



### Comment instrumenter les codes

- de manière pérenne (une fois pour toute et pas à chaque sortie de version),
- en minimisant l'impact dans le code source,
- en respectant les évolutions des gros codes (possibilité d'utilisation directe des updates, nouveaux modèles, corrections de bugs etc),
- sans interagir avec leur exécution en standalone
- en ne conservant qu'une seule version du code,
- en respectant les évolutions des bibliothèques de couplage.

## Comment instrumenter les codes : exemple d'AVBP



AVBP\_V6.X

Sources du code  
AVBP gérées sous git



OpenPALM

Sources du coupleur  
OpenPALM gérées sous  
CVS



AEROTH\_APP

Sources de l'application  
AVBP/AVTP/PRISSMA  
gérées sous git

**Aucune référence à des  
instructions de couplage et au  
coupleur dans AVBP**

## Comment instrumenter les codes : exemple d'AVBP



AVBP\_V6.X



OpenPALM



AEROTH\_APP

Sources du code  
AVBP gérées sous git

⇒ avbp.exe

⇒ libs\_avbp.a

- interf\_mpi\_init.F ⇒ CALL MPI\_Init
- interf\_init.F ⇒ Vide
- interf\_generate\_CPL.F ⇒ Vide
- interf\_generate\_CPL3D.F ⇒ Vide
- interf\_send\_mesh\_CPL.F ⇒ Vide
- interf\_exchange.F ⇒ Vide
- interf\_end.F ⇒ Vide
- interf\_mpi\_end.F ⇒ CALL MPI\_Finalize



## Aerothermal application AVBP – AVTP - PRISSMA

Comment instrumenter les codes : exemple d'AVBP



AVBP\_V6.X



OpenPALM



AEROTH\_APP

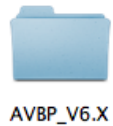
Sources du coupleur  
OpenPALM gérées sous  
CVS

⇒ palmlib.a

⇒ cwipi\_lib.a

⇒ ...

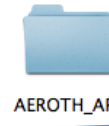
## Comment instrumenter les codes : exemple d'AVBP



AVBP\_V6.X



OpenPALM



AEROTH\_APP

Sources de l'application  
AVBP/AVTP/PRISSMA  
gérées sous git

UNITS/

⇒ AVBP/

⇒ AVTP/

⇒ PRISSMA/

APPLICATION/

⇒ RUN\_AVBP/ → **Répertoires de calcul**

⇒ RUN\_AVTP/ **!! Même répertoire CPL ou pas !!**

⇒ RUN\_PRISSMA/

⇒ palm\_main

⇒ main\_avtp, main\_avbp, main\_prissma

⇒ avbp\_avtp\_prissma.pil

⇒ **coupling.choices**

## Comment instrumenter les codes : exemple d'AVBP

