Compilation

Introduction

Adrien Guatto

Master 1 Informatique
2022–2023

Bienvenue en Master 1 !

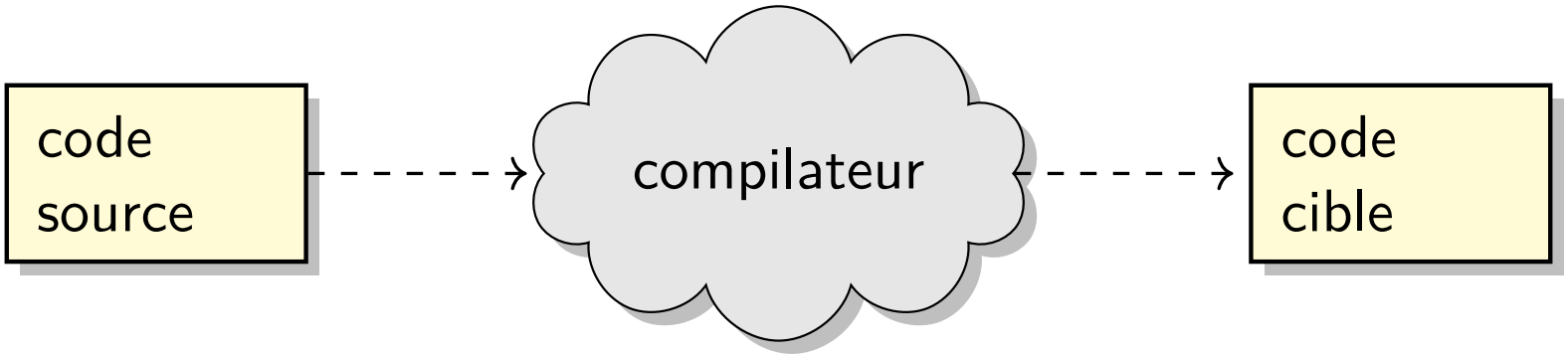
En licence, on apprend à :

- programmer dans plusieurs langages,
- reconnaître et employer les bons algorithmes,
- utiliser efficacement son système (e.g., les outils UNIX).

En master, on ouvre les boîtes noires…

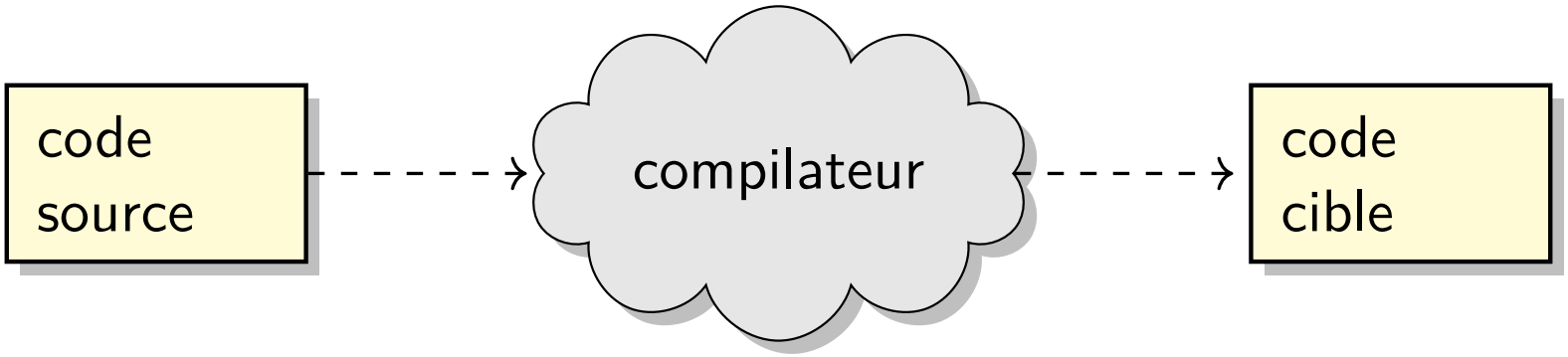
- Comment fonctionne un processeur ?
  - Cf. *architecture des ordinateurs* (S1).
- Comment conçoit-on un langage de programmation ?
  - Cf. *compilation* (S1), *sémantique des langages de prog.* (S2).

