Les 25 ans d'Aconit

Au cœur de l'informatique, le logiciel

Sacha Krakowiak
Université de Grenoble

20 mai 2010

Questions sur le logiciel

Questions sur le logiciel

- À quoi çà sert ?
- Quels en sont les principes ?
- À quoi çà ressemble ?
- Comment çà marche ?
- Comment çà se fabrique ?
- Qui s'en occupe ?
  - Les acteurs du logiciel
- Comment est-on arrivé là ?
  - Un peu d'histoire
- Où va-t-on ?
  - Les grands défis du logiciel

Le logiciel : à quoi çà sert ?

Le logiciel : à quoi çà sert ?

- “Instruire” une machine (ordinateur, processeur) pour la rendre capable d’accomplir une fonction

Le logiciel : à quoi çà sert ?

- “Instruire” une machine (ordinateur, processeur) pour la rendre capable d’accomplir une fonction
- Exemples
  - Un traitement de texte
  - Un jeu vidéo
  - Une messagerie
  - Un achat en ligne
  - Une recherche sur l’Internet
  - Le programme d’une machine à laver
  - Le programme d’un téléphone portable
  - La commande d’un métro automatique
  - … et bien d’autres encore

Le logiciel : à quoi çà sert ?

- “Instruire” une machine (ordinateur, processeur) pour la rendre capable d’accomplir une fonction
- Exemples
  - Un traitement de texte
  - Un jeu vidéo
  - Une messagerie
  - Un achat en ligne
  - Une recherche sur l’Internet
  - Le programme d’une machine à laver
  - Le programme d’un téléphone portable
  - La commande d’un métro automatique
  - … et bien d’autres encore

Autre façon de voir : le logiciel transforme une machine *universelle* en une machine *spécialisée*

Le logiciel : à quoi çà sert ?

- “Instruire” une machine (ordinateur, processeur) pour la rendre capable d’accomplir une fonction
- Exemples
  - Un traitement de texte
  - Un jeu vidéo
  - Une messagerie
  - Un achat en ligne
  - Une recherche sur l’Internet
  - Le programme d’une machine à laver
  - Le programme d’un téléphone portable
  - La commande d’un métro automatique
  - … et bien d’autres encore

Autre façon de voir : le logiciel transforme une machine *universelle* en une machine *spécialisée*

Plusieurs de ces machines “virtuelles” peuvent coexister sur une machine “physique”

Le logiciel : principes de base (1/4)

Le logiciel : principes de base (1/4)

- Le problème

  La “distance” entre ce que peut faire la machine et ce qu’on demande à l’application

Le logiciel : principes de base (1/4)

- Le problème
  - La “distance” entre ce que peut faire la machine et ce qu’on demande à l’application
  - Ce que peut faire la machine : combiner, comparer, déplacer (très vite et sans faute) des “bits” d’information (0 ou 1)

Le logiciel : principes de base (1/4)

- Le problème
  - La “distance” entre ce que peut faire la machine et ce qu’on demande à l’application
  - Ce que peut faire la machine : combiner, comparer, déplacer (très vite et sans faute) des “bits” d’information (0 ou 1)
  - Ce qu’on demande à l’application : des choses *beaucoup* plus complexes
    - faire des calculs compliqués
    - enchaîner des actions par une démarche logique
    - explorer des grandes masses d’information
    - afficher des images, les faire changer de taille ou de forme
    - accéder à des services sur l’Internet
    - piloter des appareils en temps réel

Le logiciel : principes de base (2/4)

Le logiciel : principes de base (2/4)

- La distance machine-applications
  - Différence de complexité
  - Différence de mode d'expression

Le logiciel : principes de base (2/4)

- La distance machine-applications
  - Différence de complexité
  - Différence de mode d'expression
- Des voies d'approche
  - Franchir la distance *progressivement*
    - a) par niveaux
      - ne pas chercher à traiter tous les aspects à la fois
      - se concentrer au début sur l'essentiel (abstraction), préciser les détails plus tard (raffinement)

Le logiciel : principes de base (2/4)

- La distance machine-applications
  - Différence de complexité
  - Différence de mode d'expression
- Des voies d'approche
  - Franchir la distance *progressivement*
    - a) par niveaux
      - ne pas chercher à traiter tous les aspects à la fois
      - se concentrer au début sur l'essentiel (abstraction), préciser les détails plus tard (raffinement)
    - b) par étapes à chaque niveau (décomposition)
      - diviser chaque problème en sous-problèmes

Le logiciel : principes de base (2/4)

- La distance machine-applications
  - Différence de complexité
  - Différence de mode d'expression
- Des voies d'approche
  - Franchir la distance *progressivement*
    - a) par niveaux
      - ne pas chercher à traiter tous les aspects à la fois
      - se concentrer au début sur l'essentiel (abstraction), préciser les détails plus tard (raffinement)
    - b) par étapes à chaque niveau (décomposition)
      - diviser chaque problème en sous-problèmes
  - Utiliser les bons outils
    - Langages d'expression, outils de composition
    - L'informatique fabrique ses propres outils !

Le logiciel : principes de base (3/4)

Le logiciel : principes de base (3/4)

- Un petit exemple
  - Recherche dans un annuaire …

Le logiciel : principes de base (3/4)

- Un petit exemple
  - Recherche dans un annuaire …

Aaron A. <adresse, tel.>

…

Zyzzywa Z. <adresse, tel.>

Le logiciel : principes de base (3/4)

- Un petit exemple

  Recherche dans un annuaire …

Garin

Aaron A. <adresse, tel.>

…

Zyzzywa Z. <adresse, tel.>

Le logiciel : principes de base (3/4)

- Un petit exemple

  Recherche dans un annuaire …

Garin

Aaron A. <adresse, tel.>

Garin G. <adresse, tel.>

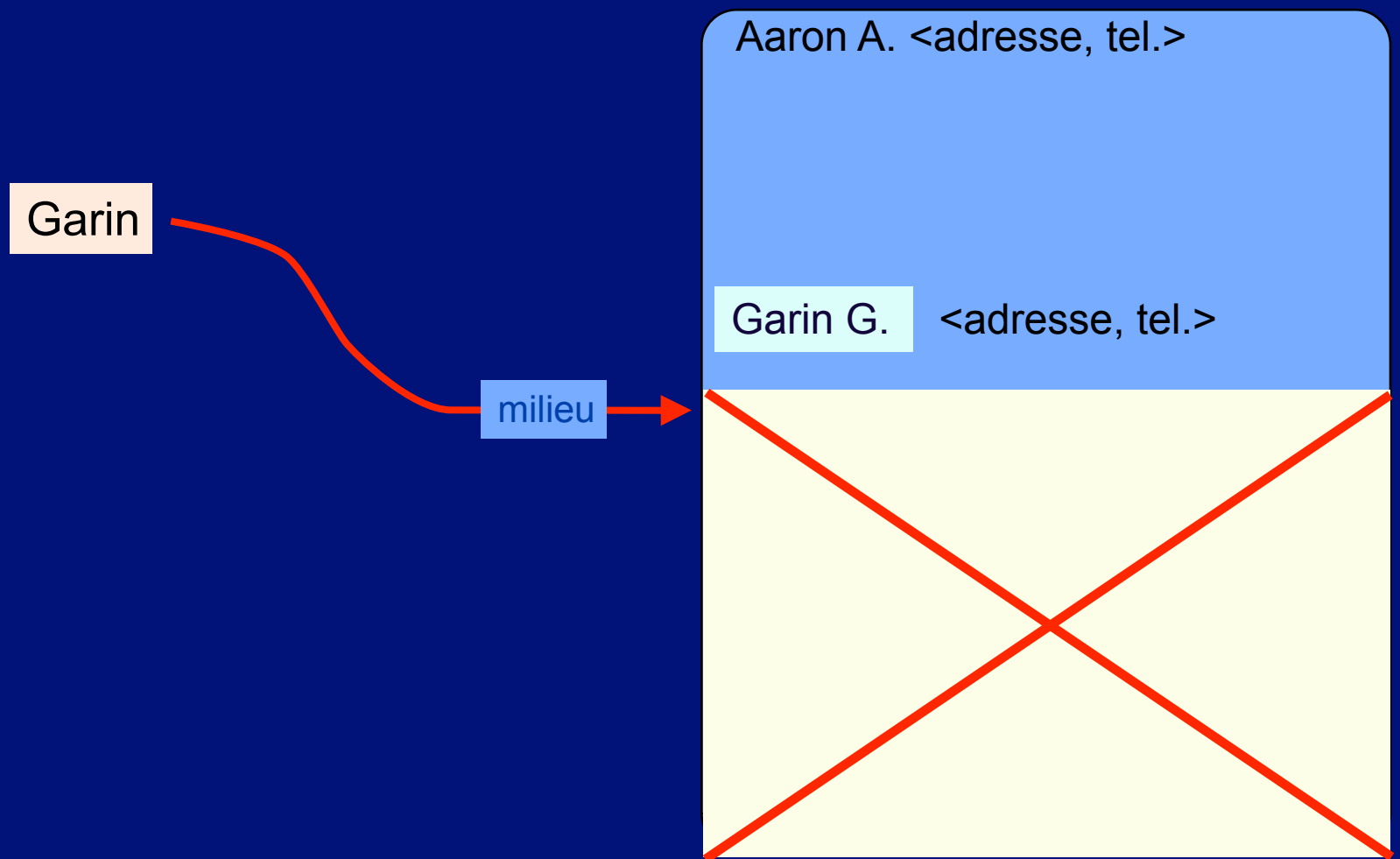
…

Zyzzywa Z. <adresse, tel.>

Le logiciel : principes de base (3/4)

- Un petit exemple

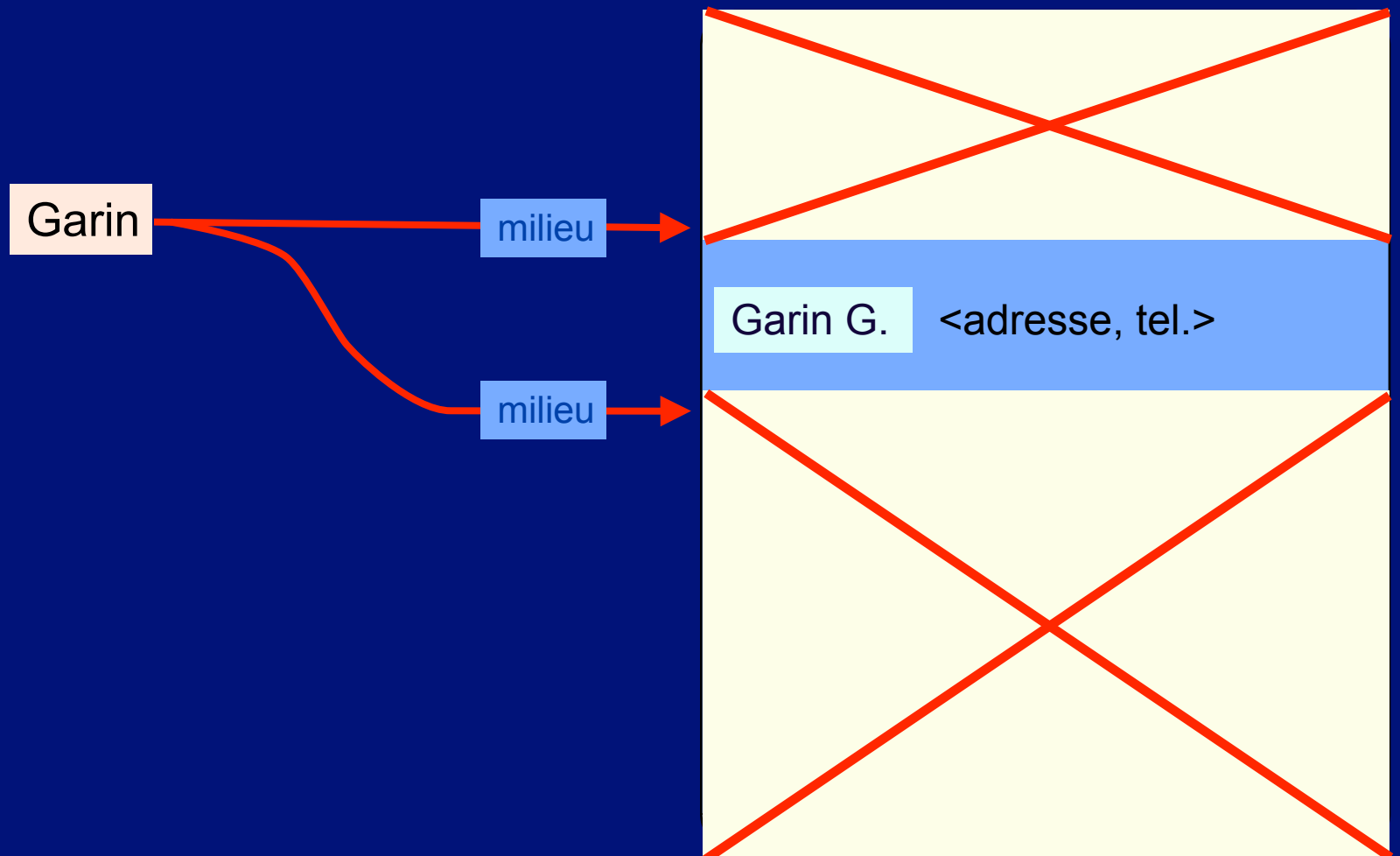
  Recherche dans un annuaire …



Le logiciel : principes de base (3/4)

- Un petit exemple

  Recherche dans un annuaire …



Le logiciel : principes de base (3/4)

- Un petit exemple

  Recherche dans un annuaire …

Garin

Aaron A. <adresse, tel.>

Garin G. <adresse, tel.>

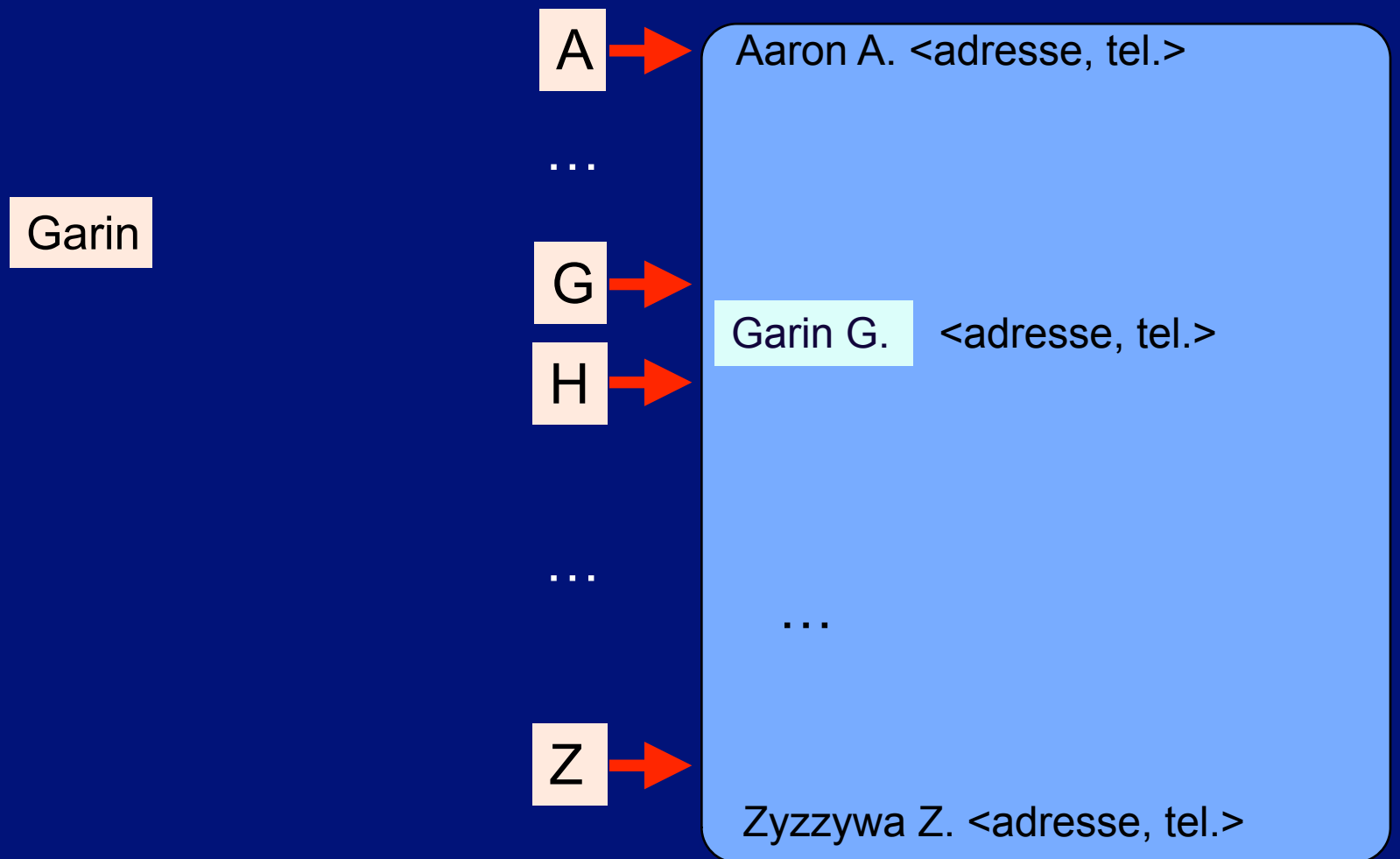
…

Zyzzywa Z. <adresse, tel.>

Le logiciel : principes de base (3/4)

