

# Gestion du code source & Application pour Morph-M

Centre de Morphologie Mathématique  
École des Mines de Paris

Atelier Morph-M



## 1 Motivation

Contexte d'utilisation

Exemple pratique

La morale de l'exemple

## 2 La gestion des sources

Généralités

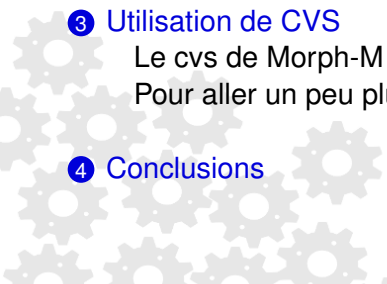
Présentation de CVS

## 3 Utilisation de CVS

Le cvs de Morph-M

Pour aller un peu plus loin...

## 4 Conclusions



## 1 Motivation

Contexte d'utilisation

Exemple pratique

La morale de l'exemple

## 2 La gestion des sources

Généralités

Présentation de CVS

## 3 Utilisation de CVS

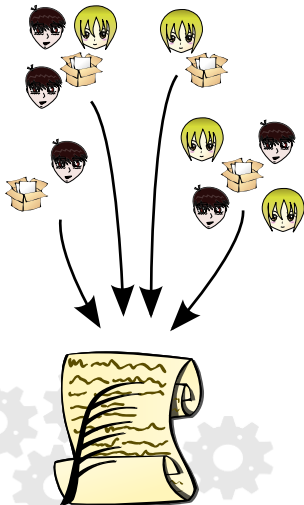
Le cvs de Morph-M

Pour aller un peu plus loin...

## 4 Conclusions



# Développement de projets avec Morph-M



- Nombreux développeurs
- Nombreux projets
- Code partagé
- Corrections de bugs
- Ajouts de fonctionnalités

## Un exemple *ordinaire*

### Programme de Thomas

```
void tFonct(imIn, imOut)
{ ...
  invert(imA, imB);
  ... }
```

### Programme de Jean

```
void jFonct(imIn, imOut)
{ ...
  invert(imC, imD);
  ... }
```

→ Thomas trouve un bug et son origine !

### Fichier commun "arith.cpp"

```
void invert(im1, im2) {
  for ipix, opix
    in im1, im2 {
      opix = 255-ipix;
    }}
}}
```

## Un exemple *ordinaire*

### Programme de Thomas

```
void tFonct(imIn, imOut)
{ ...
  invert(imA, imB);
  ... }
```

### Programme de Jean

```
void jFonct(imIn, imOut)
{ ...
  invert(imC, imD);
  ... }
```

→ Thomas trouve un bug et son origine !

### Fichier commun "arith.cpp"

```
void invert(im1, im2) {
  for ipix, opix
    in im1, im2 {
      opix = 255-ipix;
    }
}
```

## Un exemple *ordinaire*

### Programme de Thomas

```
void tFonct(imIn, imOut)
{ ...
  invert(imA, imB);
  ... }
```

### Programme de Jean

```
void jFonct(imIn, imOut)
{ ...
  invert(imC, imD);
  ... }
```

→ Thomas trouve un bug et son origine !

### Fichier commun "arith.cpp"

```
void invert(im1, im2) {
  for ipix, opix
    in im1, im2 {
      opix = 255-ipix;
    }}
}}
```

## Un exemple ordinaire (suite)

“arith.cpp” chez Thomas

```
void invert(im1,im2){  
    for ipix,opix  
        in im1,im2 {  
        opix = max(im1)-ipix;  
    }}  
}}
```

“arith.cpp” chez Jean

```
void invert(im1,im2){  
    for ipix,opix  
        in im1,im2 {  
        opix = 255-ipix;  
    }}  
}}
```

→ Le bug est corrigé chez Thomas !



## Un exemple ordinaire (suite)

“arith.cpp” chez Thomas

```
void invert(im1,im2){  
    for ipix,opix  
        in im1,im2 {  
        opix = max(im1)-ipix;  
    }}
```

“arith.cpp” chez Jean

```
void invert(im1,im2){  
    for ipix,opix  
        in im1,im2 {  
        opix = 255-ipix;  
    }}
```

→ Le bug est corrigé chez Thomas !