Le module Compilation, objectif premier



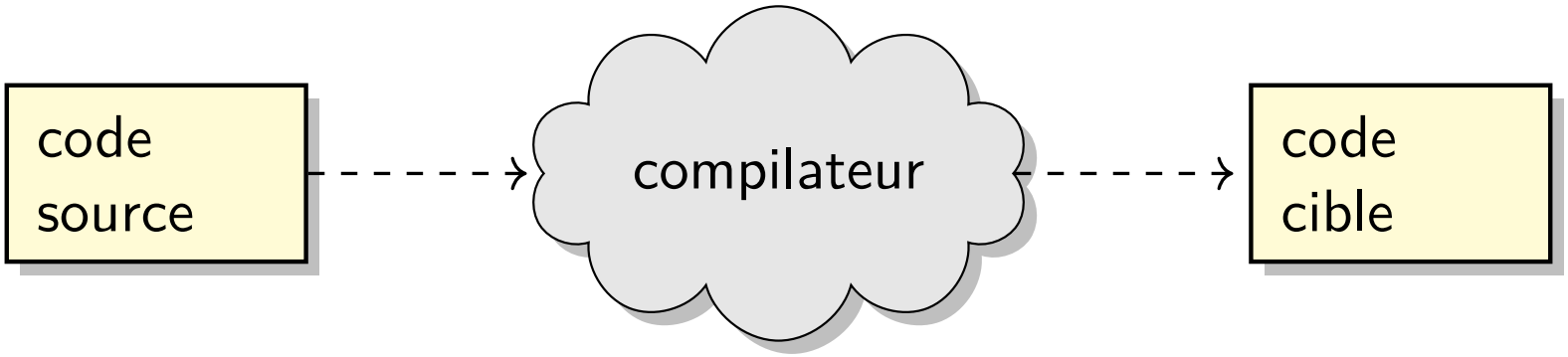
- Entrée : du code *source*, écrit dans un certain langage.

Le module Compilation, objectif premier



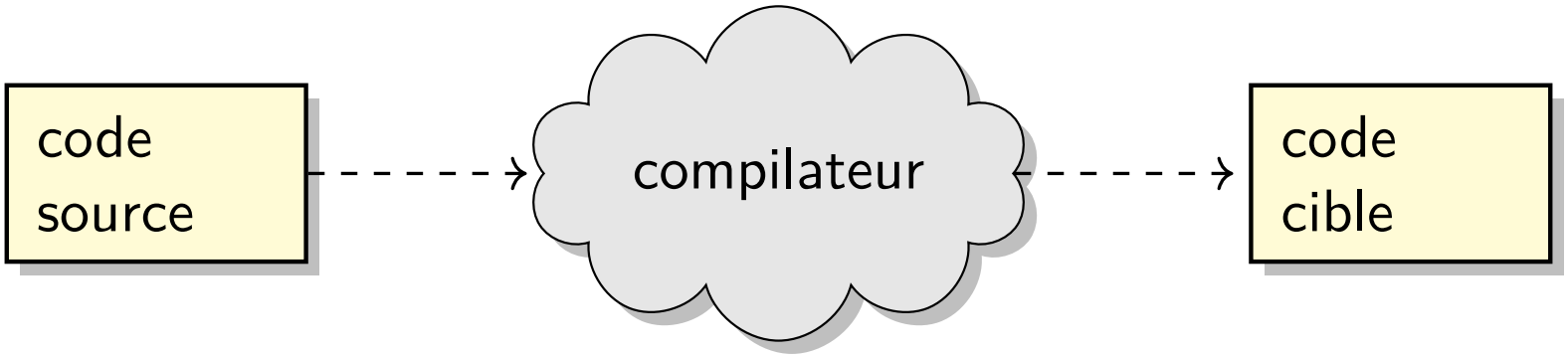
- Entrée : du code *source*, écrit dans un certain langage.
  - Traditionnellement : une suite de caractères dans un fichier.