- Un petit exemple

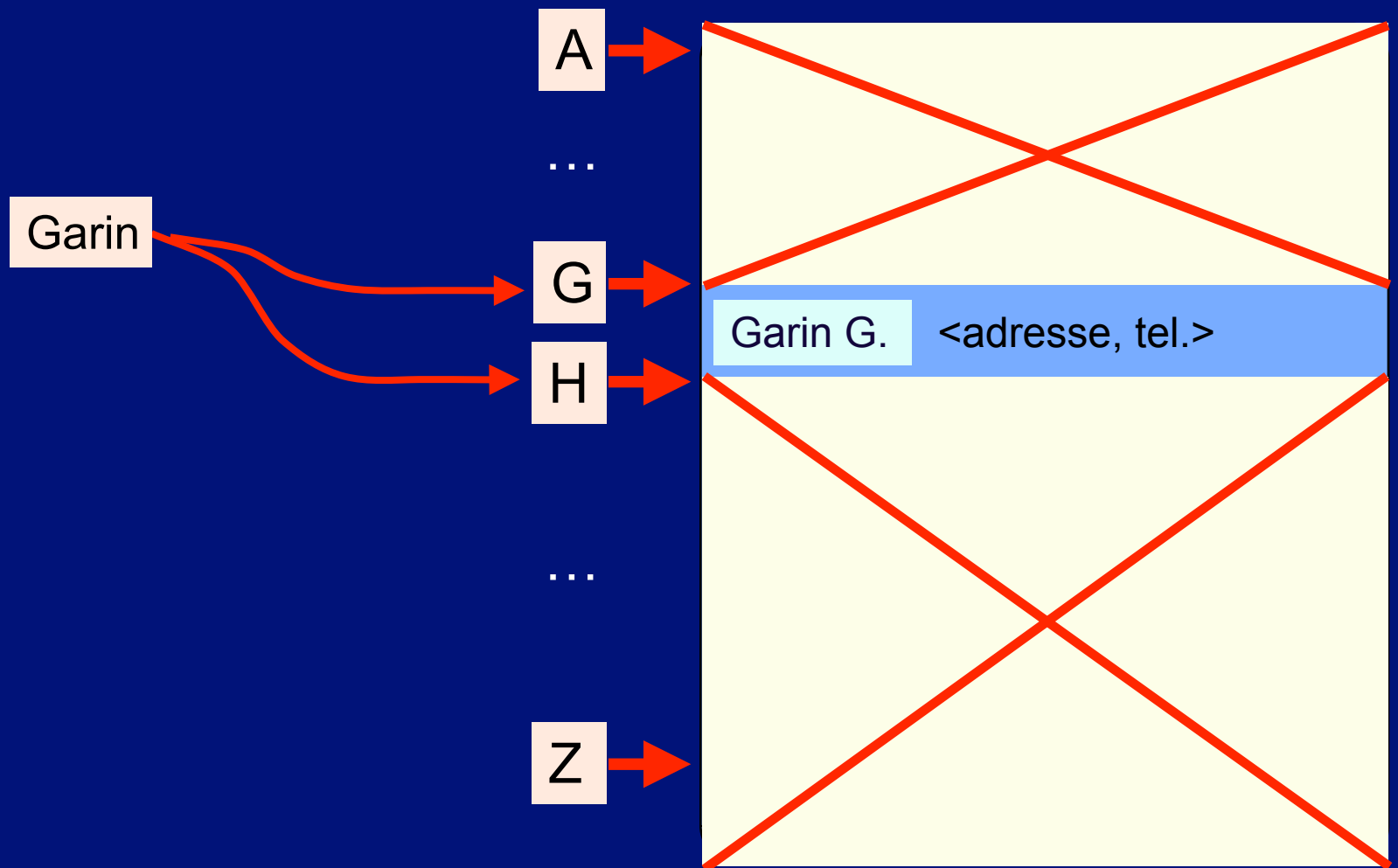
  Recherche dans un annuaire …



Le logiciel : principes de base (3/4)

- Un petit exemple

  Recherche dans un annuaire …



Le logiciel : principes de base (3/4)

- Un petit exemple

  Recherche dans un annuaire …

Garin

Aaron A. <adresse, tel.>

Garin G. <adresse, tel.>

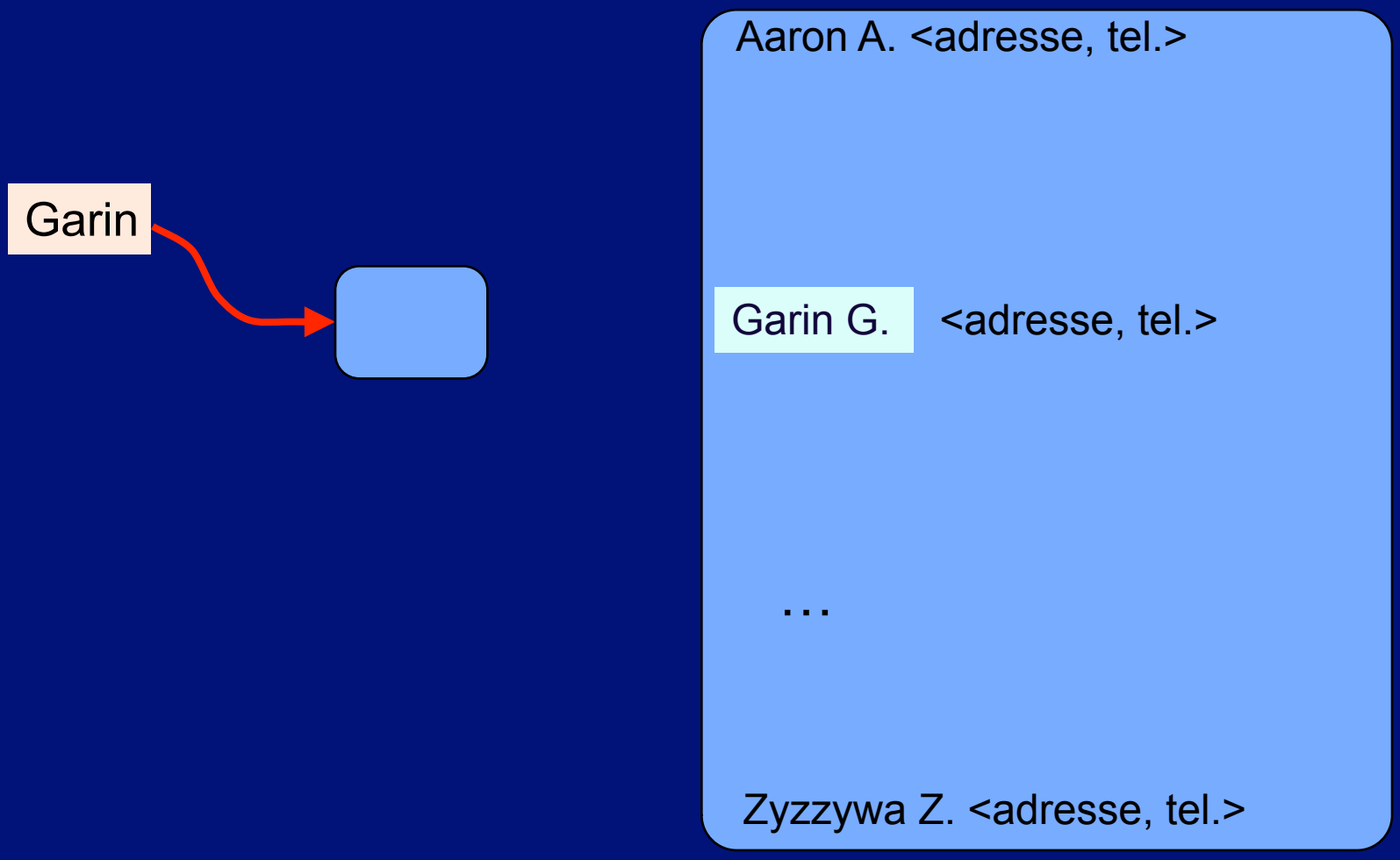
…

Zyzzywa Z. <adresse, tel.>

Le logiciel : principes de base (3/4)

- Un petit exemple

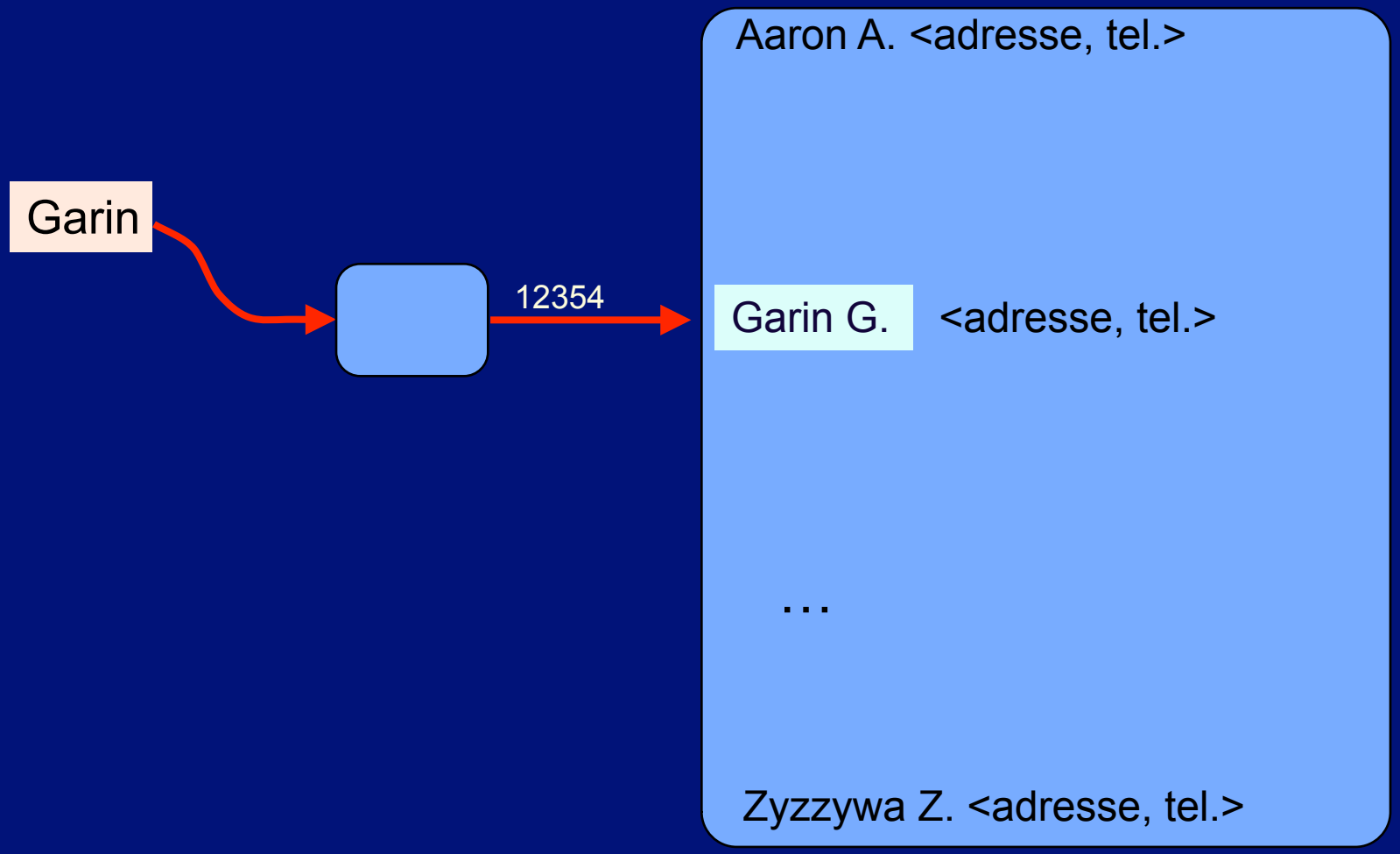
  Recherche dans un annuaire …



Le logiciel : principes de base (3/4)

- Un petit exemple

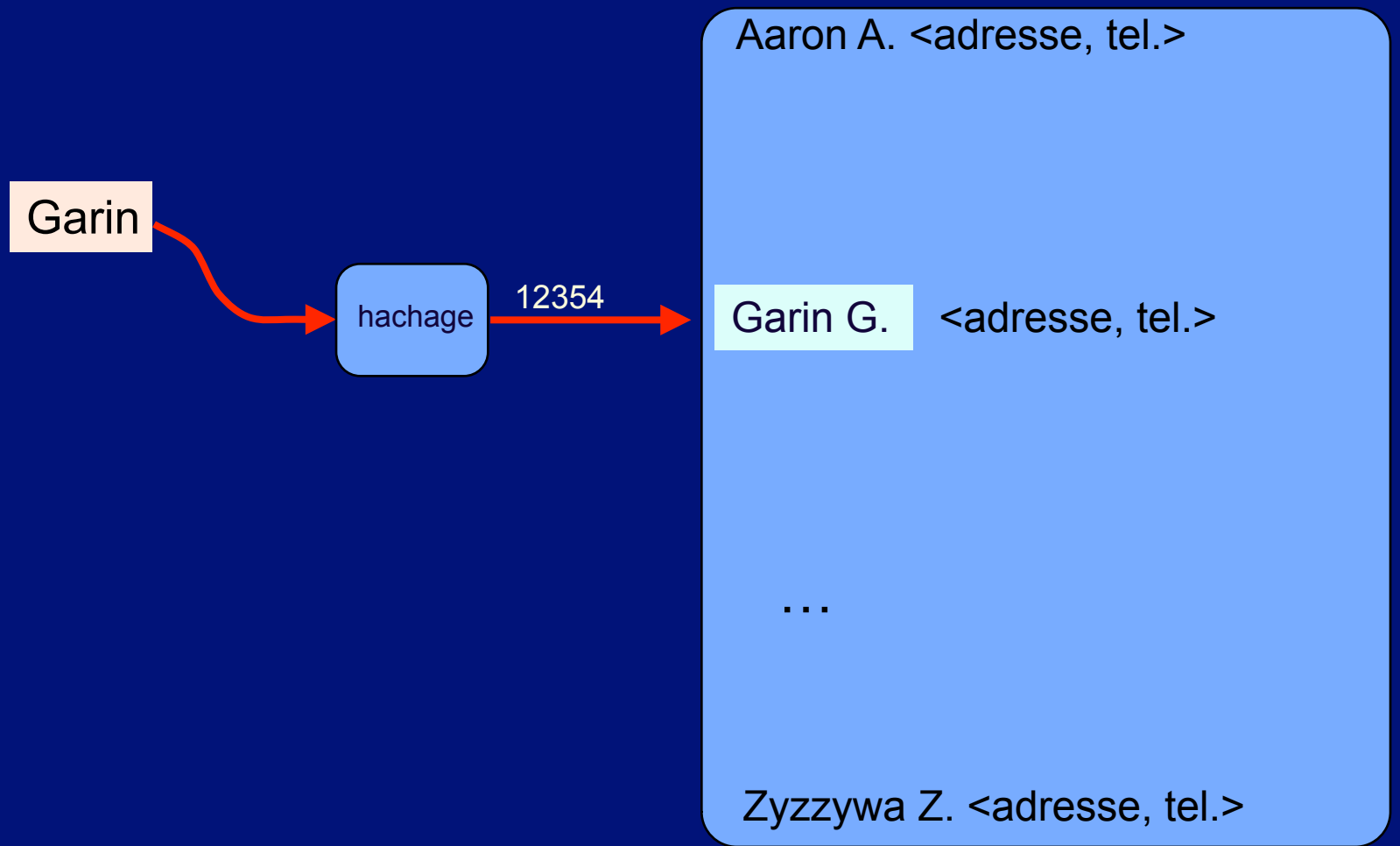
  Recherche dans un annuaire …



Le logiciel : principes de base (3/4)

- Un petit exemple

  Recherche dans un annuaire …



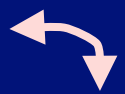
Le logiciel : principes de base (4/4)

Le logiciel : principes de base (4/4)

- Les choses à faire
  - Choisir une méthode

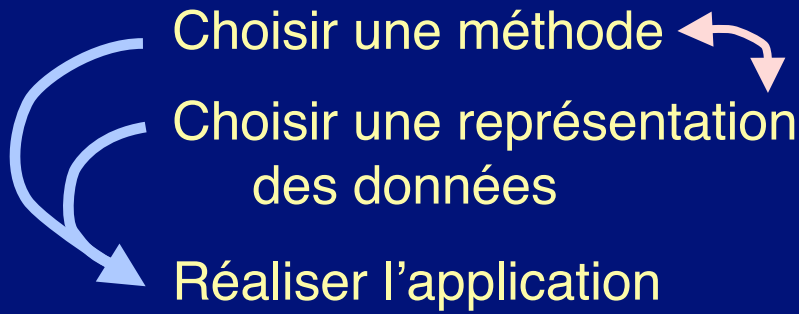
Le logiciel : principes de base (4/4)

- Les choses à faire
  - Choisir une méthode
  - Choisir une représentation des données



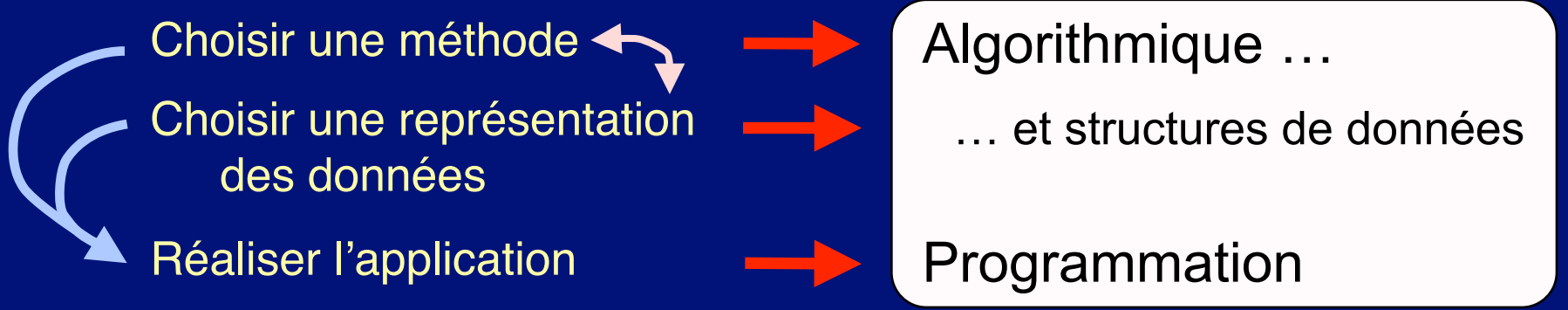
Le logiciel : principes de base (4/4)

