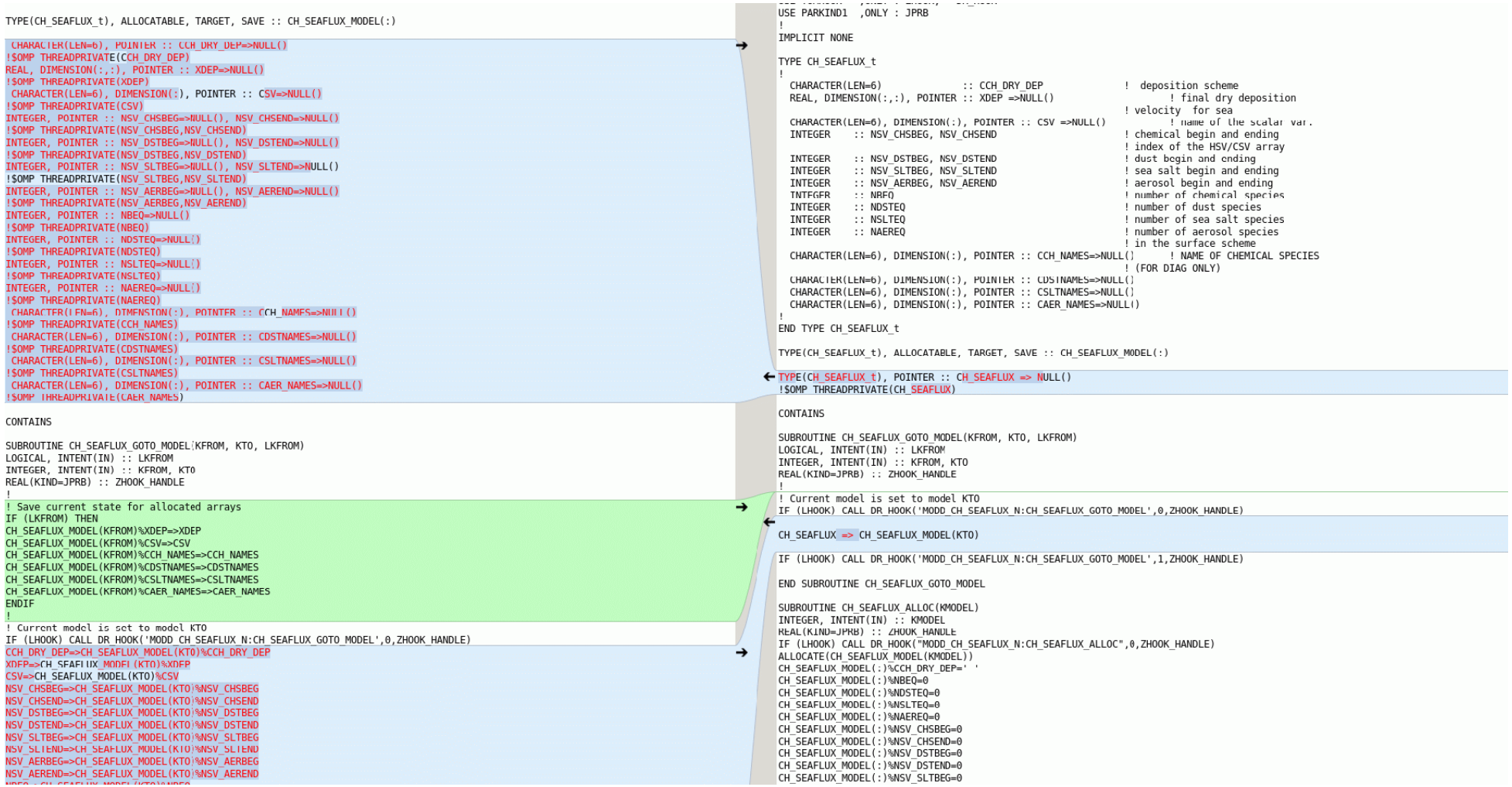
## Transformation du code se réalise en 2 phases :

Phase 1 : Pré-fixage des variables globales par le nom de la structure

Phase 2 : Passage par argument des structures

## Phase 1 : Pré-fixage des variables globales par le nom de la structure

1. Transformation intermédiaire des modules avec
   - ajout d'une variable POINTER de type TYPE défini dans le module, avec THREADPRIVATE
   - suppression des POINTER des champs de la structure et des THREADPRIVATE associés
   - simplification des subroutines _GOTO_MODEL et _ALLOC