Le module Compilation, objectif premier



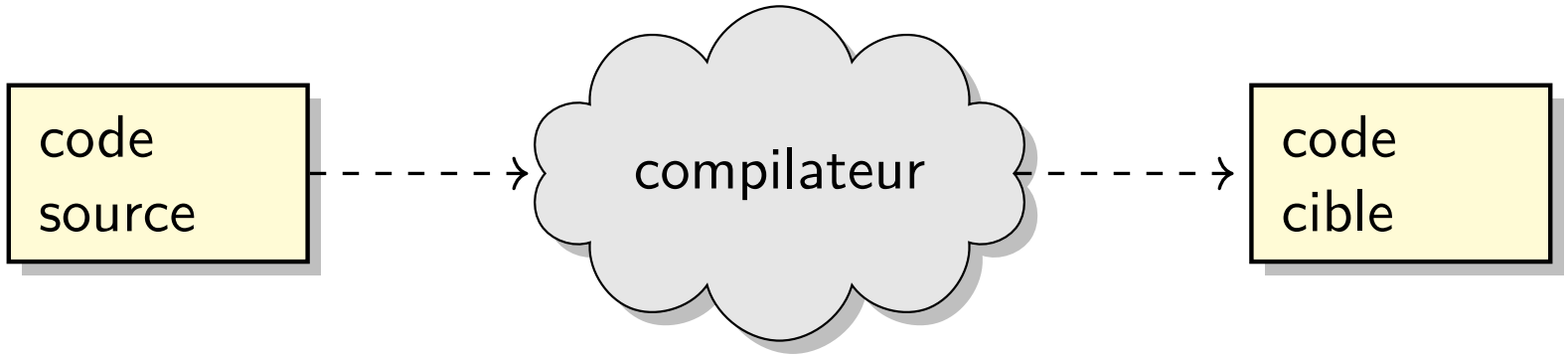
- Entrée : du code *source*, écrit dans un certain langage.
  - Traditionnellement : une suite de caractères dans un fichier.
- Sortie : du code *cible*, dans un langage plus proche de la machine.

Le module Compilation, objectif premier



- Entrée : du code *source*, écrit dans un certain langage.
  - Traditionnellement : une suite de caractères dans un fichier.
- Sortie : du code *cible*, dans un langage plus proche de la machine.
  - Traditionnellement : un fichier binaire exécutable.

Le module Compilation, objectif premier



- Entrée : du code *source*, écrit dans un certain langage.
  - Traditionnellement : une suite de caractères dans un fichier.
- Sortie : du code *cible*, dans un langage plus proche de la machine.
  - Traditionnellement : un fichier binaire exécutable.

Objectif premier

Réaliser un compilateur qui traduit un langage fonctionnel et impératif de haut niveau vers du code exécutable pour processeurs x86-64 (Intel/AMD).

Le module Compilation, objectif second

Vous allez écrire un compilateur :

- en OCaml, en partant d'une base de code existante ;
- tout au long du semestre mais étape par étape, de façon guidée ;
- organisés en binômes fixés.

Le module Compilation, objectif second

Vous allez écrire un compilateur :

- en OCaml, en partant d'une base de code existante ;
- tout au long du semestre mais étape par étape, de façon guidée ;
- organisés en binômes fixés.

Objectif second

Vous faire programmer, beaucoup programmer, dans un contexte réaliste.

Le module Compilation, objectif second

Vous allez écrire un compilateur :

- en OCaml, en partant d'une base de code existante ;
- tout au long du semestre mais étape par étape, de façon guidée ;
- organisés en binômes fixés.

Objectif second

Vous faire programmer, beaucoup programmer, dans un contexte réaliste.

Un contexte réaliste

- Omniprésence de la gestion de version (via `git`)
- Batterie de tests automatiques (fournis)
- Importante base de code, où on *lit* plus de code qu'on en écrit.

Pourquoi étudier la compilation ?

Pourquoi étudier la compilation ?

- Pas directement mobilisable dans la plupart des emplois.
  - (Demande pour les spécialistes réelle à l'international, e.g., JavaScript.)