- Les choses à faire



Le logiciel : principes de base (4/4)

- Les choses à faire



Le logiciel : principes de base (4/4)

- Les choses à faire

Choisir une méthode ↔ → Algorithmique …

Choisir une représentation des données → … et structures de données

Réaliser l'application → Programmation

- Une bonne méthode, c'est important …

Le logiciel : principes de base (4/4)

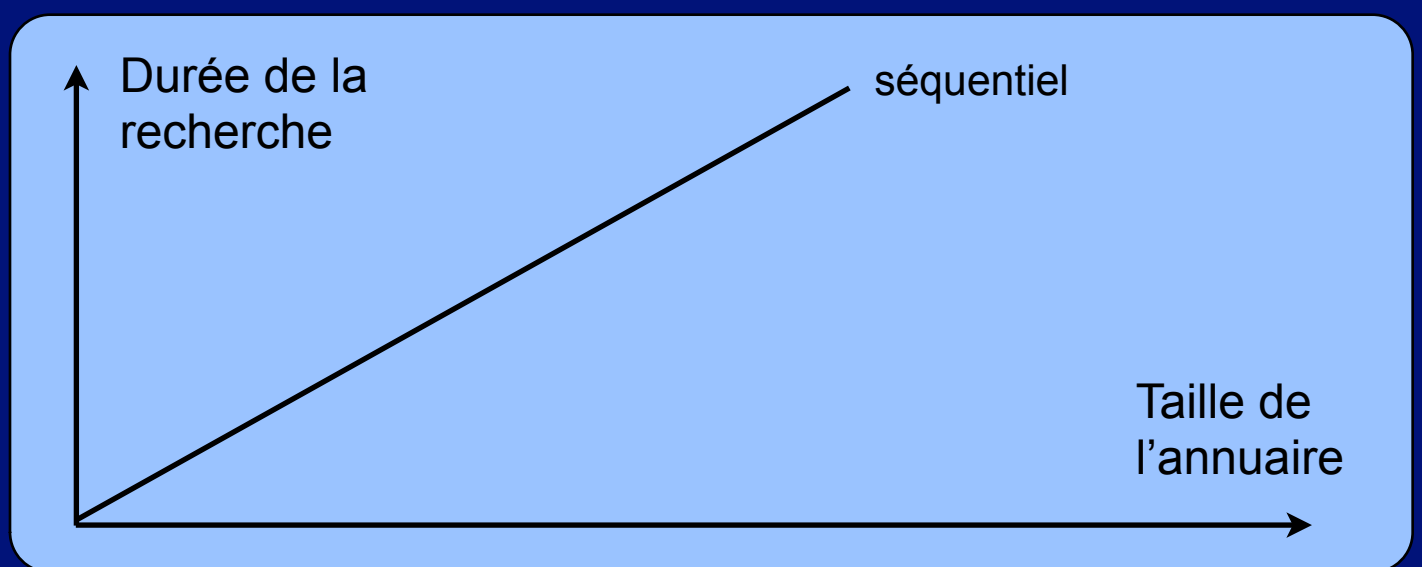
- Les choses à faire

Choisir une méthode ↔ → Algorithmique …

Choisir une représentation des données → … et structures de données

Réaliser l'application → Programmation

- Une bonne méthode, c'est important …



Le logiciel : principes de base (4/4)

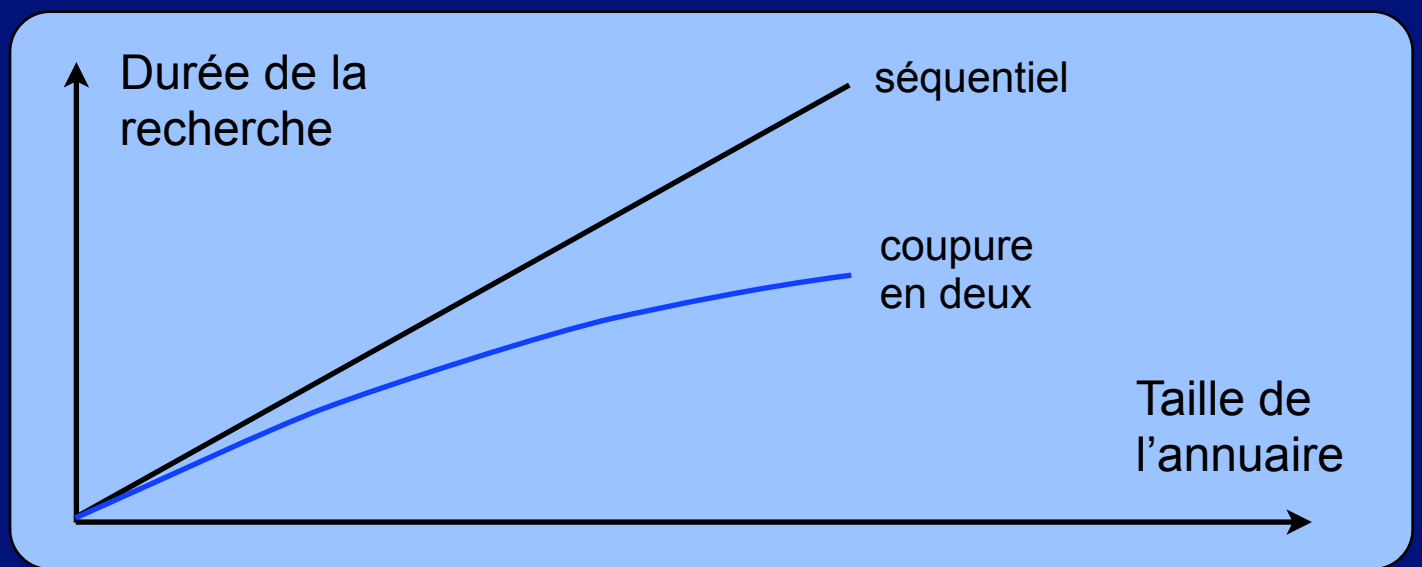
- Les choses à faire

Choisir une méthode ↔ Choisir une représentation des données → Algorithmique …

… et structures de données

Réaliser l'application → Programmation

- Une bonne méthode, c'est important …



Le logiciel : principes de base (4/4)

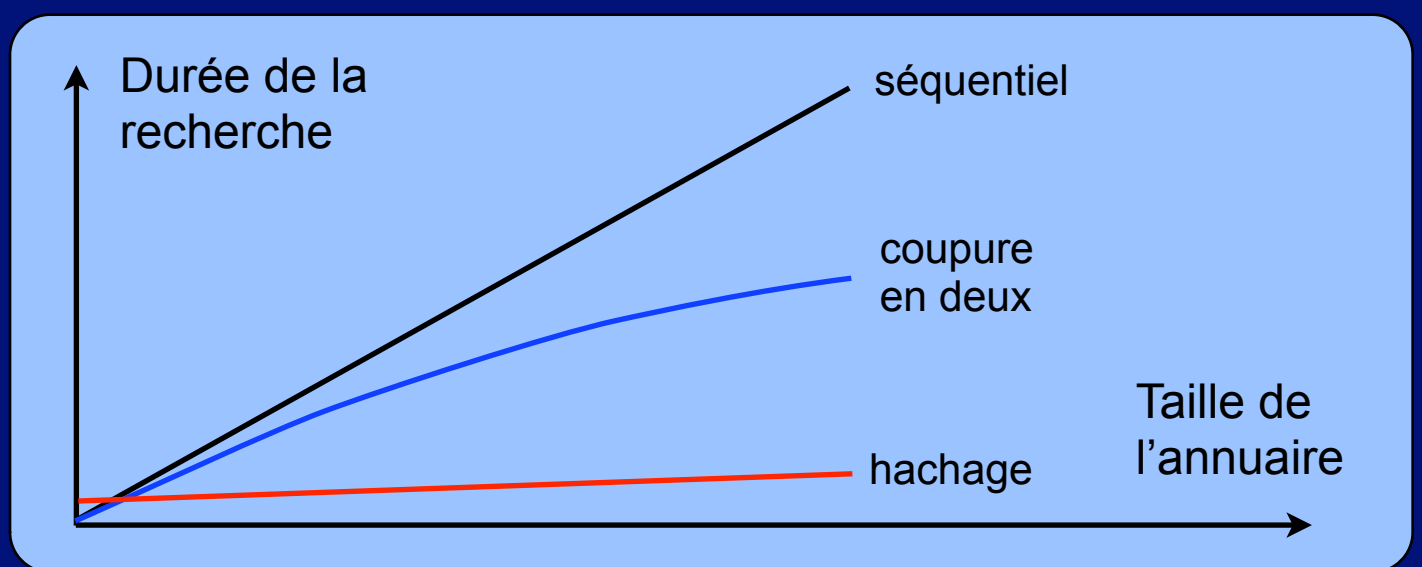
- Les choses à faire

Choisir une méthode ↔ Choisir une représentation des données → Algorithmique …

… et structures de données

Réaliser l'application → Programmation

- Une bonne méthode, c'est important …



Le logiciel : principes de base (4/4)

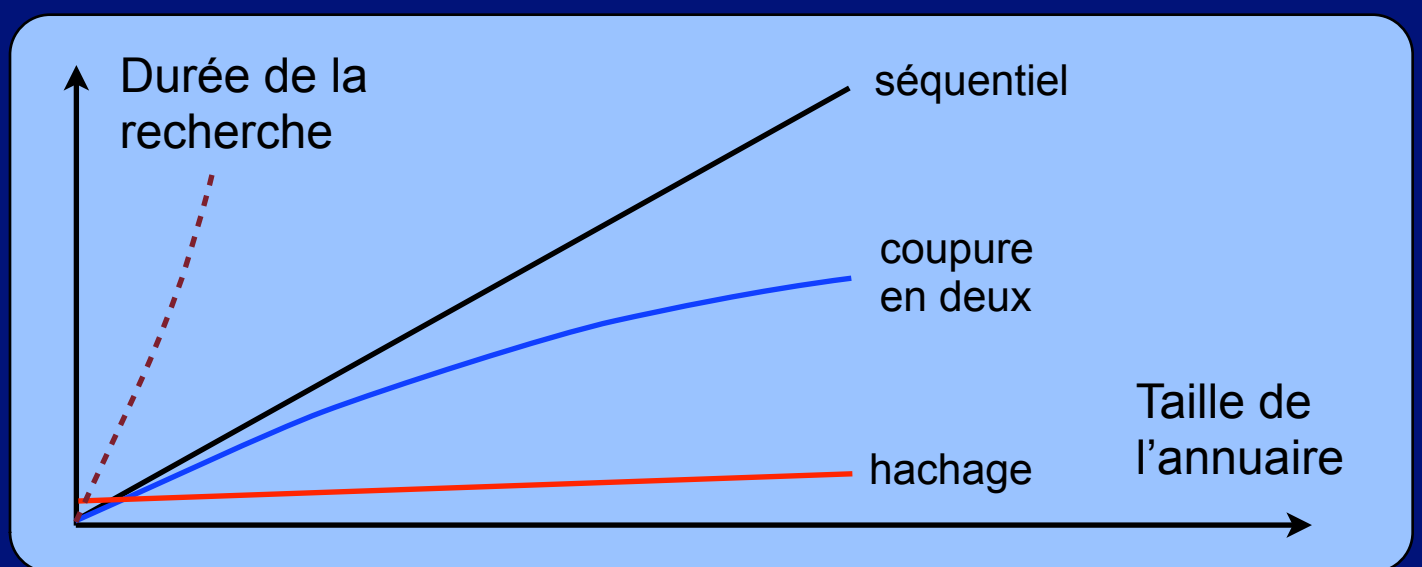
- Les choses à faire
  - Choisir une méthode
  - Choisir une représentation des données
  - Réaliser l'application