## Un exemple ordinaire (ça se complique)

### “arith.cpp” chez Thomas

```
void invert(im1,im2){
  for ipix,opix
    in im1,im2 {
      opix = max(im1)-ipix;
    }
}
```

### “arith.cpp” chez Jean

```
void invert(im1,im2){
  for ipix,opix
    in im1,im2 {
      opix = 255-ipix;
    }
}
```

### “arith.cpp” chez Ali :

```
void invert(im1,im2){
  switch(im1.type){
    case UINT8:
      invert_u8(im1,im2);break;
    case UINT32:
      invert_u32(im1,im2);break; }...}
```

Les codes d'Ali et Thomas sont corrects mais peuvent donner des résultats différents !

## Situations à éviter

La correction d'un algorithme **n'est pas propagée** vers tous les utilisateurs.

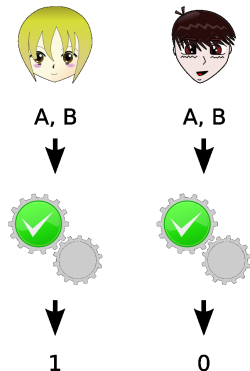


Non !



# Situations à éviter

Des implémentations **correctes** donnent  
des résultats **différents**.



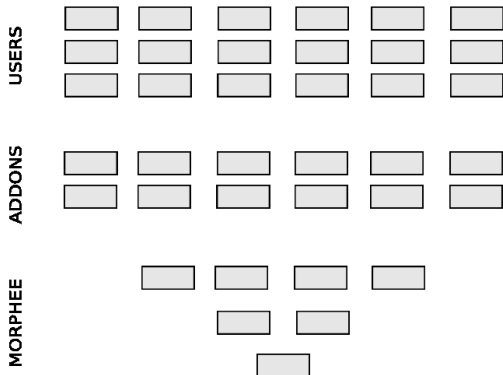
Non !

## Situations à éviter

Il faut prendre en compte **plusieurs centaines de milliers** de lignes de code **dupliquées** sur les machines d'une **quinzaine** de développeurs.