Pourquoi étudier la compilation ?

- Pas directement mobilisable dans la plupart des emplois.
  - (Demande pour les spécialistes réelle à l'international, e.g., JavaScript.)
+ Fera de vous de meilleurs programmeurs, maîtres de leurs outils.
  - Ne pas se retrouver démuni quand les outils dysfonctionnent.

Pourquoi étudier la compilation ?

- Pas directement mobilisable dans la plupart des emplois.
  - (Demande pour les spécialistes réelle à l'international, e.g., JavaScript.)
+ Fera de vous de meilleurs programmeurs, maîtres de leurs outils.
  - Ne pas se retrouver démuni quand les outils dysfonctionnent.
+ Sujet pluridisciplinaire, à la croisée de presque toute l'informatique.
  - Architecture des ordinateurs, génie logiciel, méthodes formelles
  - Théorie des graphes, théorie des langages formels et automates
  - Sémantique des langages de programmation, logique

Pourquoi étudier la compilation ?

- – Pas directement mobilisable dans la plupart des emplois.
  - (Demande pour les spécialistes réelle à l'international, e.g., JavaScript.)
- \+ Fera de vous de meilleurs programmeurs, maîtres de leurs outils.
  - Ne pas se retrouver démuni quand les outils dysfonctionnent.
- \+ Sujet pluridisciplinaire, à la croisée de presque toute l'informatique.
  - Architecture des ordinateurs, génie logiciel, méthodes formelles
  - Théorie des graphes, théorie des langages formels et automates
  - Sémantique des langages de programmation, logique
- \+ Ce module : augmenter votre expérience de la programmation.

Fonctionnement du module

Source unique d'information, le dépôt git du module.

`https://gaufre.informatique.univ-paris-diderot.fr/aguatto/compilation-m1-2023`

Première tâche : lire le syllabus. Seconde : s'inscrire sur la liste de diffusion.

`https://listes.u-paris.fr/wws/info/m1.2023.compilation.info`

Fonctionnement du module

Source unique d'information, le dépôt git du module.

`https://gaufre.informatique.univ-paris-diderot.fr/aguatto/compilation-m1-2023`

Première tâche : lire le syllabus. Seconde : s'inscrire sur la liste de diffusion.

`https://listes.u-paris.fr/wws/info/m1.2023.compilation.info`

Principes de fonctionnement

- Intégralement centré sur la réalisation du projet.
- Le projet est structuré en *jalons* indépendants.
  - Prennent la forme de code à trou. Beaucoup de code à lire.
  - Publiés régulièrement, à rendre toutes les trois semaines.
  - 15–30 minutes de Q/R au début de chaque séance de cours.
- En cours, je présente les concepts impliqués, on en discute. Interactif !
- En travaux pratiques, vous travaillez les jalons en cours de réalisation, discutez avec l'enseignant et entre vous pour avancer.

Ce que les enseignants attendent de vous

- Travailler de manière continue et régulière tout le semestre.
- Terminer les jalons à temps. Le rendu est automatique.
- Prêter attention à la qualité du code.
- En cours : prendre des notes ; mener une réflexion critique.
- En TP : participer et discuter des jalons.
- Chez vous : lire le code et les documents fournis, poster sur la liste.

Ce que les enseignants attendent de vous

- Travailler de manière continue et régulière tout le semestre.
- Terminer les jalons à temps. Le rendu est automatique.
- Prêter attention à la qualité du code.
- En cours : prendre des notes ; mener une réflexion critique.
- En TP : participer et discuter des jalons.
- Chez vous : lire le code et les documents fournis, poster sur la liste.

Évaluation

- Note du module = 70% projet + 30% examen terminal.
- La note de projet est fixée après des soutenances individuelles.
  - La qualité du code est prise en compte.

Le reste de cette séance

On découvre la compilation avec `cours-01/marthe.ml`.

Le reste de cette séance

On découvre la compilation avec `cours-01/marthe.ml`.

Pour la prochaine fois

- S'assurer d'avoir un OCaml, OPAM et Dune fonctionnels.
- Répondre aux exercices de `marthe.ml`.