→ Algorithmique …

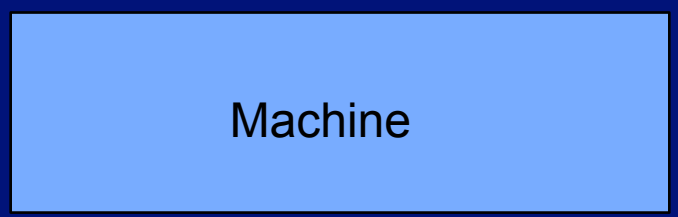
→ … et structures de données

→ Programmation

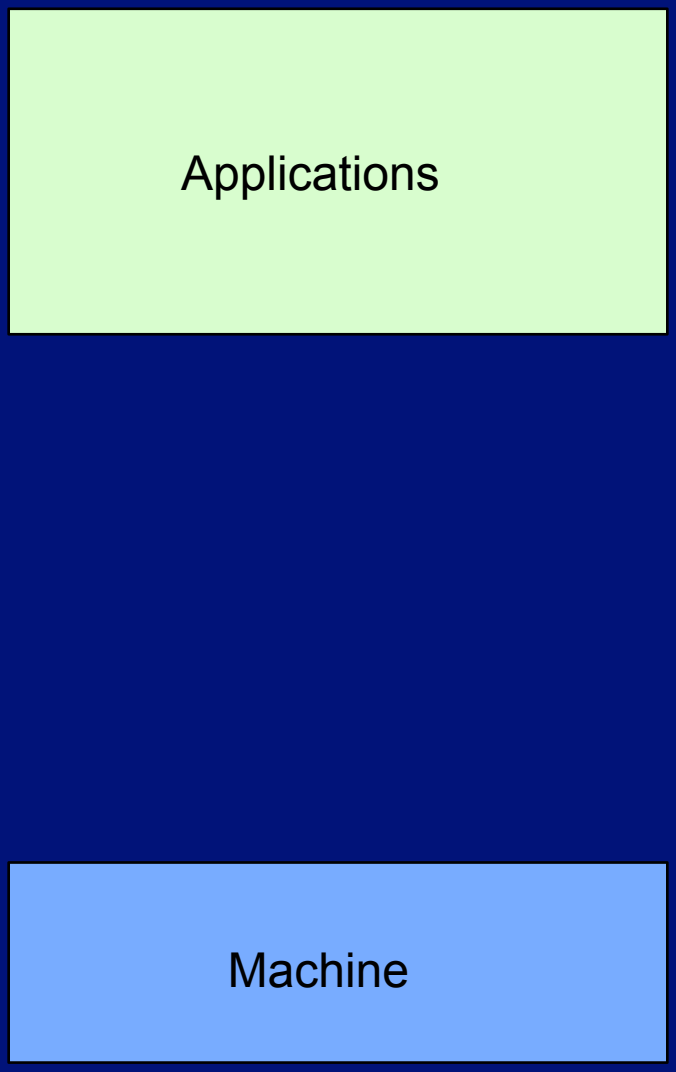
- Une bonne méthode, c'est important …



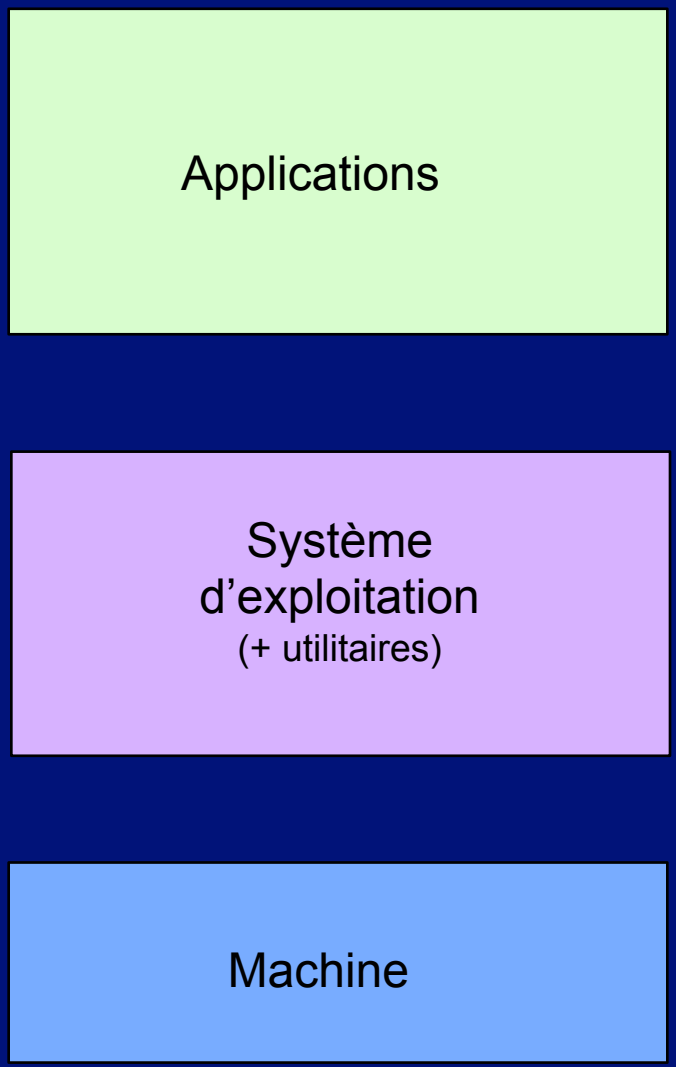
Les niveaux de logiciel



Les niveaux de logiciel

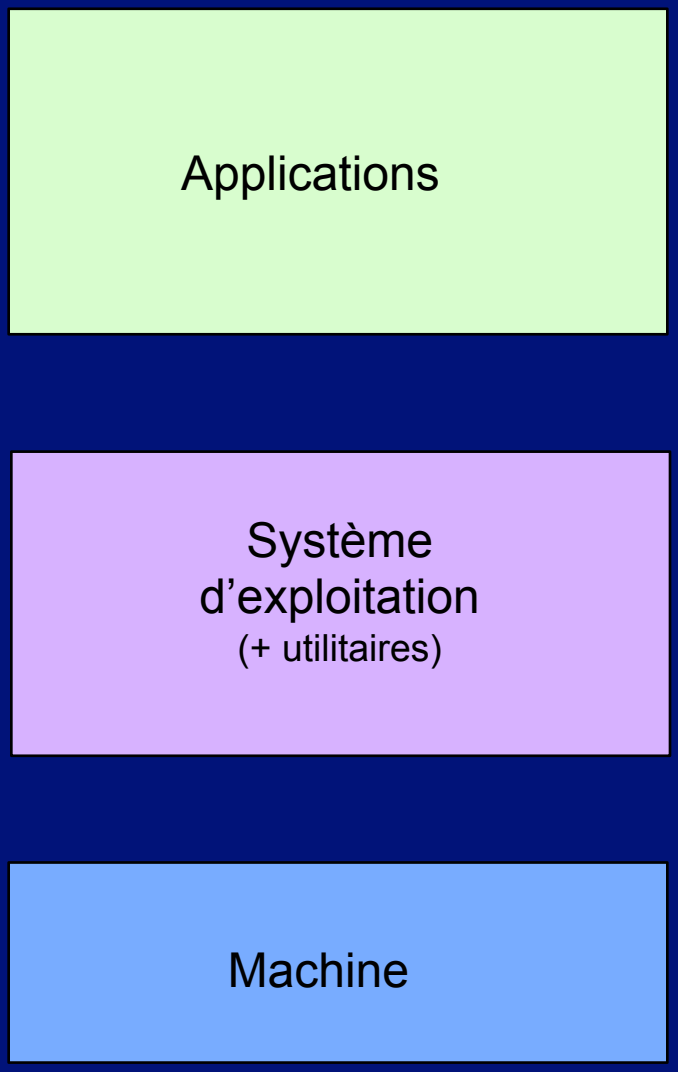


Les niveaux de logiciel



Les services communs à plusieurs applications : le système d'exploitation

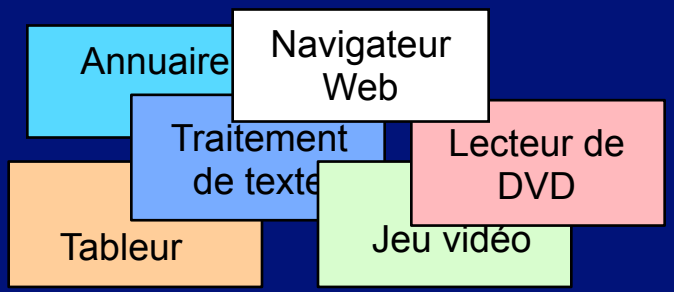
Les niveaux de logiciel



Les services communs à plusieurs applications : le système d'exploitation

Gestion aisée des informations (fichiers)
Exécution de plusieurs applications à la fois
Organisation de l'écran (fenêtres)
Communication avec l'Internet

Les niveaux de logiciel



Les fonctions propres à des applications particulières : les logiciels d'application

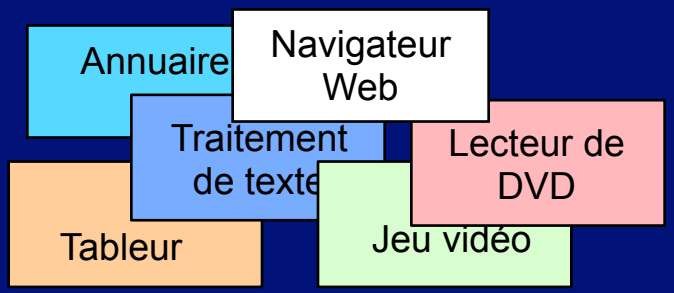
Système d'exploitation
(+ utilitaires)

Les services communs à plusieurs applications : le système d'exploitation

Machine

Gestion aisée des informations (fichiers)
Exécution de plusieurs applications à la fois
Organisation de l'écran (fenêtres)
Communication avec l'Internet

Les niveaux de logiciel



Système d'exploitation
(+ utilitaires)

Machine

Les fonctions propres à des applications particulières : les logiciels d'application

Chaque application est elle-même décomposée en plusieurs niveaux …

Les services communs à plusieurs applications : le système d'exploitation

Gestion aisée des informations (fichiers)
Exécution de plusieurs applications à la fois
Organisation de l'écran (fenêtres)
Communication avec l'Internet

Organisation d’un service : les fichiers

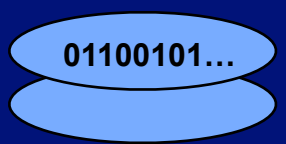
Organisation d'un service : les fichiers

- Gérer les fichiers : une fonction du système d'exploitation
  - Un fichier : une manière de conserver et d'organiser des données
    - leur donner un nom, les classer
    - les retrouver et les utiliser facilement
  - L'idée : mettre en commun la gestion de tous les types de fichiers
    - du texte, des photos, des images, des programmes, de la musique, …

Organisation d'un service : les fichiers

- Gérer les fichiers : une fonction du système d'exploitation
  - Un fichier : une manière de conserver et d'organiser des données
    - leur donner un nom, les classer
    - les retrouver et les utiliser facilement
  - L'idée : mettre en commun la gestion de tous les types de fichiers
    - du texte, des photos, des images, des programmes, de la musique, …

Disque dur

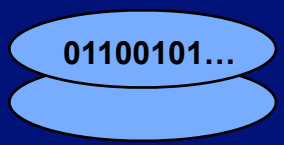


Organisation d'un service : les fichiers

- Gérer les fichiers : une fonction du système d'exploitation
  - Un fichier : une manière de conserver et d'organiser des données
    - leur donner un nom, les classer
    - les retrouver et les utiliser facilement
  - L'idée : mettre en commun la gestion de tous les types de fichiers
    - du texte, des photos, des images, des programmes, de la musique, …

Fichiers

Disque dur



Organisation d'un service : les fichiers

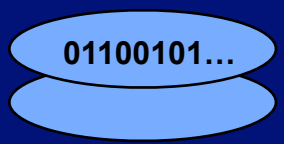
- Gérer les fichiers : une fonction du système d'exploitation
  - Un fichier : une manière de conserver et d'organiser des données
    - leur donner un nom, les classer
    - les retrouver et les utiliser facilement
  - L'idée : mettre en commun la gestion de tous les types de fichiers
    - du texte, des photos, des images, des programmes, de la musique, …

Fichiers

Une partie du système d'exploitation

Système de gestion de fichiers

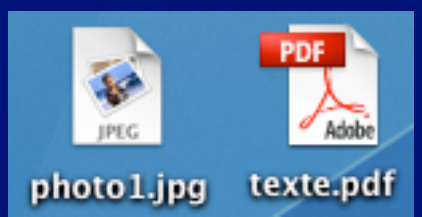
Disque dur



Organisation d'un service : les fichiers

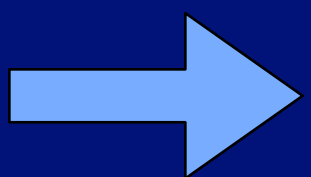
- Gérer les fichiers : une fonction du système d'exploitation
  - Un fichier : une manière de conserver et d'organiser des données
    - leur donner un nom, les classer
    - les retrouver et les utiliser facilement
  - L'idée : mettre en commun la gestion de tous les types de fichiers
    - du texte, des photos, des images, des programmes, de la musique, …

Fichiers



Une partie du système d'exploitation

Système de gestion de fichiers



Disque dur

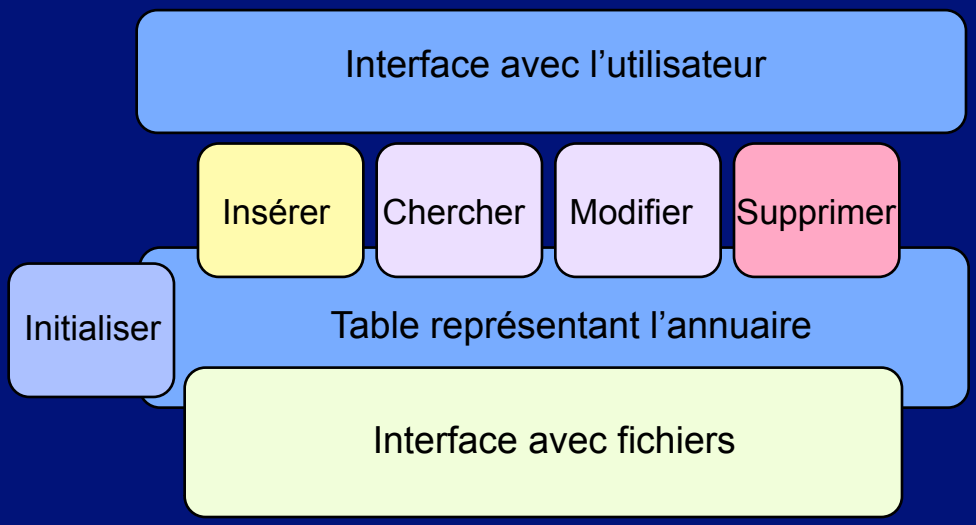
- Afficher les icones à l'écran
- Rechercher à partir du nom
- Créer et organiser les noms
- Transférer entre disque et mémoire
- Créer/supprimer un fichier
- Choisir un codage (convention de représentation)
- Trouver de la place sur le disque

Organisation d'une application : l'annuaire

L'application *Annuaire*

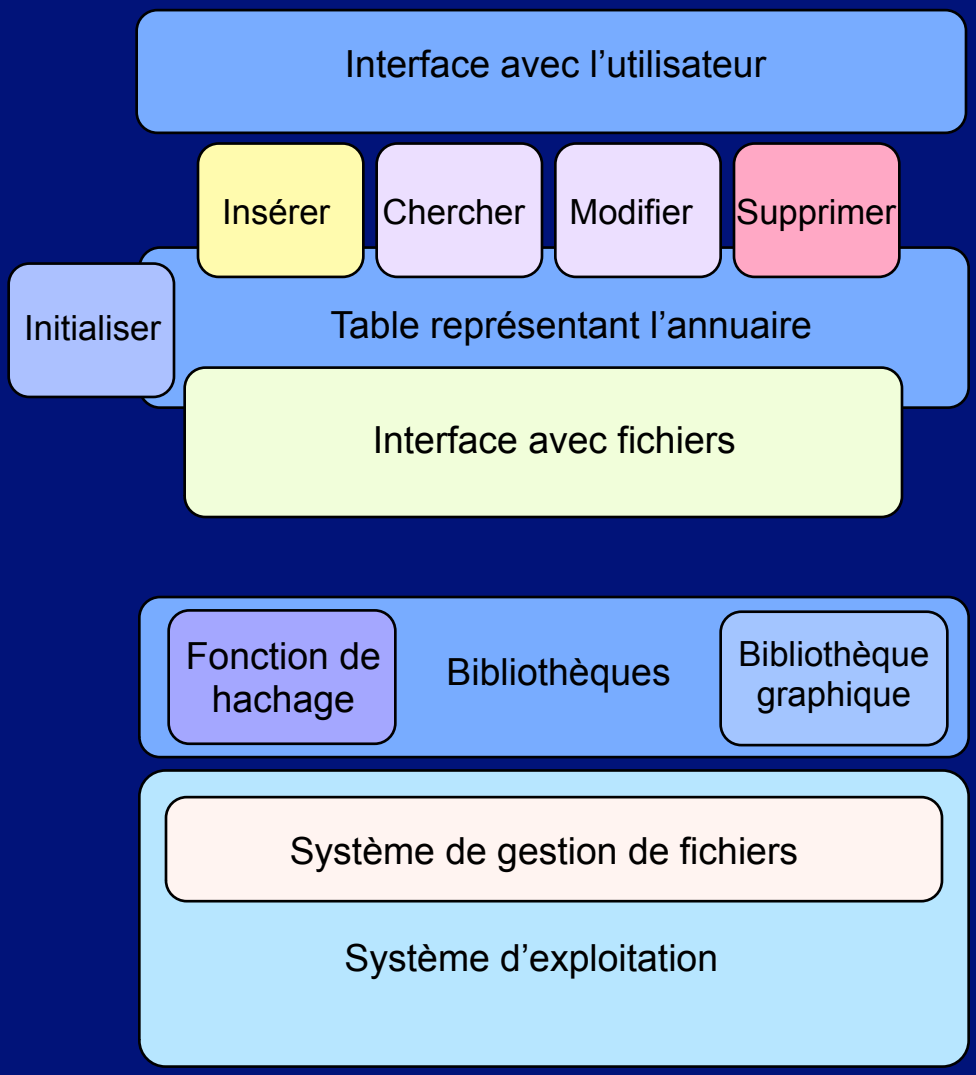
Le logiciel de base

Organisation d'une application : l'annuaire

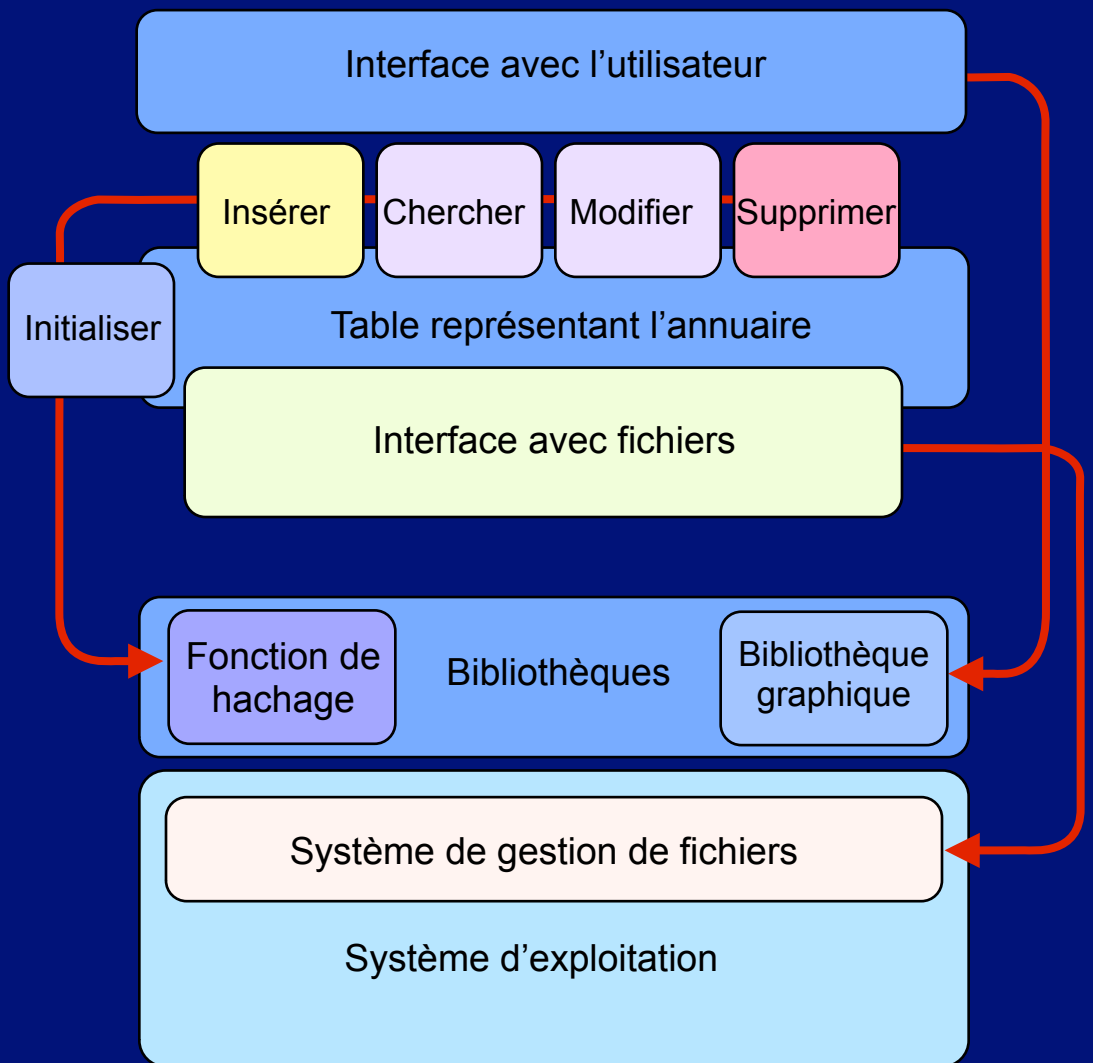


Le logiciel de base

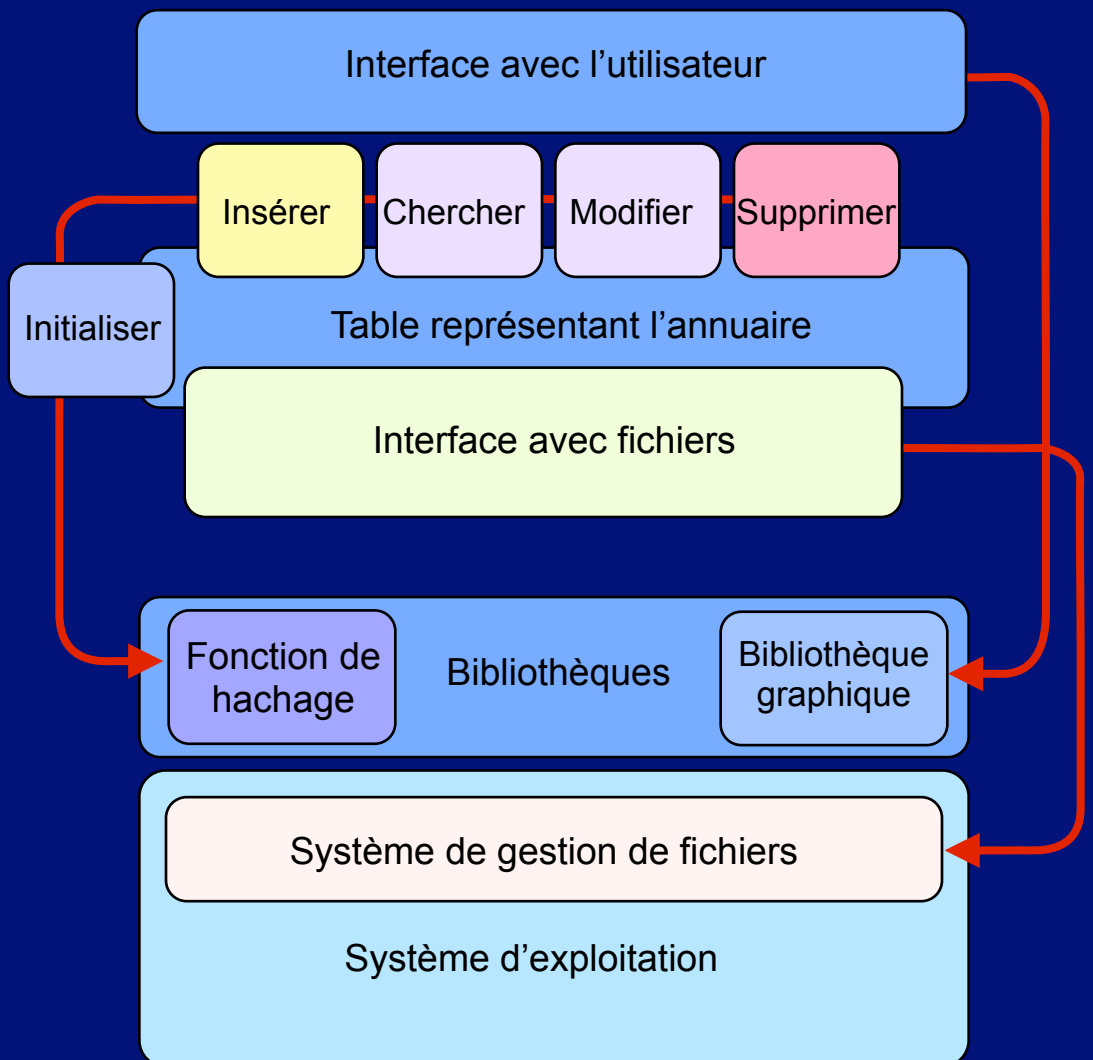
Organisation d'une application : l'annuaire



Organisation d'une application : l'annuaire



Organisation d'une application : l'annuaire



Ce qu'on peut retenir

il est utile d'avoir une vue globale de l'application, en laissant les détails pour plus tard

un programme, même simple, comprend de nombreuses parties reliées entre elles

une application ne s'exécute pas "toute seule" mais fait appel à des services existants

Le logiciel : à quoi çà ressemble ?