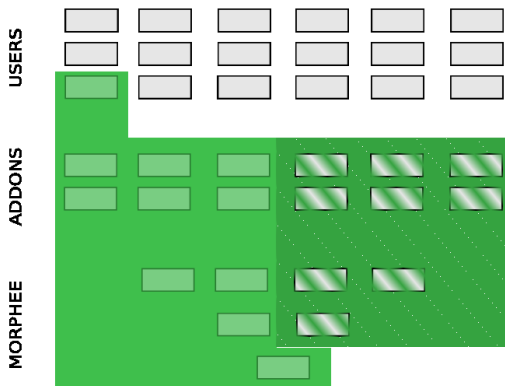


# Le cas de Morph-M



Box : 10000 < lignes de code < 30000

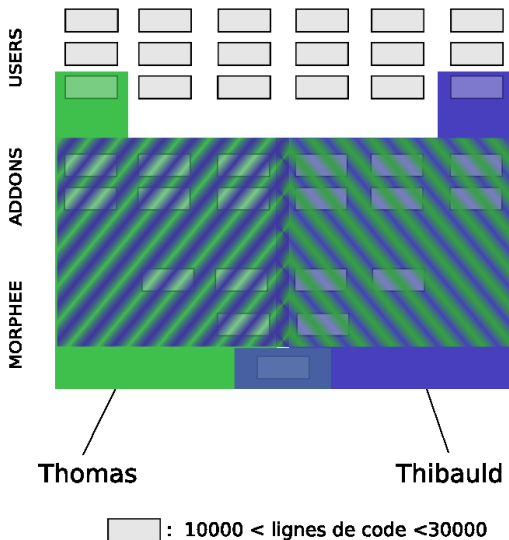
# Le cas de Morph-M



Thomas

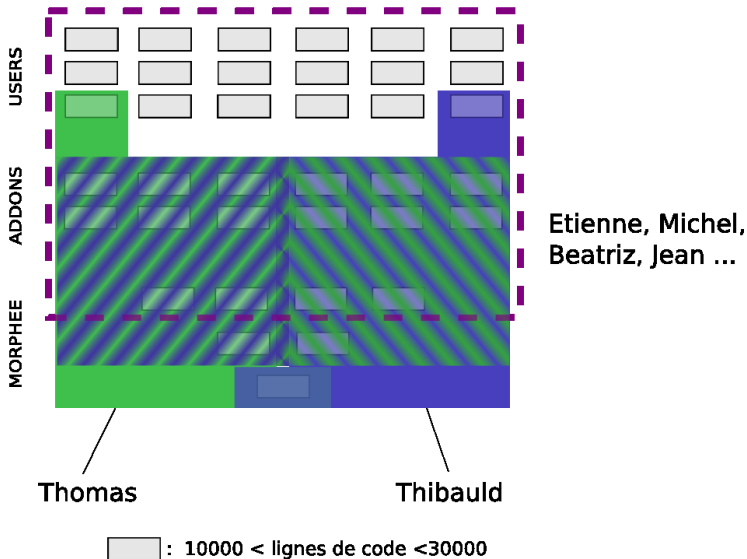
□ : 10000 < lignes de code < 30000

# Le cas de Morph-M

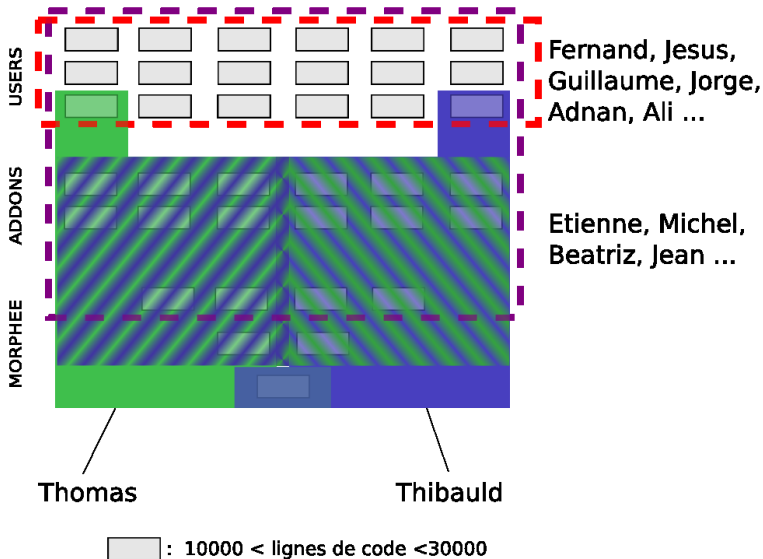




# Le cas de Morph-M



# Le cas de Morph-M



## ① Motivation

Contexte d'utilisation

Exemple pratique

La morale de l'exemple

## ② La gestion des sources

Généralités

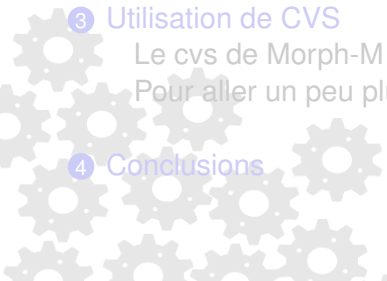
Présentation de CVS

## ③ Utilisation de CVS

Le cvs de Morph-M

Pour aller un peu plus loin...

## ④ Conclusions



# Les possibilités offertes

- Plusieurs développeurs pour un même projet.
- Historique des changements.
- Changements réversibles.
- Versionnement



# Un peu de vocabulaire

**SCM** Software Configuration Management

... Source Code Management

**VCS** Version Control System



## Quelques programmes connus

**CVS** Concurrent Versions System

**Subversion** le “remplaçant” de CVS

**Bzr** “bazar” pour une gestion distribuée

**Git** gestionnaire utilisé pour le noyau Linux



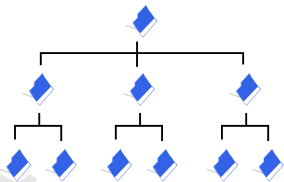
# Une gestion centralisée



Les sources sont sauvegardées dans un **dépôt central** (“repository”).



## Une organisation rigide



Les sources sont classées par projet dans une **hiérarchie de dossiers**.





# Un suivi des changements

Chaque changement est **enregistré** auprès du dépôt.

```
revision 1.7.2.9
```

```
date: 2004-01-16 15:20:17; author: lerallut;
```

```
* Plus de verbosité dans les operateurs CVariant
```

```
* Definition explicite des DataTraits::dc et sdt.
```

```
C'est sale, mais apparemment c'est ce qu'il faut  
faire... A retravailler.
```

```
-----
```



## ① Motivation

Contexte d'utilisation

Exemple pratique

La morale de l'exemple

## ② La gestion des sources

Généralités

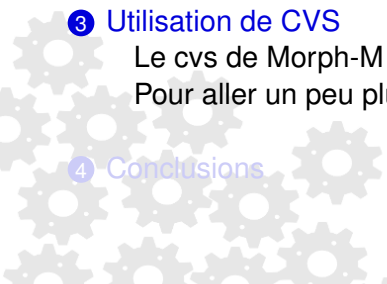
Présentation de CVS

## ③ Utilisation de CVS

Le cvs de Morph-M

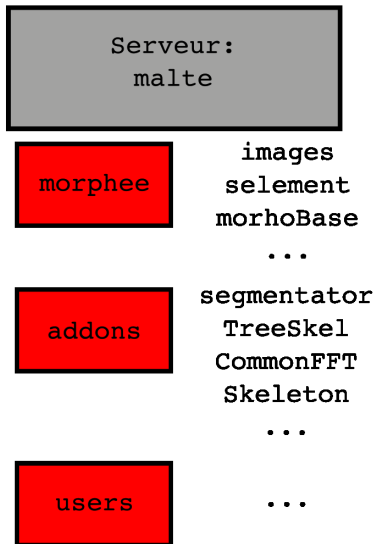
Pour aller un peu plus loin...