2. Transformation des subroutines utilisant ces modules
   - sur la ligne « USE MODD_ », remplacement des variables globales par la structure
   - pré-fixage des variables globales par le nom de la structure

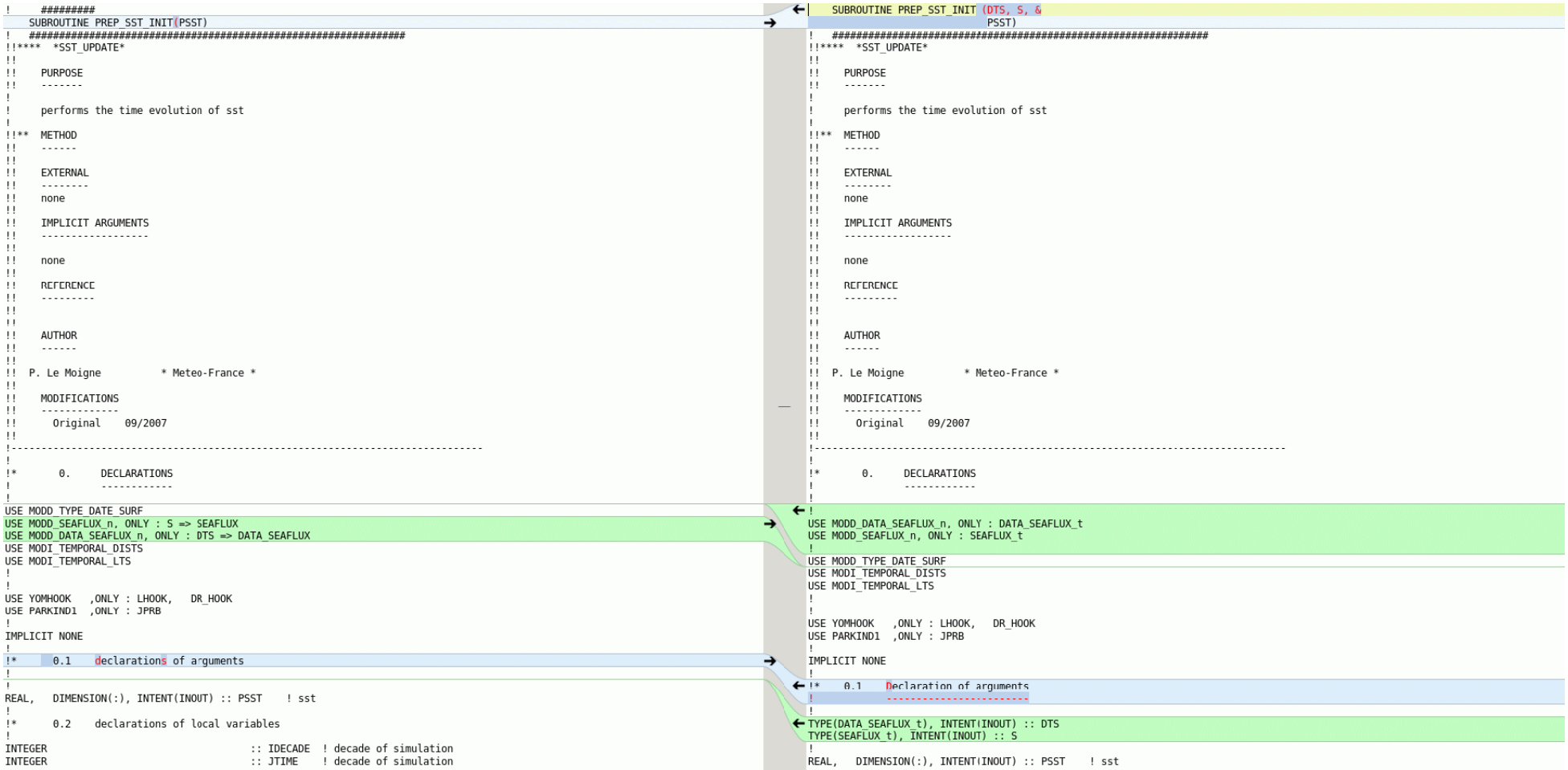
```
!                                                                  !
!*       0.     DECLARATIONS                                      !*       0.     DECLARATIONS
!              ------------                                      !              ------------
!                                                                  !
USE MODD_TYPE_DATE_SURF                                           USE MODD_TYPE_DATE_SURF
USE MODD_SEAFLUX_n, ONLY : XSST_INI, TTIME, JSX                   USE MODD_SEAFLUX_n, ONLY : S => SEAFLUX
USE MODD_DATA_SEAFLUX_n, ONLY : NTIME, XDATA_SST, TDATA_SST       USE MODD_DATA_SEAFLUX_n, ONLY : DTS => DATA_SEAFLUX
USE MODI_TEMPORAL_DISTS                                           USE MODI_TEMPORAL_DISTS
USE MODI_TEMPORAL_LTS                                             USE MODI_TEMPORAL_LTS
!                                                                 !
!                                                                 !
USE YOMHOOK   ,ONLY : LHOOK,   DR_HOOK                            USE YOMHOOK   ,ONLY : LHOOK,   DR_HOOK
USE PARKIND1  ,ONLY : JPRB                                        USE PARKIND1  ,ONLY : JPRB
!                                                                 !
IMPLICIT NONE                                                     IMPLICIT NONE
!                                                                 !
!*      0.1    declarations of arguments                          !*      0.1    declarations of arguments
!                                                                 !
!                                                                 !
REAL,   DIMENSION(:), INTENT(INOUT) :: PSST    ! SST             REAL,   DIMENSION(:), INTENT(INOUT) :: PSST    ! SST
!                                                                 !
!*      0.2    declarations of local variables                    !*      0.2    declarations of local variables
!                                                                 !
INTEGER                                 :: IDECADE  ! decade of simulation   INTEGER                                 :: IDECADE  ! decade of simulation
INTEGER                                 :: JTIME    ! decade of simulation   INTEGER                                 :: JTIME    ! decade of simulation
INTEGER, SAVE                           :: JT                                INTEGER, SAVE                           :: JT
INTEGER                                 :: JXP                               INTEGER                                 :: JXP
REAL, DIMENSION(SIZE(PSST))             :: ZSST                              REAL, DIMENSION(SIZE(PSST))             :: ZSST
REAL, SAVE                              :: ZSDTJX                            REAL, SAVE                              :: ZSDTJX
REAL                                    :: ZDT, ZALPHA                       REAL                                    :: ZDT, ZALPHA
REAL(KIND=JPRB) :: ZHOOK_HANDLE                                   REAL(KIND=JPRB) :: ZHOOK_HANDLE
!------------------------------------------------------------------!------------------------------------------------------------------