Le premier niveau du programme d'un serveur Web

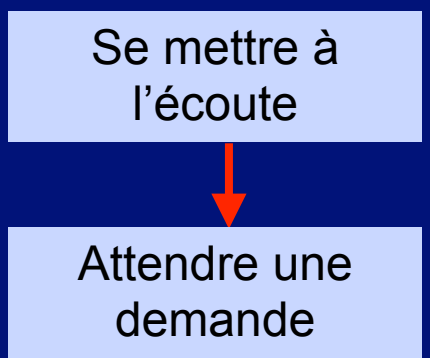
Le logiciel : à quoi çà ressemble ?

Le premier niveau du programme d'un serveur Web

Se mettre à l'écoute

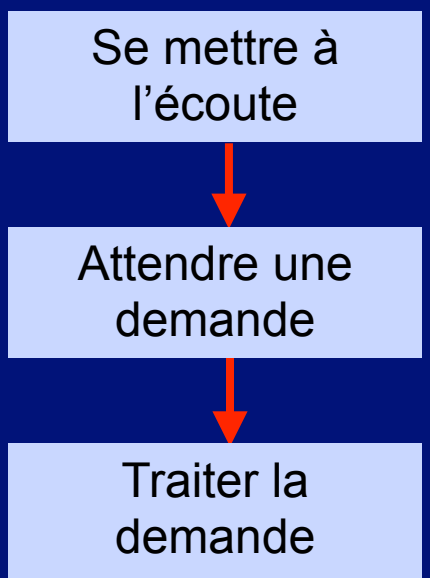
Le logiciel : à quoi çà ressemble ?

Le premier niveau du programme d'un serveur Web



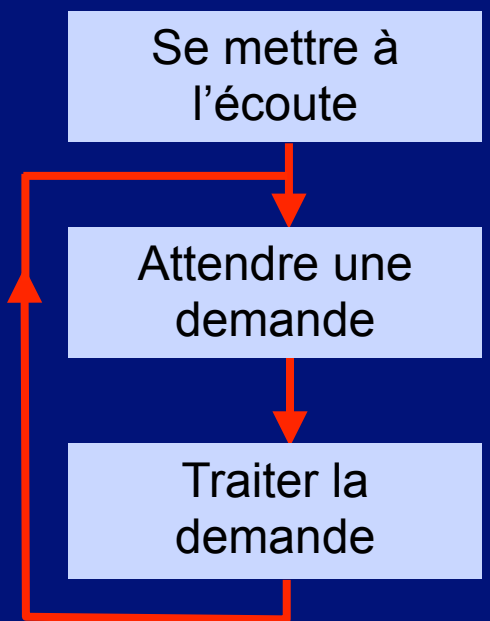
Le logiciel : à quoi çà ressemble ?

Le premier niveau du programme d'un serveur Web



Le logiciel : à quoi çà ressemble ?

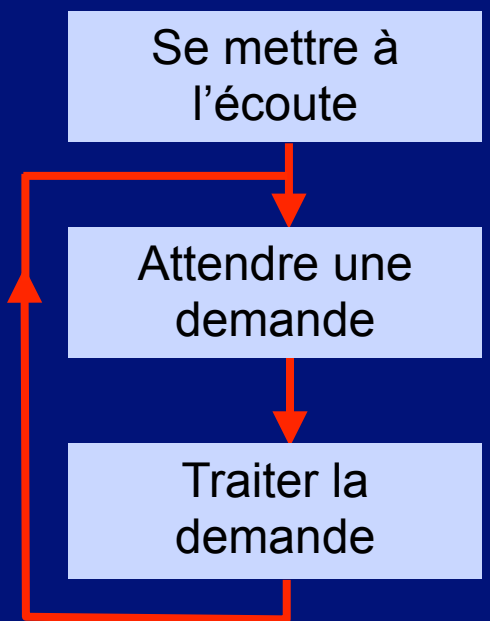
Le premier niveau du programme d'un serveur Web



Le logiciel : à quoi çà ressemble ?

```c
#include "csapp.h"
void read_requesthdrs(rio_t *rp);
int parse_uri(char *uri, char *filename, char *cgiargs);
void serve_static(int fd, char *filename, int filesize);
void get_filetype(char *filename, char *filetype);
void serve_dynamic(int fd, char *filename, char *cgiargs);
void clienterror(int fd, char *cause, char *errnum,
        char *shortmsg, char *longmsg);
int main(int argc, char **argv) {
    int listenfd, connfd, port, clientlen;
    struct sockaddr_in clientaddr;
    /* Vérifier paramètres de la commande */
    if (argc != 2) {
        fprintf(stderr, "usage: %s <port>\n", argv[0]);
        exit(1);
    }
    port = atoi(argv[1]);
    listenfd = Open_listenfd(port); /* Ouvrir la connexion */
    while (TRUE) { /* Boucle d'attente des requêtes */
        clientlen = sizeof(clientaddr);
        connfd = Accept(listenfd, (SA *)&clientaddr,
                    &clientlen);
        traiter(connfd); /* Traitement de la requête */
        Close(connfd);
    }
```

Le premier niveau du programme d'un serveur Web



Source : R. E. Bryant, D. O'Hallaron. *Computer Systems: a Programmer's Perspective*, Prentice Hall, 2003

Programme d’un serveur web (2)

Programme d'un serveur web (2)

```
void traiter(int fd) {
    int is_static;
    struct stat sbuf;
    char buf[MAXLINE], method[MAXLINE],
           uri[MAXLINE],  version[MAXLINE];
    char filename[MAXLINE], cgiargs[MAXLINE];
    rio_t rio;

    /* Lire l'en-tête de la requête */
    Rio_readinitb(&rio, fd);
    Rio_readlineb(&rio, buf, MAXLINE);
    sscanf(buf, "%s %s %s", method, uri, version);
    if (strcasecmp(method, "GET")) {
       clienterror(fd, method, "501", "Not Implemented",
         "Tiny does not implement this method");
        return;
    }
    read_requesthdrs(&rio);

    /* Extraire l'URI de la requête GET */
    is_static = parse_uri(uri, filename, cgiargs);
    if (stat(filename, &sbuf) < 0) {
      clienterror(fd, filename, "404", "Not found",
       "Tiny couldn't find this file");
          return;
    }
   /* à suivre ... */
```

programme `traiter` : exécution d'une requête HTTP

analyse première ligne de la requête HTTP

n'implémente que la requête HTTP GET

détermine si la requête est statique ou dynamique et isole les paramètres s'il y en a

Source : R. E. Bryant, D. O'Hallaron. *Computer Systems: a Programmer's Perspective*, Prentice Hall, 2003

Programme d’un serveur web (3)

Programme d'un serveur web (3)

```
...
if (is_static) { /* Servir contenu statique */
     if (!(S_ISREG(sbuf.st_mode)) ||  !(S_IRUSR & sbuf.st_mode))
      {
      clienterror(fd, filename, "403", "Forbidden",
        "Tiny couldn't read the file");
      return;
      }
      serve_static(fd, filename, sbuf.st_size);
    }
    else { /* Servir contenu dynamique */


if (!(S_ISREG(sbuf.st_mode)) ||
                !(S_IXUSR & sbuf.st_mode)) {
    clienterror(fd, filename, "403", "Forbidden",
       "Tiny couldn't run the CGI program");
      return;
      }
      serve_dynamic(fd, filename, cgiargs);
    }
 }
}
```

fournit un contenu statique : fichier

erreur si ce n'est pas un fichier "ordinaire" ou si la lecture est interdite

fournit un contenu dynamique : exécution d'un script CGI

erreur si ce n'est pas un fichier "ordinaire" ou si l'exécution est interdite

Source : R. E. Bryant, D. O'Hallaron. *Computer Systems: a Programmer's Perspective*, Prentice Hall, 2003

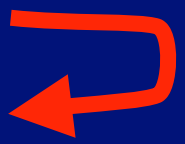
Le logiciel : comment çà se fabrique ?

Le logiciel : comment çà se fabrique ?

- Les étapes de la création
  - Que veut-on faire ?
    - Cahier des charges
    - Spécification

Le logiciel : comment çà se fabrique ?

- Les étapes de la création
  - Que veut-on faire ?
    - Cahier des charges
    - Spécification

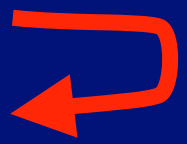


Modélisation

Représentation des objets du monde réel par des objets informatiques

Le logiciel : comment çà se fabrique ?

- Les étapes de la création
  - Que veut-on faire ?
    - Cahier des charges
    - Spécification
  - Comment le faire ?
    - Le principe : l'algorithme
      - une méthode correcte et efficace

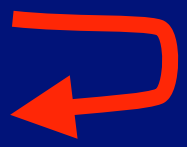


Modélisation

Représentation des objets du monde réel par des objets informatiques

Le logiciel : comment çà se fabrique ?

- Les étapes de la création
  - Que veut-on faire ?
    - Cahier des charges
    - Spécification
  - Comment le faire ?
    - Le principe : l'algorithme
      - une méthode correcte et efficace
    - La mise en œuvre : le programme
      - une réalisation correcte et efficace de l'algorithme

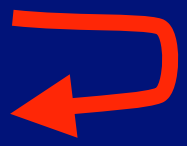


Modélisation

Représentation des objets du monde réel par des objets informatiques

Le logiciel : comment çà se fabrique ?

- Les étapes de la création
  - Que veut-on faire ?
    - Cahier des charges
    - Spécification
  - Comment le faire ?
    - Le principe : l'algorithme
      - une méthode correcte et efficace
    - La mise en œuvre : le programme
      - une réalisation correcte et efficace de l'algorithme
  - A-t-on réussi ?
    - Test
    - Validation et vérification
    - Preuve

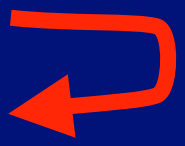


Modélisation

Représentation des objets du monde réel par des objets informatiques

Le logiciel : comment çà se fabrique ?

- Les étapes de la création
  - Que veut-on faire ?
    - Cahier des charges
    - Spécification
  - Comment le faire ?
    - Le principe : l'algorithme
      - une méthode correcte et efficace
    - La mise en œuvre : le programme
      - une réalisation correcte et efficace de l'algorithme
  - A-t-on réussi ?
    - Test
    - Validation et vérification
    - Preuve



Modélisation

Représentation des objets du monde réel par des objets informatiques

La programmation n'est qu'une petite partie de la construction du logiciel

Le logiciel : comment çà se fabrique ?

Le logiciel : comment çà se fabrique ?

- Un mode d'expression
  - Selon le niveau : modèles, langages

Le logiciel : comment çà se fabrique ?

- Un mode d'expression
  - Selon le niveau : modèles, langages
- Des méthodes de travail
  - Définition du problème
  - Recherche d'une solution
  - Décomposition en niveaux et en éléments
  - Programmation et mise au point des éléments
  - Intégration et mise au point de l'ensemble
  - … et on recommence

Le logiciel : comment çà se fabrique ?

- Un mode d'expression
  - Selon le niveau : modèles, langages
- Des méthodes de travail
  - Définition du problème
  - Recherche d'une solution
  - Décomposition en niveaux et en éléments
  - Programmation et mise au point des éléments
  - Intégration et mise au point de l'ensemble
  - … et on recommence
- Des outils
  - Modélisation, traduction (compilateurs), composition, …
  - Test, mise au mise au point, vérification, …

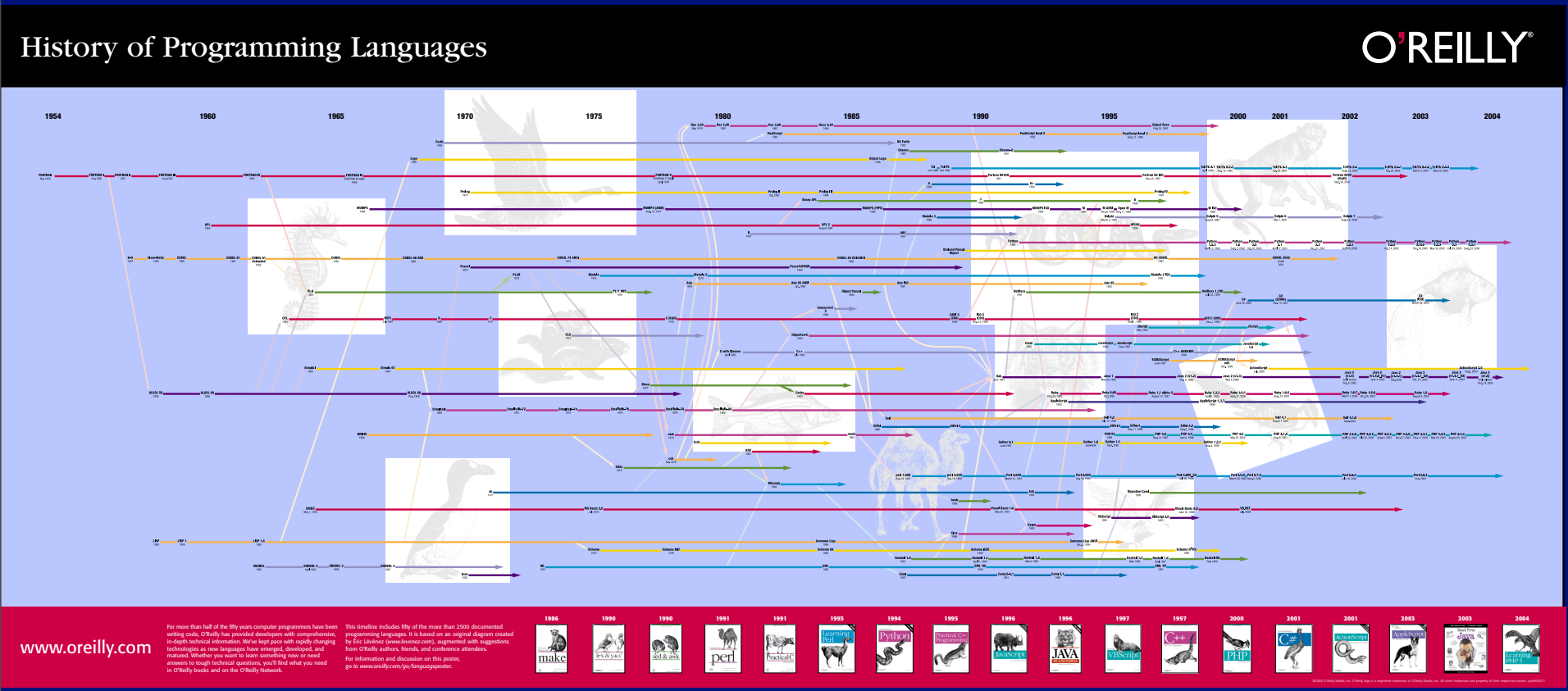
Le logiciel : comment çà se fabrique ?

