Programmation « orientée système »

Approfondissements semaine 11

Jean-Cédric Chappelier

Laboratoire d'Intelligence Artificielle
Faculté I&C

Rappels des points clés

compilation
édition des liens

- **toutes** les étapes de la « compilation »
- principes de fonctionnement de l'éditeur de liens
- principes de base des `Makefile` (cible, dépendances, commande)

(re-)rappels semaine 10 aussi :

- les *deux* prototypes de `main()`
- savoir utiliser `argc` et `argv`
- ce que sont les macros (*uniquement* de la *ré-écriture*)
- protéger les arguments des macros (parenthèses)
- éviter d'utiliser deux fois un argument dans une macro (effets de bord)
- compilation conditionnelle

Jean-Cédric Chappelier
EPFL

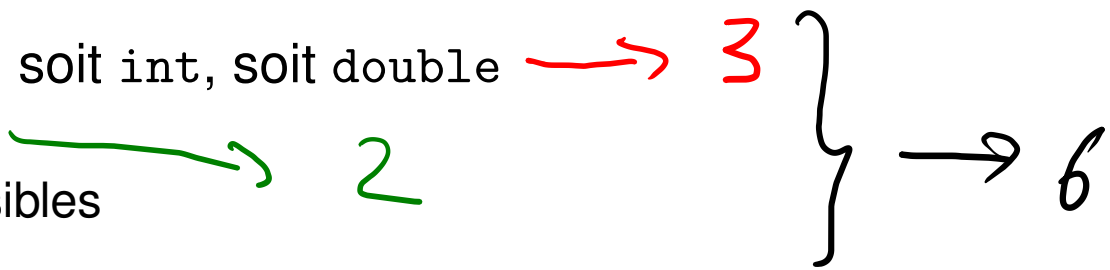
Etudes de cas

- reprendre en détail l'un ou l'autre exemple du cours ?
- compilation séparée et compilation conditionnelle :

  (un peu artificiel – cf extension ci-dessous)

  un « dump mémoire » soit `char`, soit `int`, soit `double`
  soit en français, soit en anglais

  ☞ six exécutables différents possibles



  Version plus réaliste (exercice : faites le) :

  - 1 seul exécutable avec arguments de la ligne de commande
    (plutôt que de multiples exécutables différents)

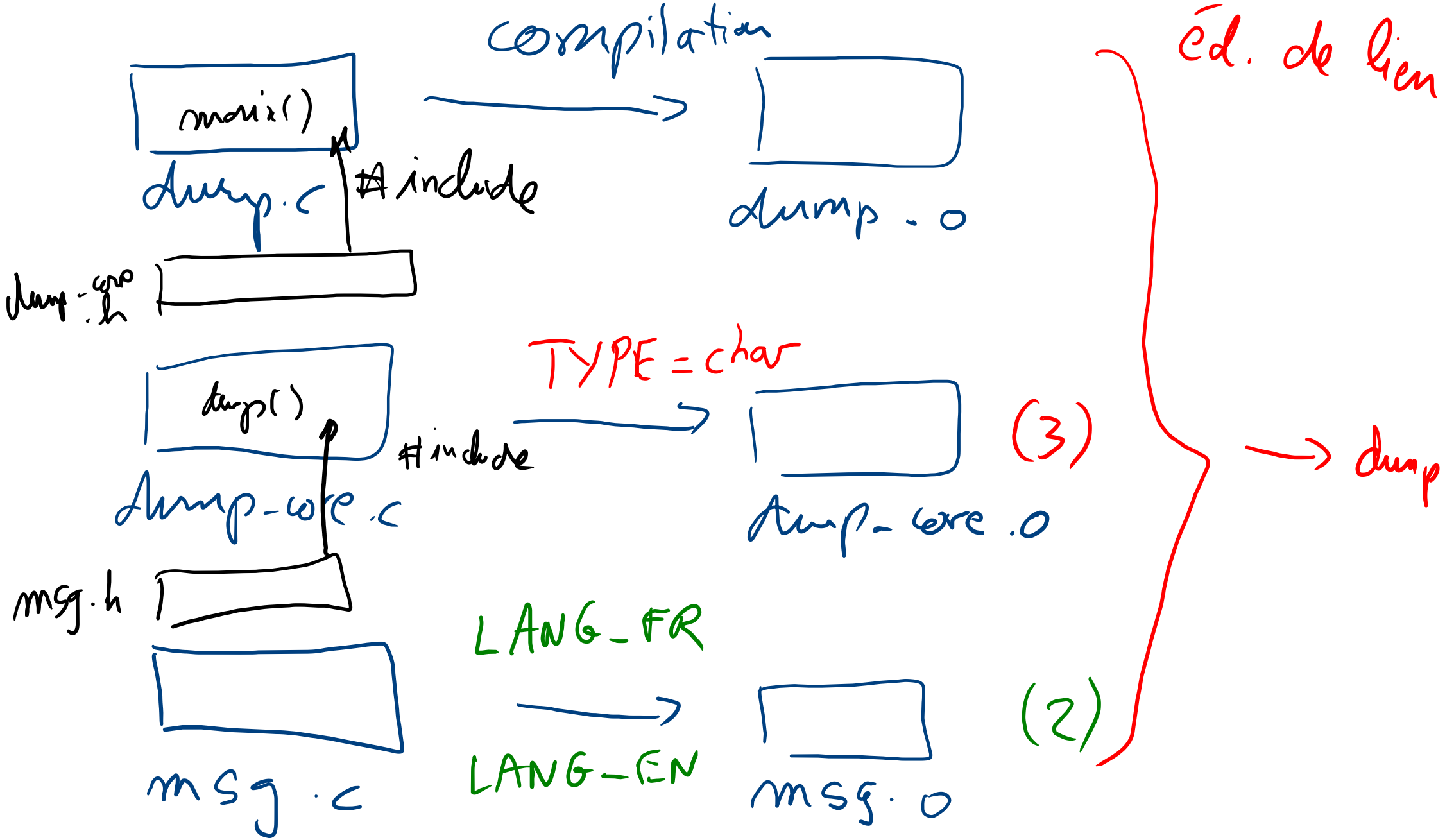
Structure du code

- `dump.c` : le `main()`, qui va simplement appeler une fonction `dump()` sur un tableau
  ce code va être utiliser pour générer : `dump_char_fr`, `dump_int_fr`, `dump_double_fr`, `dump_char_en`, ...

- `dump_core.c` : fourni la fonction
  ```
  void dump(const void* mem, size_t total_size)
  ```
  laquelle appelle `heading()`
  puis « dump » le contenu mémoire indiqué par la macro `TYPE`
  (☞ Attention nécessité d'une double macro ici)

- `messages.c` : fourni la fonction
  ```
  void heading(const char* type, const void* mem);
  ```
  qui affiche p.ex. (via la macro `LANG_EN` ou `LANG_FR`) :
  `memory content starting from 0x7fff932d5f20 dumped as char:`
  ou
  `le contenu de la mémoire à l'adresse 0x7ffc6d6dd4a0 vu comme des double :`

EPFL


compilation
éd. de lien
main()
dump.c
#include
dump.o
dump-core.h
dump()
TYPE = char
#include
(3)
dump-core.c
dump-core.o
msg.h
LANG_FR
LANG_EN
(2)
msg.c
msg.o
dump