!                                                                  !
!                                                                  !
IF (LHOOK) CALL DR_HOOK('PREP_SST_INIT',0,ZHOOK_HANDLE)           IF (LHOOK) CALL DR_HOOK('PREP_SST_INIT',0,ZHOOK_HANDLE)
LOOP: DO JI = NTIME-1,1,-1                                        LOOP: DO JI = DTS%NTIME-1,1,-1
         JSX = JI                                                          S%JSX = JI
         IF (.NOT.TEMPORAL_LTS(TTIME,TDATA_SST(JSX))) EXIT LOOP            IF (.NOT.TEMPORAL_LTS(S%TTIME,DTS%TDATA_SST(S%JSX))) EXIT LOOP
      ENDDO LOOP                                                        ENDDO LOOP

IF ( TEMPORAL_LTS ( TTIME, TDATA_SST(JSX) ) ) THEN                IF ( TEMPORAL_LTS ( S%TTIME, DTS%TDATA_SST(S%JSX) ) ) THEN
   ZSST(:) = XDATA_SST(:,JSX)                                        ZSST(:) = DTS%XDATA_SST(:,S%JSX)
ELSE IF ( .NOT. TEMPORAL_LTS ( TTIME, TDATA_SST(NTIME) ) ) THEN   ELSE IF ( .NOT. TEMPORAL_LTS ( S%TTIME, DTS%TDATA_SST(DTS%NTIME) ) ) THEN
  ZSST(:) = XDATA_SST(:,NTIME)                                      ZSST(:) = DTS%XDATA_SST(:,DTS%NTIME)
ELSE                                                              ELSE

   CALL TEMPORAL_DISTS ( TDATA_SST(JSX+1)%TDATE%YEAR,TDATA_SST(JSX+1)%TDATE%MONTH,   &      CALL TEMPORAL_DISTS ( DTS%TDATA_SST(S%JSX+1)%TDATE%YEAR,DTS%TDATA_SST(S%JSX+1)%TDATE%MONTH,   &
                         TDATA_SST(JSX+1)%TDATE%DAY ,TDATA_SST(JSX+1)%TIME,         &                                DTS%TDATA_SST(S%JSX+1)%TDATE%DAY ,DTS%TDATA_SST(S%JSX+1)%TIME,         &
                         TDATA_SST(JSX)%TDATE%YEAR,TDATA_SST(JSX)%TDATE%MONTH,      &                                DTS%TDATA_SST(S%JSX)%TDATE%YEAR,DTS%TDATA_SST(S%JSX)%TDATE%MONTH,      &
                         TDATA_SST(JSX)%TDATE%DAY ,TDATA_SST(JSX)%TIME,             &                                DTS%TDATA_SST(S%JSX)%TDATE%DAY ,DTS%TDATA_SST(S%JSX)%TIME,             &
                         ZSDTJX                                                     )                                ZSDTJX                                                     )

   CALL TEMPORAL_DISTS ( TTIME%TDATE%YEAR   ,TTIME%TDATE%MONTH,                 &      CALL TEMPORAL_DISTS ( S%TTIME%TDATE%YEAR   ,S%TTIME%TDATE%MONTH,                 &
                         TTIME%TDATE%DAY    ,TTIME%TIME,                        &                            S%TTIME%TDATE%DAY    ,S%TTIME%TIME,                        &
                         TDATA_SST(JSX)%TDATE%YEAR,TDATA_SST(JSX)%TDATE%MONTH,  &                            DTS%TDATA_SST(S%JSX)%TDATE%YEAR,DTS%TDATA_SST(S%JSX)%TDATE%MONTH,  &
                         TDATA_SST(JSX)%TDATE%DAY ,TDATA_SST(JSX)%TIME,         &                            DTS%TDATA_SST(S%JSX)%TDATE%DAY ,DTS%TDATA_SST(S%JSX)%TIME,         &
                         ZDT                                                    )                            ZDT                                                    )
!                                                                 !
    ZALPHA = ZDT / ZSDTJX                                             ZALPHA = ZDT / ZSDTJX
!                                                                 !
    ZSST(:)= XDATA_SST(:,JSX)+(XDATA_SST(:,JSX+1)-XDATA_SST(:,JSX))*ZALPHA    ZSST(:)= DTS%XDATA_SST(:,S%JSX)+(DTS%XDATA_SST(:,S%JSX+1)-DTS%XDATA_SST(:,S%JSX))*ZALPHA
```