- Un mode d'expression
  - Selon le niveau : modèles, langages
- Des méthodes de travail
  - Définition du problème
  - Recherche d'une solution
  - Décomposition en niveaux et en éléments
  - Programmation et mise au point des éléments
  - Intégration et mise au point de l'ensemble
  - … et on recommence
- Des outils
  - Modélisation, traduction (compilateurs), composition, …
  - Test, mise au mise au point, vérification, …

Le génie logiciel

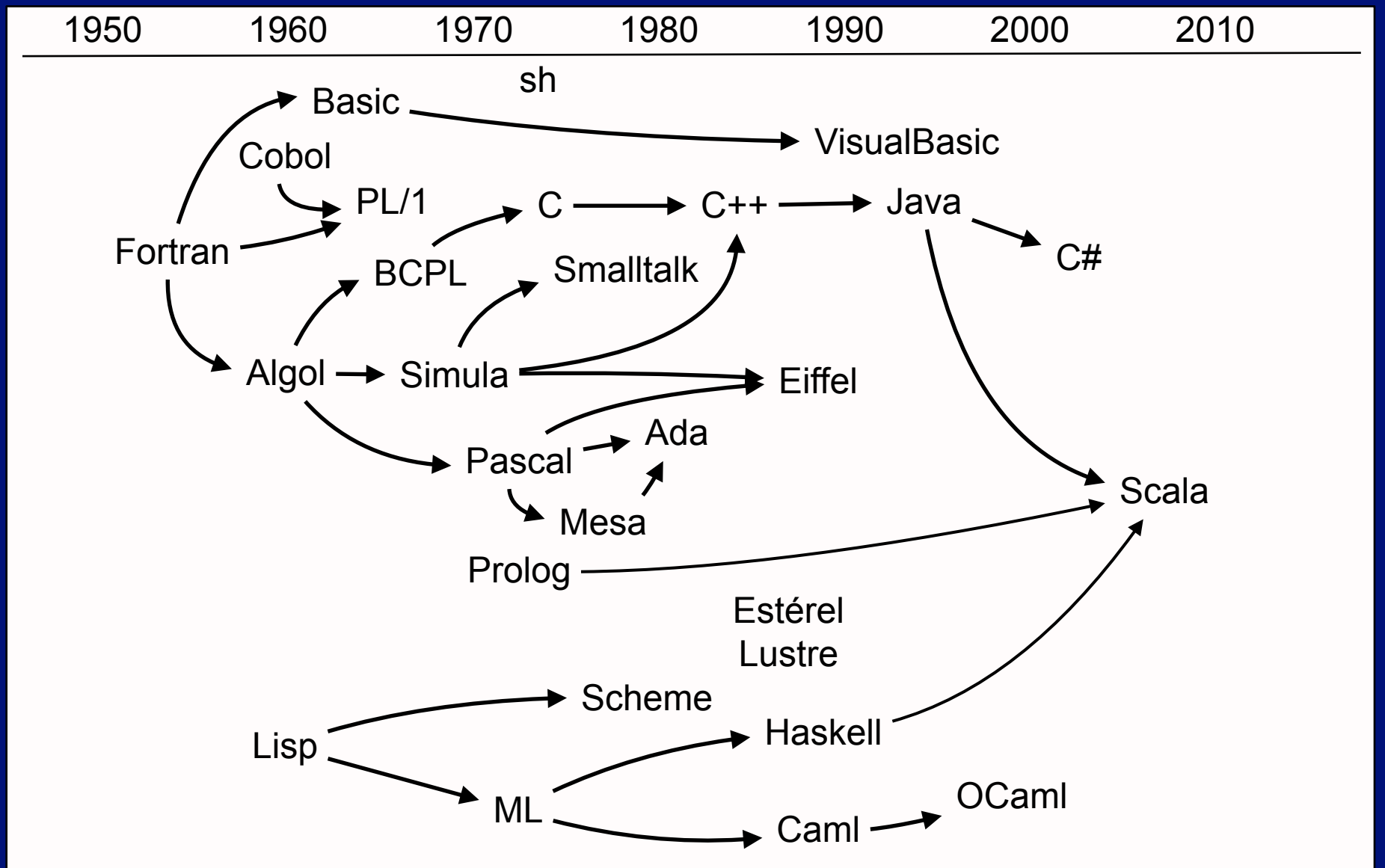
Les langages de programmation

Les langages de programmation

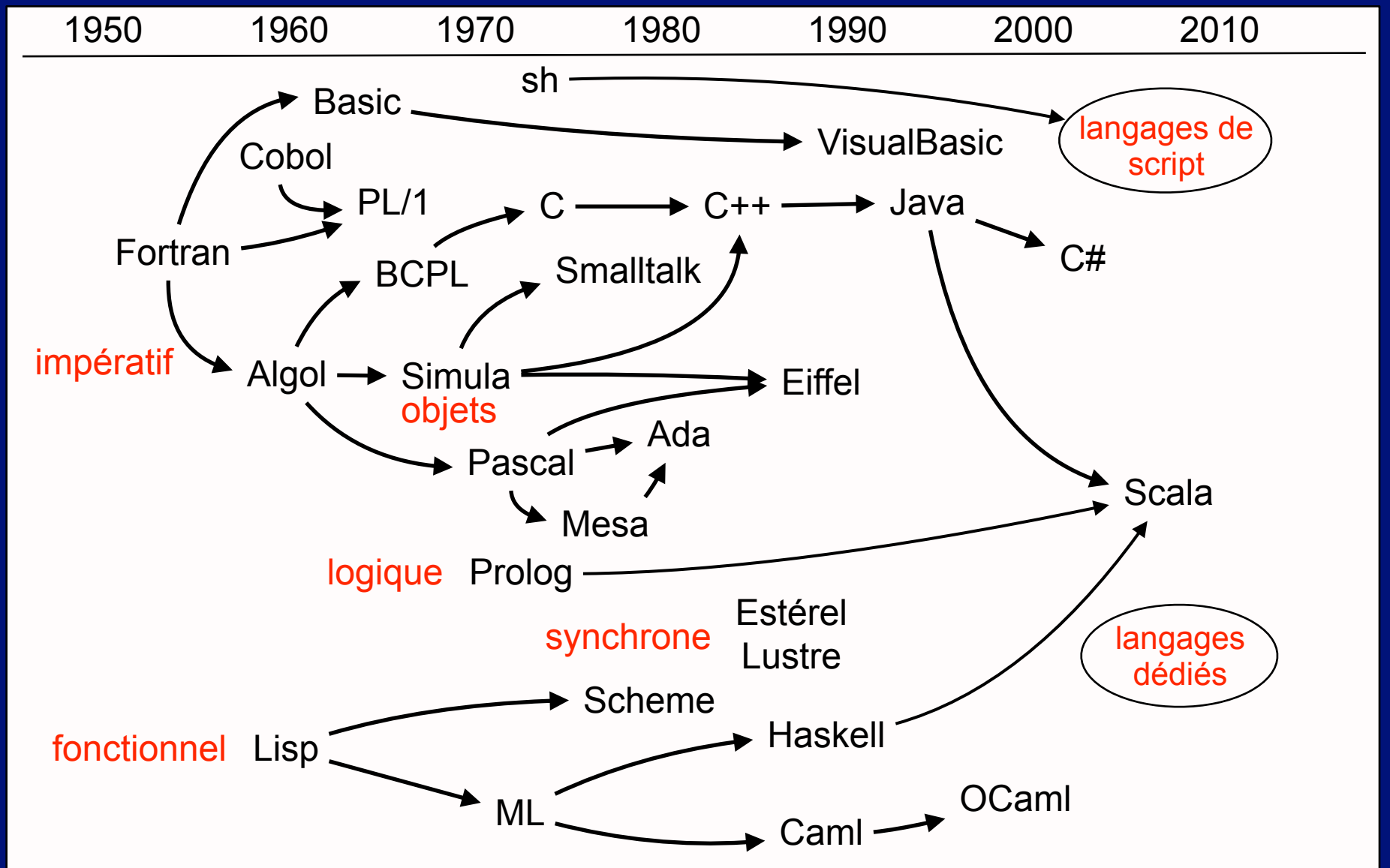


Voir www.oreilly.com/news/graphics/prog_lang_poster.pdf
Plus de 8 000 langages recensés …

Quelques langages de programmation



Quelques langages de programmation



Qu’est-ce qu’un “bon” langage ?

Qu’est-ce qu’un “bon” langage ?

- Un langage sûr

  qui empêche de faire les erreurs les plus courantes

  exemple : ne pas ajouter des pommes et des oranges …

Qu’est-ce qu’un “bon” langage ?

- Un langage sûr

  qui empêche de faire les erreurs les plus courantes

  exemple : ne pas ajouter des pommes et des oranges …

- Un langage expressif

Qu’est-ce qu’un “bon” langage ?

- Un langage sûr
  - qui empêche de faire les erreurs les plus courantes
  - exemple : ne pas ajouter des pommes et des oranges …
- Un langage expressif
- Un langage rigoureux
  - au minimum : une “sémantique” bien définie (on sait précisément ce qu’on fait)
  - idéalement : on peut prouver que le programme fait bien ce qu’on veut qu’il fasse

Qu'est-ce qu'un "bon" langage ?

- Un langage sûr
  - qui empêche de faire les erreurs les plus courantes
  - exemple : ne pas ajouter des pommes et des oranges …
- Un langage expressif
- Un langage rigoureux
  - au minimum : une "sémantique" bien définie (on sait précisément ce qu'on fait)
  - idéalement : on peut prouver que le programme fait bien ce qu'on veut qu'il fasse
- Un langage élégant et lisible
  - un programme est fait autant pour être lu (et compris) que pour être exécuté

Le logiciel : comment çà marche ?

Le logiciel : comment çà marche ?

Problème : combler l'écart entre le programme en langage de haut niveau et le code binaire exécutable de la machine

```
…
if (argc != 2) {
    fprintf(stderr, "usage:
    %s <port>\n", argv[0]);
    exit(1);}
…
```

Machine

exécute du code binaire

Le logiciel : comment çà marche ?

Problème : combler l'écart entre le programme en langage de haut niveau et le code binaire exécutable de la machine

```
…
if (argc != 2) {
    fprintf(stderr, "usage:
    %s <port>\n", argv[0]);
    exit(1);}
…
```

Compilateur
(traduit le programme)

Machine
exécute du code binaire

Le logiciel : comment çà marche ?

Problème : combler l'écart entre le programme en langage de haut niveau et le code binaire exécutable de la machine

```
…
if (argc != 2) {
    fprintf(stderr, "usage:
    %s <port>\n", argv[0]);
    exit(1);}
…
```

On "descend" le programme au niveau de la machine

Compilateur
(traduit le programme)

```
00101100011101010010110110101000011101011010011101010001010000011011001110101101101011011110011111101011 …
```

Machine
exécute du code binaire

Le logiciel : comment çà marche ?

Problème : combler l'écart entre le programme en langage de haut niveau et le code binaire exécutable de la machine

```
…
if (argc != 2) {
    fprintf(stderr, "usage:
    %s <port>\n", argv[0]);
    exit(1);}
…
```

On “descend” le programme au niveau de la machine

Compilateur
(traduit le programme)

```
00101100011101010010110110101
10000111010110100111010100011
01000001101100111010110110101
10110011111101011 …
```

Machine
exécute du code binaire

```
…
if (argc != 2) {
    fprintf(stderr, "usage:
    %s <port>\n", argv[0]);
    exit(1);}
…
```

Machine

Le logiciel : comment çà marche ?

Problème : combler l'écart entre le programme en langage de haut niveau et le code binaire exécutable de la machine

```
…
if (argc != 2) {
    fprintf(stderr, "usage:
    %s <port>\n", argv[0]);
    exit(1);}
…
```

Compilateur
(traduit le programme)

```
0010110001110101001011011010
1000011101011010011101010001
0100000110110011101011011010
110110011111101011 …
```

Machine
exécute du code binaire

On "descend" le programme au niveau de la machine

On "remonte" la machine au niveau du programme

```
…
if (argc != 2) {
    fprintf(stderr, "usage:
    %s <port>\n", argv[0]);
    exit(1);}
…
```

Machine virtuelle
(exécute le programme)

Le logiciel : comment çà marche ?

Problème : combler l'écart entre le programme en langage de haut niveau et le code binaire exécutable de la machine

```
…
if (argc != 2) {
    fprintf(stderr, "usage:
    %s <port>\n", argv[0]);
    exit(1);}
…
```

Compilateur
(traduit le programme)

On "descend" le programme au niveau de la machine

```
001011000111010100101101101
100001110101101001110101000
010000011011001110101101101
110110011111101011 …
```

Machine
exécute du code binaire

Machine virtuelle

```
…
if (argc != 2) {
    fprintf(stderr, "usage:
    %s <port>\n", argv[0]);
    exit(1);}
…
```

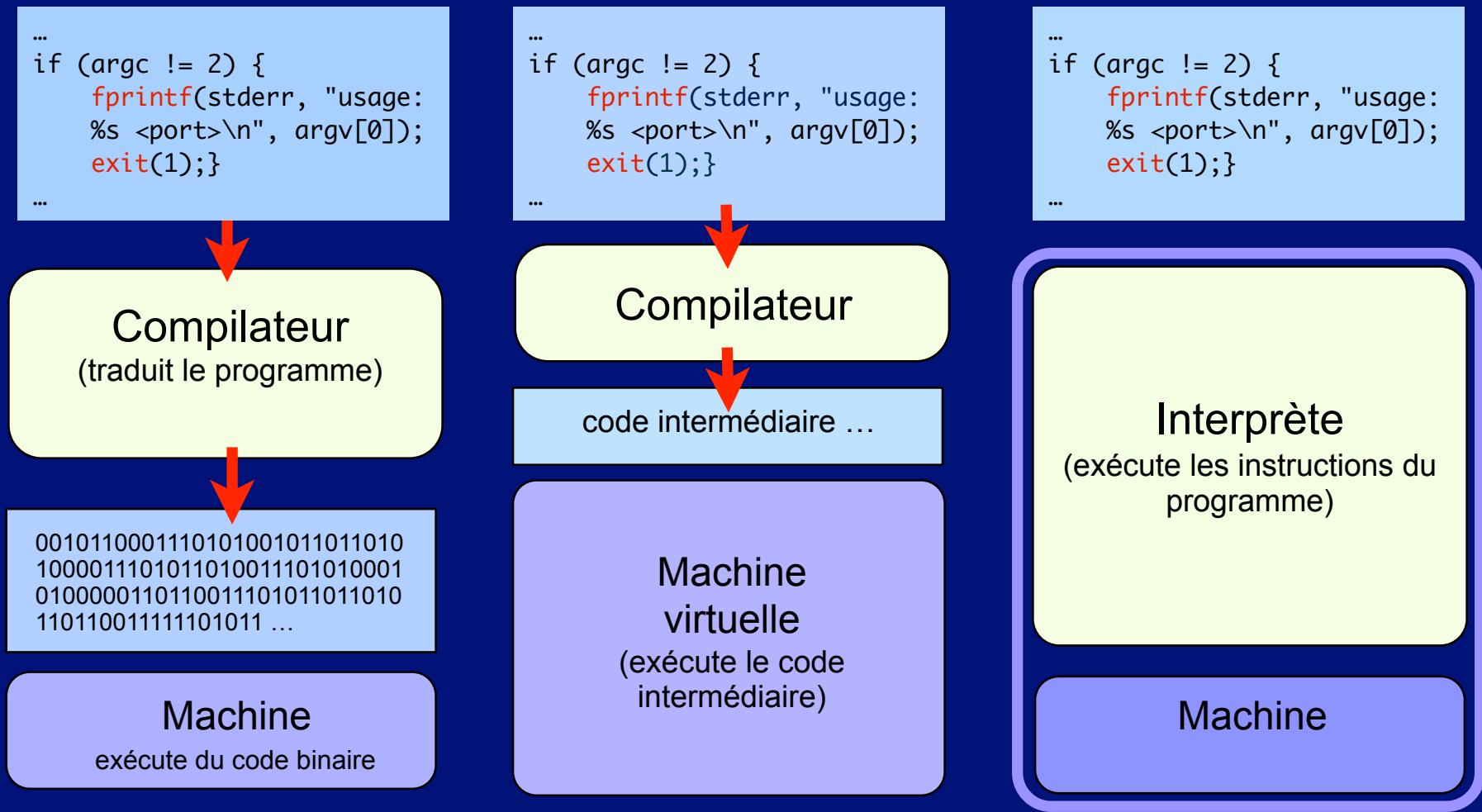
Interprète
(exécute les instructions du programme)

On "remonte" la machine au niveau du programme

Machine

Le logiciel : comment çà marche ?

