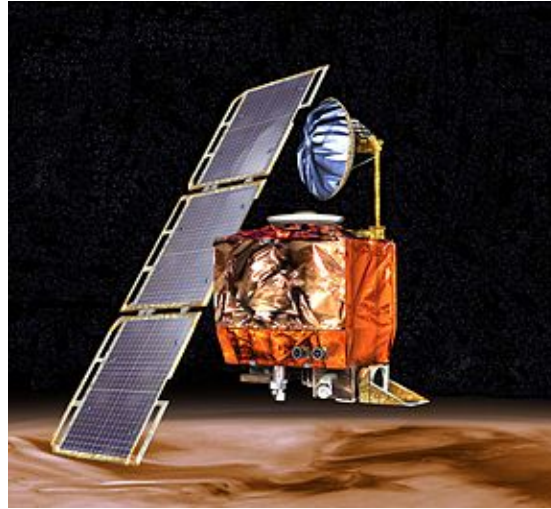


Techniques de Programmation

Introduction

Nicolas Belloir

nicolas.belloir@univ-pau.fr



Prenons quelques exemples

Ariane 5 – Vol 501

- Vidéo
 - <http://youtu.be/kYUrqdUyEpl>



Ariane 5 – Vol 501

- Accident
 - H0+36s : défaillance du système de référence inertielle (SRI) de secours et presque simultanément du SRI actif
 - braquage des tuyères puis du moteur Vulcain
 - basculement brutal du lanceur
 - autodestruction du lanceur



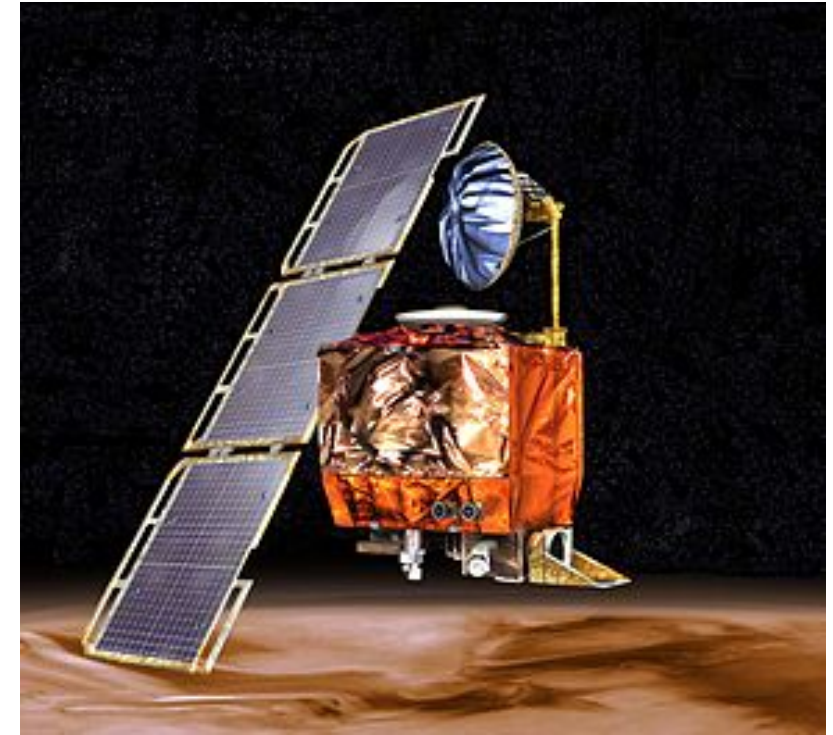
Ariane 5 – Vol 501

- Rapport enquête
 - « *Le braquage des tuyères a été commandé par le logiciel du calculateur de bord agissant sur la base de données transmises par le SRI* ».
- Erreur
 - conversion d'un nombre flottant sur 64 bits en un entier sur 16 bits (ADA).
 - Variables non protégées
- Quel type d'erreur?
 - Erreur de réutilisation [Jezequel, 1997].
 - Simple erreur de « *cast* ».
 - Fonction mal testée.
- Coût : 500 millions de \$



Mars Climate Orbiter

- Explosion en vol le 23/11/1999
- Erreur
 - confusion dans les unités de données
 - modèle anglais : pieds/s Vs modèle métrique : Newtons/s
- Incompréhension des équipes de développement
 - Les ingénieurs de Lockheed Martin Astronautics (Denver, Colorado) -> système anglo-saxon
 - Les ingénieurs du Jet Propulsion Laboratory (Pasadena en Californie) -> Système métrique
- Coût : 260millions de \$



F22 Raptor

- Bug
 - Franchissement de la ligne de changement de date
 - Plantage de presque tous les systèmes :
 - Navigation, Communication, Gestion carburant ...
- Erreur
 - Erreur de programmation classique
 - Correction : moins de 48 heures ...
- Coût potentiel : $6 * 135$ millions de \$



De quoi parle-t'on?