Résultats de cette première phase :

- le nombre d'instructions OMP THREADPRIVATE restantes est de 473 (sur 2789)
- le nombre de fichiers impactés est de 598 (sur 1542)

## Phase 2 : Passage par argument des structures

1. Transformation des subroutines utilisant ces modules
   - mise en argument des structures
   - sur la ligne « USE MODD_ », remplacement de la structure par le nom du TYPE
   - déclaration des structures



2. Transformation des modules
   - suppression du pointer de structure introduit précédemment
   - suppression de la subroutine _GOTO_MODEL

**Modules particuliers avec THREADPRIVATE**

- les modules MODD_PACK_* : absence de TYPE => à introduire et traiter comme les autres

- les modules MODD_IO_SURF_*
  MODD_MASK => à traiter à la fin
  MODD_SURFEX_MPI et _OMP

- les « nouveaux » modules (version 7.3) définissant plusieurs TYPE :
  MODD_BEM_n
  MODD_DIAG_MISC_TEB_n
  MODD_TEB_GARDEN_n => splités en 2 ou 4 pour définir un seul TYPE par module
  MODD_TEB_GREENROOF_n
  MODD_TEB_n

  mais il apparait dans ces modules des tableaux 2D de TYPE :
  TYPE(BEM_t), ALLOCATABLE, TARGET, SAVE :: BEM_MODEL(:,:)
  avec
  SUBROUTINE BEM_GOTO_MODEL(KFROM, KTO, LKFROM, KFROM_PATCH, KTO_PATCH)