Problème : combler l'écart entre le programme en langage de haut niveau et le code binaire exécutable de la machine



Le logiciel et l'Internet

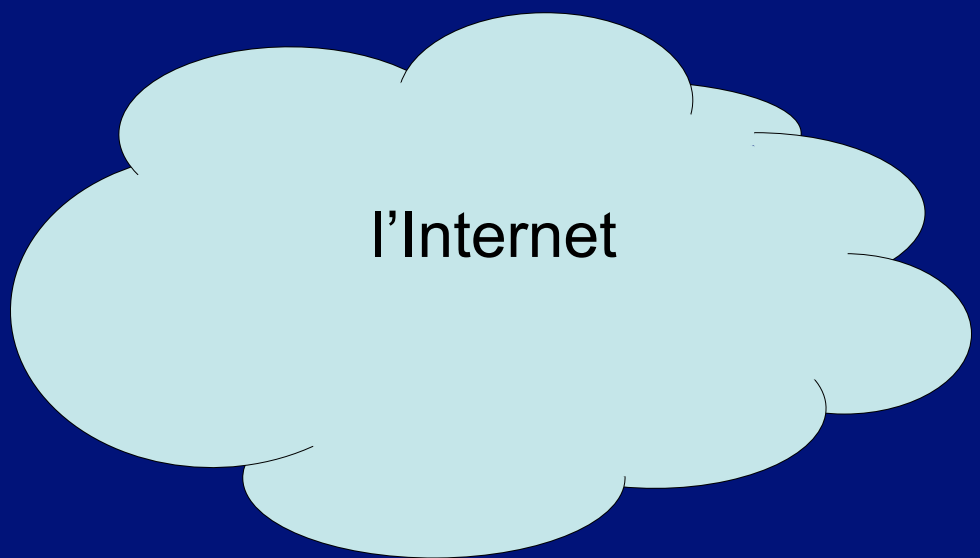
Le logiciel et l'Internet

- Le logiciel est essentiel
  - pour l'utilisation de l'Internet (clients, serveurs)
  - et pour son fonctionnement interne (protocoles, routeurs, annuaires)

Le logiciel et l'Internet

Le logiciel est essentiel

- pour l'utilisation de l'Internet (clients, serveurs)
- et pour son fonctionnement interne (protocoles, routeurs, annuaires)

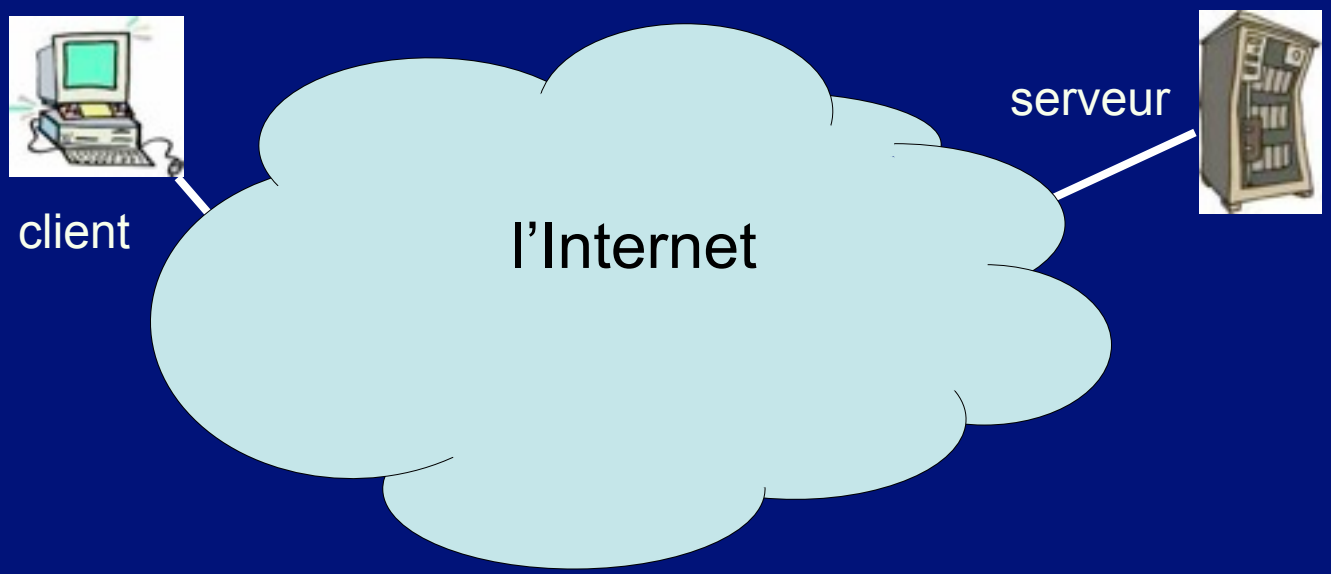


Le logiciel et l'Internet

Le logiciel est essentiel

pour l'utilisation de l'Internet (clients, serveurs)

et pour son fonctionnement interne (protocoles, routeurs, annuaires)

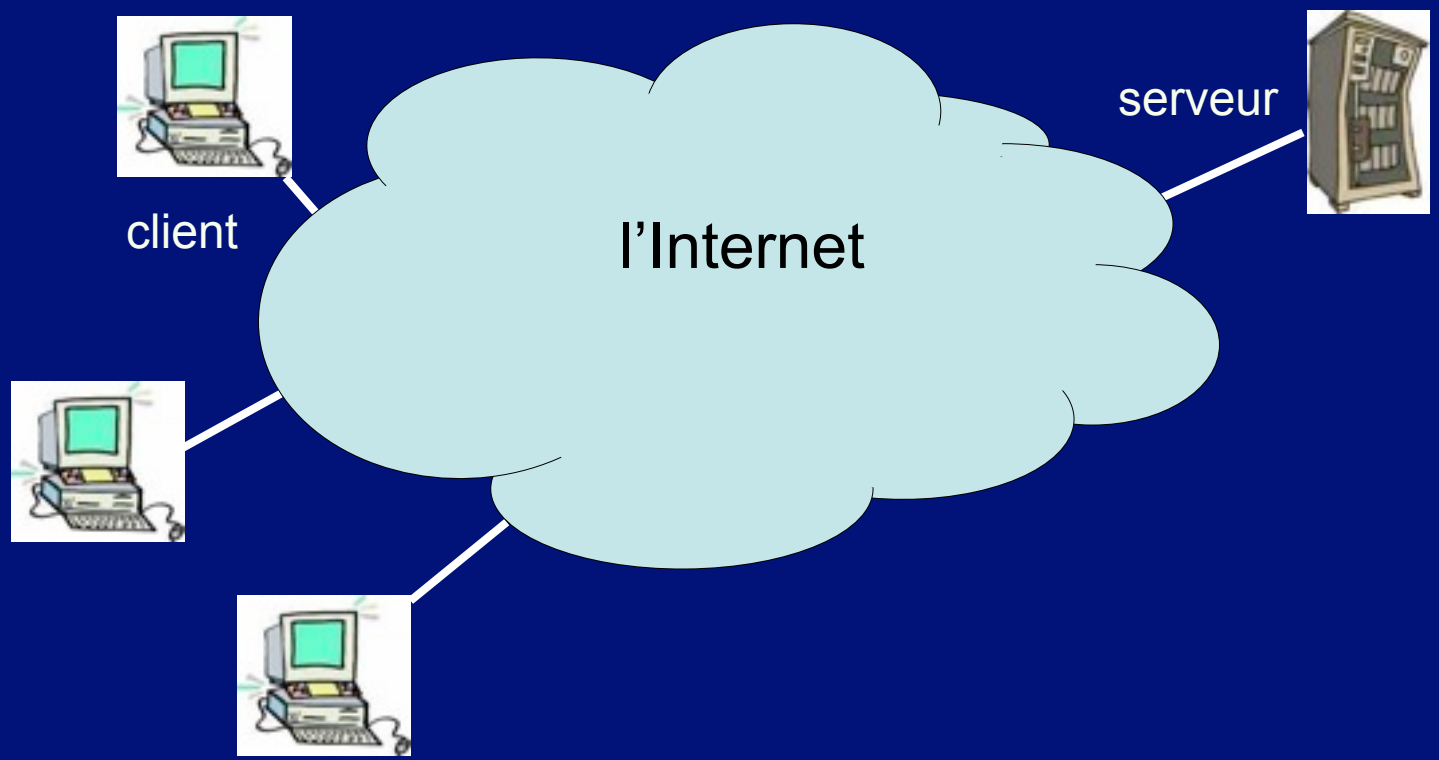


Le logiciel et l'Internet

Le logiciel est essentiel

pour l'utilisation de l'Internet (clients, serveurs)

et pour son fonctionnement interne (protocoles, routeurs, annuaires)

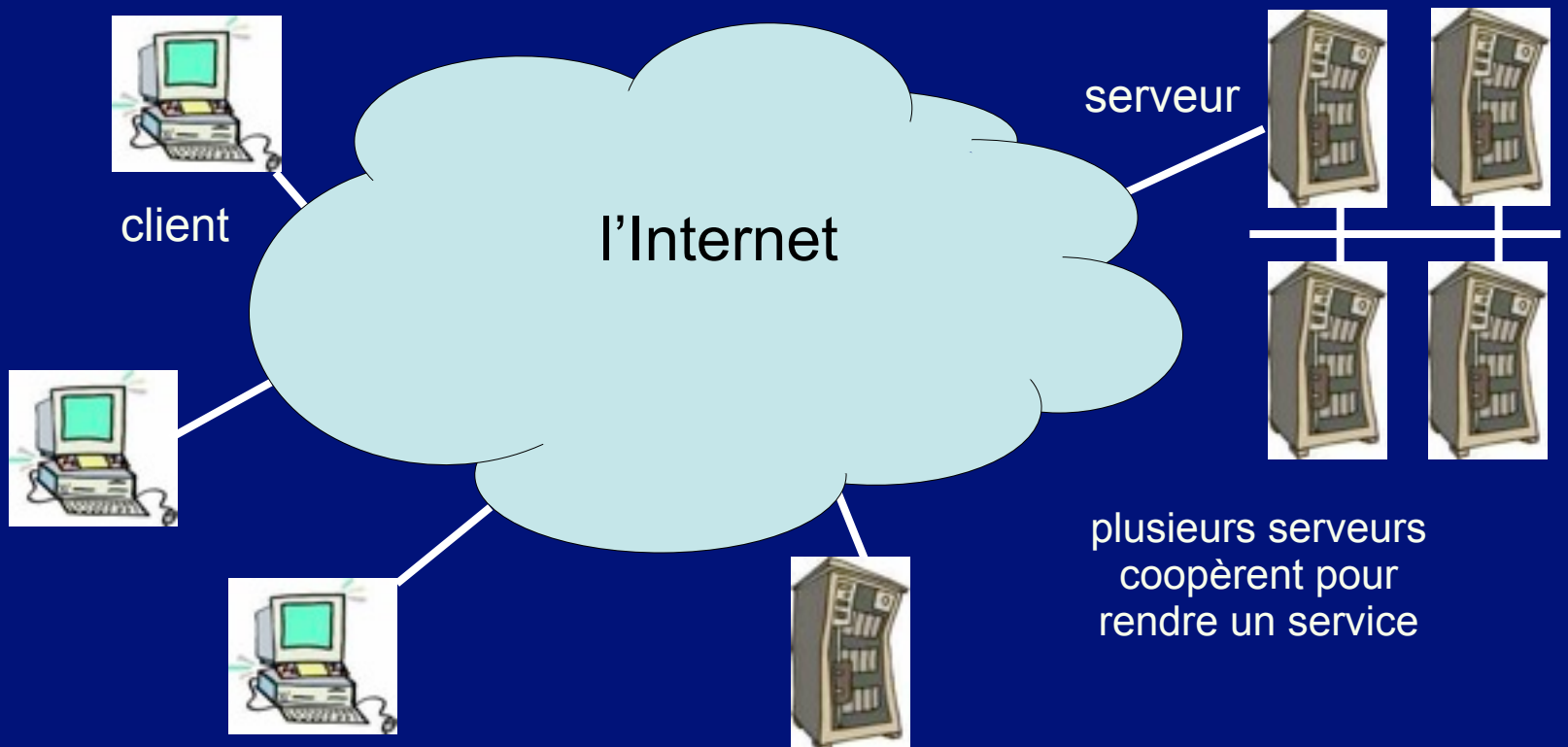


Le logiciel et l'Internet

Le logiciel est essentiel

pour l'utilisation de l'Internet (clients, serveurs)

et pour son fonctionnement interne (protocoles, routeurs, annuaires)

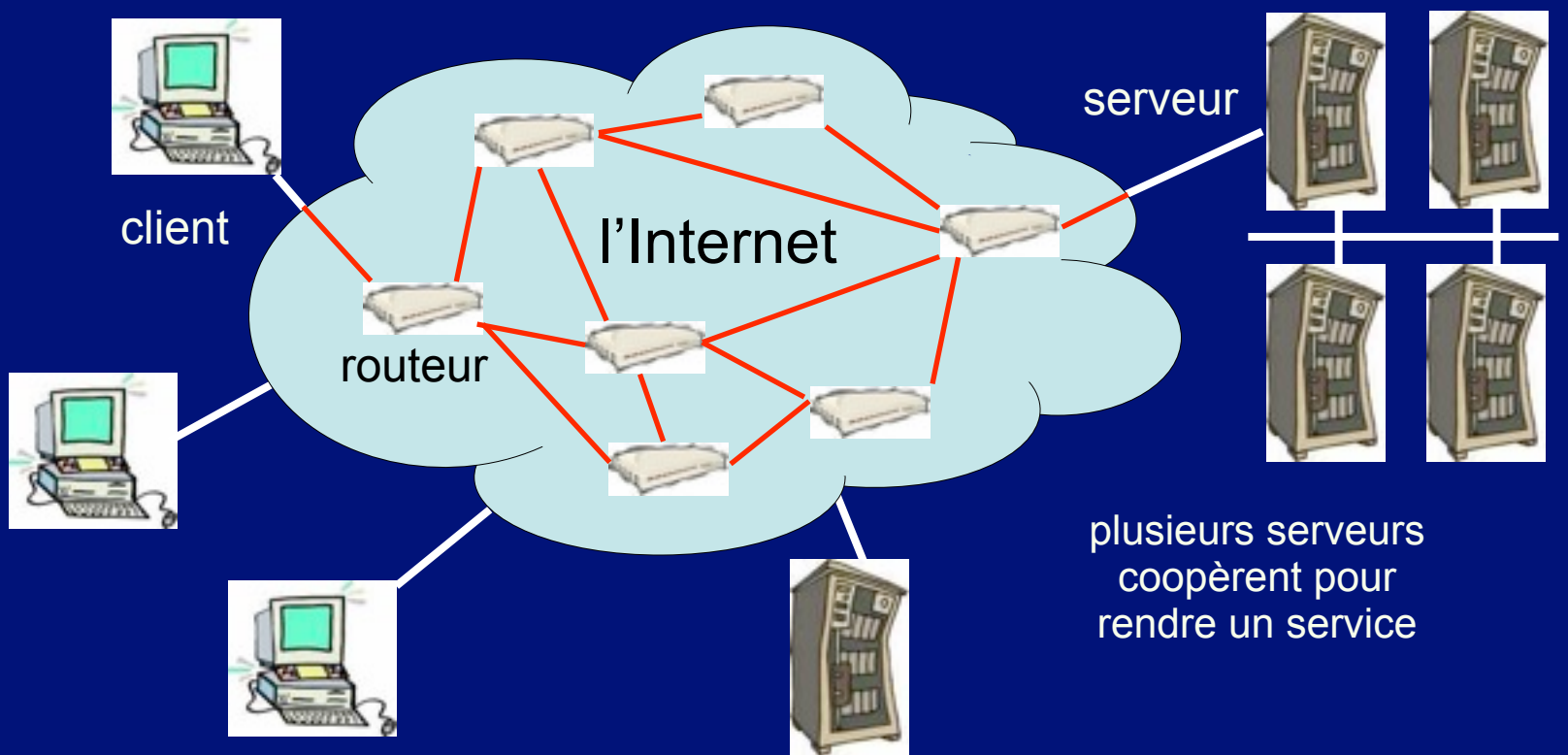


Le logiciel et l'Internet

Le logiciel est essentiel

pour l'utilisation de l'Internet (clients, serveurs)

et pour son fonctionnement interne (protocoles, routeurs, annuaires)



Les *bugs* (bogues)

Les *bugs* (bogues)

- Il n'y a pas de *bugs* informatiques, il n'y a que des erreurs humaines

> "Les programmeurs parlent de *bugs* pour préserver leur santé mentale. Reconnaître un pareil nombre d'erreurs serait psychologiquement insupportable."
>
> Martin Hopkins, 1968

Les *bugs* (bogues)

- Il n'y a pas de *bugs* informatiques, il n'y a que des erreurs humaines

  "Les programmeurs parlent de *bugs* pour préserver leur santé mentale. Reconnaître un pareil nombre d'erreurs serait psychologiquement insupportable."

  Martin Hopkins, 1968

- Quelques *bugs* (tristement) célèbres
  - Les accidents d'irradiation de Therac 25 (1985-87)
  - Le ver de Morris (exploitation malveillante d'un *bug* d'Unix, 1988)
  - La panne du réseau téléphonique ATT (1990)
  - La perte d'Ariane 5 lors de son premier vol (1996)
  - La perte de la sonde Mars Climate Orbiter (1999)
  - La panne d'électricité au Nord-Est des États-Unis (2003)

La vie du logiciel

La vie du logiciel

- La production n'est qu'un début …

La vie du logiciel

- La production n’est qu’un début …
- L’administration
  - Installer, adapter, configurer
  - Science et “magie noire”

La vie du logiciel

- La production n'est qu'un début …
- L'administration
  - Installer, adapter, configurer
  - Science et "magie noire"
- La maintenance
  - Souvent plus de 50% du coût du logiciel …
  - Quand le provisoire devient permanent
  - Le poids de l'existant

La vie du logiciel

- La production n'est qu'un début …
- L'administration
  - Installer, adapter, configurer
  - Science et "magie noire"
- La maintenance
  - Souvent plus de 50% du coût du logiciel …
  - Quand le provisoire devient permanent
  - Le poids de l'existant
- Le logiciel, produit immatériel
  - Le coût de reproduction est voisin de zéro
  - La mise à jour se fait sur place (sans rappel)

La vie du logiciel

- La production n'est qu'un début …
- L'administration
  - Installer, adapter, configurer
  - Science et "magie noire"
- La maintenance
  - Souvent plus de 50% du coût du logiciel …
  - Quand le provisoire devient permanent
  - Le poids de l'existant
- Le logiciel, produit immatériel
  - Le coût de reproduction est voisin de zéro
  - La mise à jour se fait sur place (sans rappel)

La *capacité d'évolution* est un enjeu capital !