## ④ Conclusions



# Le dépôt de Morph-M

- serveur *Malte*
- dossier  
*/home/morphee-admin*
- une structure nette en projets, modules et sous modules.



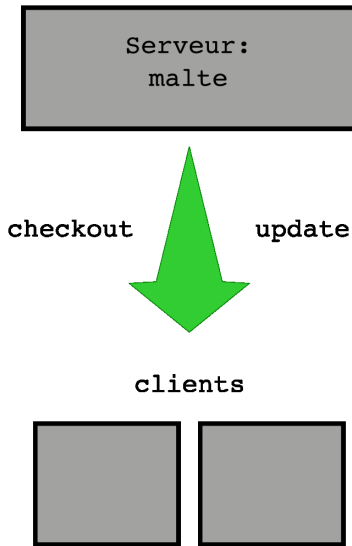
## Récupérer les sources et les mise à jour

### cv<sup>s</sup> checkout

La première fois que l'on récupère les sources.

### cv<sup>s</sup> update

À chaque fois que l'on veut récupérer la version de référence.



## Faire des modifications locales

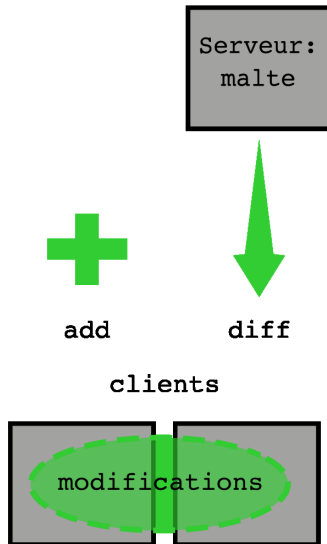
### cv`s` add

Ajouter un fichier ou un répertoire.

Modifier les fichiers locaux.

### cv`s` diff

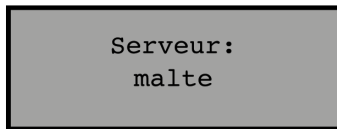
Comparer les versions locale et officielle des fichiers.



### cv<sub>s</sub> commit

Enregistrer les changements sur le dépôt.

Tant qu'un changement n'est pas commité, il faut le considérer comme inexistant !



commit

clients



## Les conflits

Lorsque la même ligne d'un même fichier est éditée à deux endroits différents.

```
<<<<<<<<<
code local
=====
code de reference
>>>>>>>>>
```

Les conflits se résolvent localement **avant** le commit du fichier fusionné.

## La commande “diff”

Affiche la liste des différences entre deux versions d'un fichier.

```
Index: commonTypes.hpp
```

```
=====
```

```
RCS file:
```

```
/home/morphee-admin/cvsroot/morphee/morphee\  
/common/include/commonTypes.hpp,v
```

```
retrieving revision 1.94
```

```
retrieving revision 1.93
```

```
diff -r1.94 -r1.93
```

```
182,183c182,183
```

```
<
```

```
    sdtMap,  //!< map-type
```

```
<    sdtPair  //!< pair type
```

```
---
```

```
>    sdtMap  //!< map-type
```

```
>    //sdtPair  //!< pair type
```



## 1 Motivation

Contexte d'utilisation

Exemple pratique

La morale de l'exemple

## 2 La gestion des sources

Généralités

Présentation de CVS

## 3 Utilisation de CVS

Le cvs de Morph-M

Pour aller un peu plus loin...

## 4 Conclusions



## Conclusion

- Mutualisation du développement
- Garde une trace de toutes les modifications
- Un code “à jour” à tout instant

### Pour Morph-M

- Le CVS est déjà en place.
- Plusieurs projets liés à Morph-M sont déjà gérés.
- Tous les projets Morph-M devraient être sur son dépôt.

# Merci !

Et n'hésitez pas à vous rendre sur le portail de Morph-M :

`http://rhodes.ensmp.fr/morphee`

Ressources graphiques : OpenClipart.org