Les acteurs du logiciel

Les acteurs du logiciel

- Les constructeurs de matériel
- Les éditeurs de logiciel
- Les services informatiques des entreprises
- Les sociétés de service
- Les fournisseurs de services sur l'Internet
- Les organismes de normalisation
- Les chercheurs
- Les "individuels"

Les enjeux économiques du logiciel

Les enjeux économiques du logiciel

- L'industrie du logiciel
  - Un secteur en pleine expansion
  - Grands groupes et PME innovantes
  - Forte demande de personnel qualifié
    - importance de la formation

Les enjeux économiques du logiciel

- L'industrie du logiciel
  - Un secteur en pleine expansion
  - Grands groupes et PME innovantes
  - Forte demande de personnel qualifié
    - importance de la formation
- Une industrie qui irrigue toutes les autres
  - Part croissante du logiciel dans le coût des produits

Les enjeux économiques du logiciel

- L'industrie du logiciel
  - Un secteur en pleine expansion
  - Grands groupes et PME innovantes
  - Forte demande de personnel qualifié
    - importance de la formation
- Une industrie qui irrigue toutes les autres
  - Part croissante du logiciel dans le coût des produits
- Une industrie en prise directe sur la recherche
  - Durée indicative du transfert idée-produit
    - dans les années 1970 : 12-15 ans (Unix)
    - dans les années 1995-2000 : 2-3 ans (Google)

Les enjeux économiques du logiciel

- L'industrie du logiciel
  - Un secteur en pleine expansion
  - Grands groupes et PME innovantes
  - Forte demande de personnel qualifié
    - importance de la formation
- Une industrie qui irrigue toutes les autres
  - Part croissante du logiciel dans le coût des produits
- Une industrie en prise directe sur la recherche
  - Durée indicative du transfert idée-produit
    - dans les années 1970 : 12-15 ans (Unix)
    - dans les années 1995-2000 : 2-3 ans (Google)

25% de la croissance mondiale

30% des offres d'emploi cadres

30% du coût d'un avion

29% de la R&D mondiale

Un peu d’histoire

Un peu d'histoire

- Les années 40
  - les origines : programmation par tableau de connexions
  - la naissance du programme enregistré

Un peu d'histoire

- Les années 40
  - les origines : programmation par tableau de connexions
  - la naissance du programme enregistré
- Les années 50
  - les premiers langages de programmation
  - les premiers systèmes d'exploitation (par lots)

Un peu d'histoire

- Les années 40
  - les origines : programmation par tableau de connexions
  - la naissance du programme enregistré
- Les années 50
  - les premiers langages de programmation
  - les premiers systèmes d'exploitation (par lots)
- Les années 60
  - nouveaux langages
  - premiers systèmes interactifs en temps partagé
  - 1968 : naissance du "génie logiciel"
  - 1969 : naissance d'Arpanet (futur Internet)

Un peu d'histoire (2)

Un peu d'histoire (2)

- Les années 70
  - développement du génie logiciel
  - les microprocesseurs
  - les bases de données, les transactions

Un peu d'histoire (2)

- Les années 70
  - développement du génie logiciel
  - les microprocesseurs
  - les bases de données, les transactions
- Les années 80
  - développement de l'Internet
  - les "stations de travail" en réseau
  - les ordinateurs individuels

Un peu d'histoire (2)

- Les années 70
  - développement du génie logiciel
  - les microprocesseurs
  - les bases de données, les transactions
- Les années 80
  - développement de l'Internet
  - les "stations de travail" en réseau
  - les ordinateurs individuels
- Les années 90
  - les applications réparties
  - 1995 : le World Wide Web et l'Internet grand public
  - les "systèmes embarqués"
  - le logiciel libre

Un peu d’histoire (3)

Un peu d'histoire (3)

- Les années 2000
  - l'Internet (presque) partout
  - les applications multimédia
  - les réseaux sociaux
  - la bioinformatique

Un peu d'histoire (3)

- Les années 2000
  - l'Internet (presque) partout
  - les applications multimédia
  - les réseaux sociaux
  - la bioinformatique
- 2010 et la suite …
  - quelques défis
    - informatique et société
    - sécurité, protection
    - systèmes sûrs et certifiés
    - nouvelles techniques de calcul
    - …

Les défis du logiciel

Les défis du logiciel

- Faire des logiciels corrects
  - Correct : qui fait ce qu'on veut qu'il fasse
  - Condition préalable : bien définir ce qu'on veut …
    - … pas si facile !

Les défis du logiciel

- Faire des logiciels corrects
  - Correct : qui fait ce qu'on veut qu'il fasse
  - Condition préalable : bien définir ce qu'on veut …
    - … pas si facile !
- Faire des logiciels efficaces
  - Critères d'efficacité
    - Vitesse, place en mémoire, communication, énergie …

Les défis du logiciel

- Faire des logiciels corrects
  - Correct : qui fait ce qu'on veut qu'il fasse
  - Condition préalable : bien définir ce qu'on veut …
    - … pas si facile !
- Faire des logiciels efficaces
  - Critères d'efficacité
    - Vitesse, place en mémoire, communication, énergie …
- Faire des logiciels sûrs
  - Sûr : qui résiste aux événements indésirables
    - Panne, surcharge, attaque, …

Les défis du logiciel

- Faire des logiciels corrects
  - Correct : qui fait ce qu'on veut qu'il fasse
  - Condition préalable : bien définir ce qu'on veut …
    - … pas si facile !
- Faire des logiciels efficaces
  - Critères d'efficacité
    - Vitesse, place en mémoire, communication, énergie …
- Faire des logiciels sûrs
  - Sûr : qui résiste aux événements indésirables
    - Panne, surcharge, attaque, …
- Faire des logiciels conviviaux
  - Qui prennent en compte les besoins des humains

Faire des logiciels corrects

Faire des logiciels corrects

- Établir une spécification
  - Complète, non ambiguë, non contradictoire, …

Faire des logiciels corrects

- Établir une spécification
  - Complète, non ambiguë, non contradictoire, …
- Assurer que le programme est conforme à la spécification
  - Test
    - Le problème du “taux de couverture”
  - Vérification
    - Un “test *complet* sans exécution”
    - Le choix du bon modèle
    - le problème de la taille
  - Preuve
    - Une logique pour l’informatique : construction = preuve
    - Des avancées spectaculaires !

Faire des logiciels efficaces

Faire des logiciels efficaces

- Réduire le temps d'exécution

Faire des logiciels efficaces

- Réduire le temps d'exécution
  - Diminuer le nombre d'opérations
    - par une méthode plus efficace
    - par un codage optimisé

Faire des logiciels efficaces

- Réduire le temps d'exécution
  - Diminuer le nombre d'opérations
    - par une méthode plus efficace
    - par un codage optimisé
  - Augmenter la vitesse des opérations
    - processeurs plus rapides, mais on atteint les limites

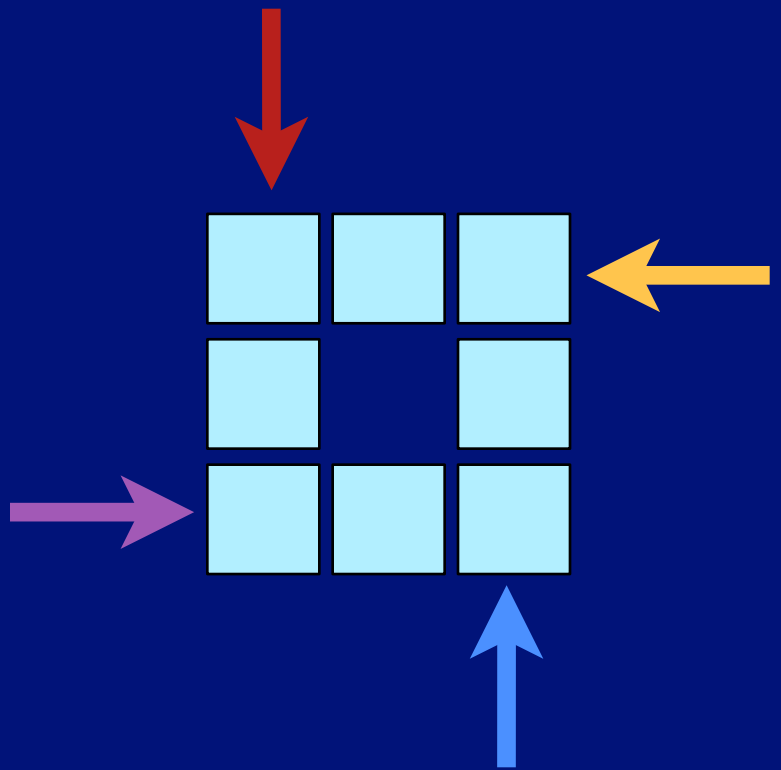
Faire des logiciels efficaces

- Réduire le temps d'exécution
  - Diminuer le nombre d'opérations
    - par une méthode plus efficace
    - par un codage optimisé
  - Augmenter la vitesse des opérations
    - processeurs plus rapides, mais on atteint les limites
  - Exécuter des opérations en parallèle
    - processeurs multi-cœurs
    - grappes et grilles

Faire des logiciels efficaces

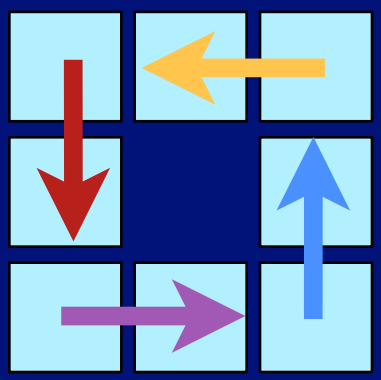
- Réduire le temps d'exécution
  - Diminuer le nombre d'opérations
    - par une méthode plus efficace
    - par un codage optimisé
  - Augmenter la vitesse des opérations
    - processeurs plus rapides, mais on atteint les limites
  - Exécuter des opérations en parallèle
    - processeurs multi-cœurs
    - grappes et grilles
- Réduire la consommation d'énergie
  - Une préoccupation croissante …
  - Des réglages de plus en plus fins

Les pièges du parallélisme



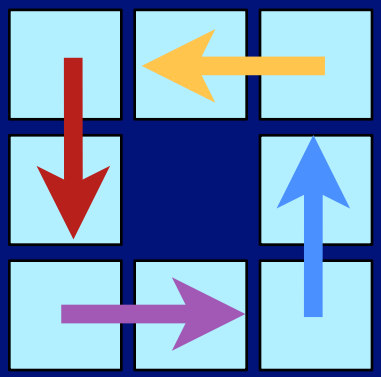
4 applications (ou parties)
Chacune a besoin de 3 blocs de mémoire

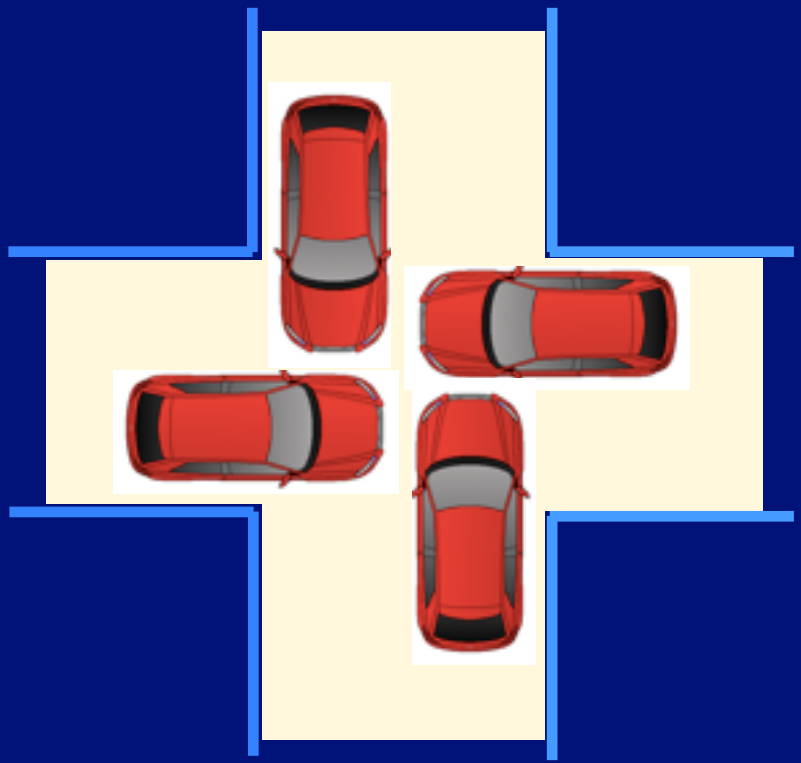
Les pièges du parallélisme



4 applications (ou parties)
Chacune a besoin de 3 blocs de mémoire

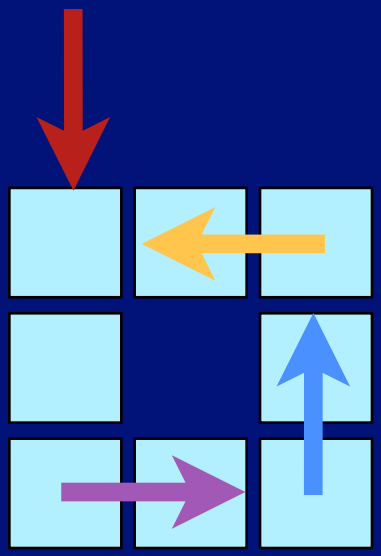
Les pièges du parallélisme

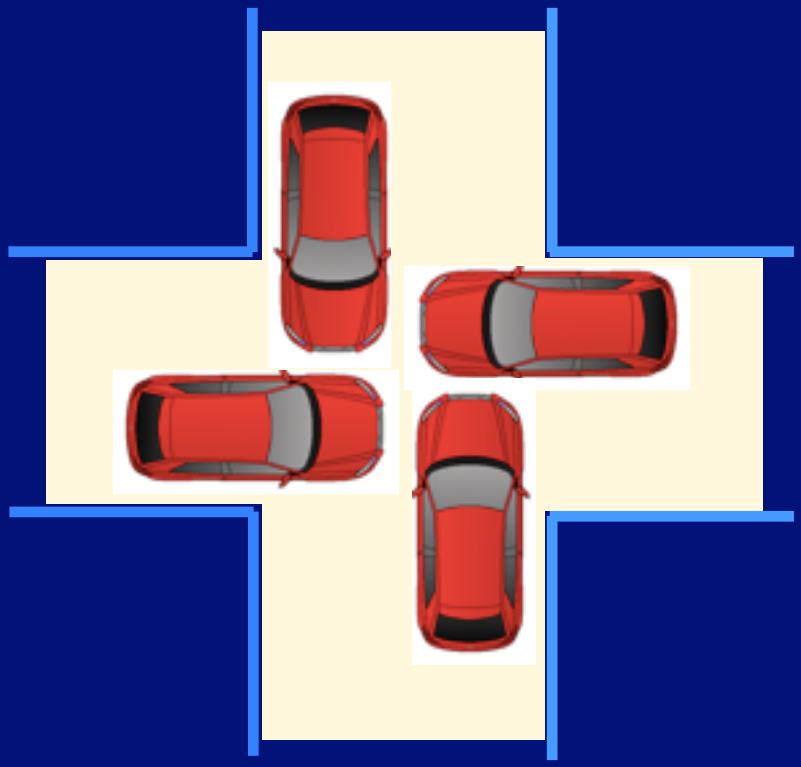




4 applications (ou parties)
Chacune a besoin de 3 blocs de mémoire

Les pièges du parallélisme





4 applications (ou parties)
Chacune a besoin de 3 blocs de mémoire

Faire des logiciels sûrs

Faire des logiciels sûrs

- Tolérer les fautes
  - Tolérer, pas éliminer (mission impossible …)
  - Un seul moyen, la redondance …
    - mais les pièges sont nombreux

Faire des logiciels sûrs

- Tolérer les fautes
  - Tolérer, pas éliminer (mission impossible …)
  - Un seul moyen, la redondance …
    - mais les pièges sont nombreux
- Résister aux pics de charge
  - Par l'adaptation

Faire des logiciels sûrs

- Tolérer les fautes
  - Tolérer, pas éliminer (mission impossible …)
  - Un seul moyen, la redondance …
    - mais les pièges sont nombreux
- Résister aux pics de charge
  - Par l'adaptation
- Survivre aux attaques
  - Prévention
  - Détection

Faire des logiciels sûrs

- Tolérer les fautes
  - Tolérer, pas éliminer (mission impossible …)
  - Un seul moyen, la redondance …
    - mais les pièges sont nombreux
- Résister aux pics de charge
  - Par l'adaptation
- Survivre aux attaques
  - Prévention
  - Détection

Vers des systèmes autonomes

Faire des logiciels conviviaux

Faire des logiciels conviviaux

- Faciliter l'utilisation
  - Quelques principes d'ergonomie
    - Se mettre à la place de l'usager
    - Logique, cohérence, économie
    - Pas de surprise !
    - Pas d'intrusion abusive

Faire des logiciels conviviaux

- Faciliter l'utilisation
  - Quelques principes d'ergonomie
    - Se mettre à la place de l'usager
    - Logique, cohérence, économie
    - Pas de surprise !
    - Pas d'intrusion abusive
- Faciliter la maintenance
  - Faire des systèmes compréhensibles
    - Organisation, documentation
    - Programmation "lettrée" (à l'usage des humains)

Faire des logiciels conviviaux

- Faciliter l'utilisation
  - Quelques principes d'ergonomie
    - Se mettre à la place de l'usager
    - Logique, cohérence, économie
    - Pas de surprise !
    - Pas d'intrusion abusive
- Faciliter la maintenance
  - Faire des systèmes compréhensibles
    - Organisation, documentation
    - Programmation "lettrée" (à l'usage des humains)
- Faciliter l'administration
  - Administration autonome

Remarques finales

Remarques finales

- Le logiciel, un “concentré d’intelligence” …
  - La logique et l’esthétique
  - Création individuelle, création collective

Remarques finales

- Le logiciel, un “concentré d’intelligence” …
  - La logique et l’esthétique
  - Création individuelle, création collective
- Le futur du logiciel
  - Le mythe de la “programmation automatique”
  - L’élévation du niveau d’expression
  - L’élévation du degré de confiance

Remarques finales

- Le logiciel, un “concentré d’intelligence” …
  - La logique et l’esthétique
  - Création individuelle, création collective
- Le futur du logiciel
  - Le mythe de la “programmation automatique”
  - L’élévation du niveau d’expression
  - L’élévation du degré de confiance
- La démarche informatique
  - Un élément de la culture générale
  - La nécessaire mutation de l’enseignement

Pour aller plus loin

Pour aller plus loin

- Un journal en ligne
  - Interstices
    - http://interstices.info
  - Introduction vivante et accessible à tous les aspects de l'informatique et des domaines connexes

Pour aller plus loin

- Un journal en ligne
  - Interstices
    - http://interstices.info
    - Introduction vivante et accessible à tous les aspects de l'informatique et des domaines connexes
- Des cours en vidéo
  - Au Collège de France
    - Gérard Berry : *Pourquoi et comment le monde devient numérique*
    - http://www.college-de-france.fr/default/EN/all/inn_tec2007/

Merci
de votre attention