La programmation

Ensemble des activités qui permettent l'écriture des programmes informatiques



```
ed boolean empty(String s) {  
    if (s == null) {  
        return true;  
    } else if (s.length() == 0) {  
        return true;  
    } else {  
        return false;  
    }  
}
```

Certains pensent que ce n'est qu'une science, d'autres qu'il s'agit d'un art, ou encore de pure magie noire...

Où il est question de développement logiciel (1)

- Observons le fonctionnement estudiantin moyen soumis a un problème informatique :
 1. conception et écriture d'un algorithme
 2. programmation du-dit algorithme dans un langage
 3. compilation du programme
 4. exécution du programme sur quelques jeux de données
 5. attente de la note après évaluation



Où il est question de développement logiciel (2)

- Observation de la réalisation d'un logiciel complexe dans le milieu industriel :

1. identification et spécification du problème
2. conception
3. implémentation
4. intégration et test du système
5. installation et test sur le terrain
6. maintenance



A toujours avoir à l'esprit (1)

« Etre un bon créateur de logiciel ce n'est pas seulement être un bon mathématicien ou un expert en langage C ».

Jean-Christophe Arnulfo, Vertigo Engineering

A toujours avoir à l'esprit (2)

«Les meilleurs programmeurs sont ceux qui ont réalisé combien leur cerveau était petit »

Edsger Dijkstra, *prix Turing 1972*

A toujours avoir à l'esprit (2)

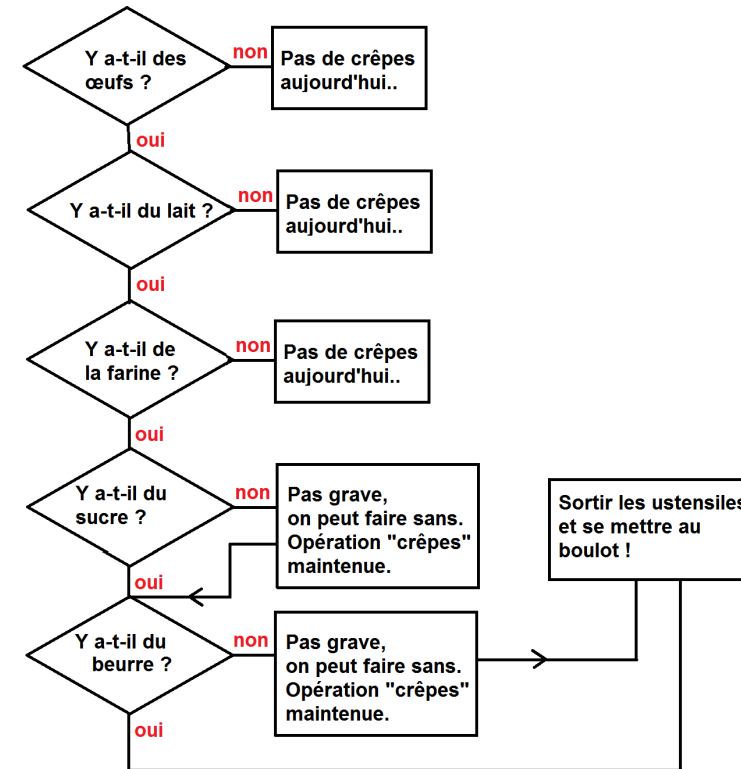
«Celui qui ne comprend pas, et qui le dit est celui qui fait le plus évidemment preuve d'intelligence, car il a compris qu'il n'a pas compris et c'est ce qui est le plus difficile à comprendre. Remercions-le, car il fait un cadeau à tous ceux qui, autour de lui, croyaient, à tort, avoir compris»

Albert Jacquard, *biologiste français*

Un cours de techniques de programmation

Pourquoi un cours de techniques de programmation ?

- Niveau « analyse et conception »
 - définition et spécification du problème
 - conception => **algorithmique**
 - => **On crée du raisonnement**
- Niveau réalisation et livraison
 - Ecriture du code source
 - Compilation
 - Test et corrections
 - Livraison
 - => **On crée de l'exécutable, du réel!**



Avec quels outils

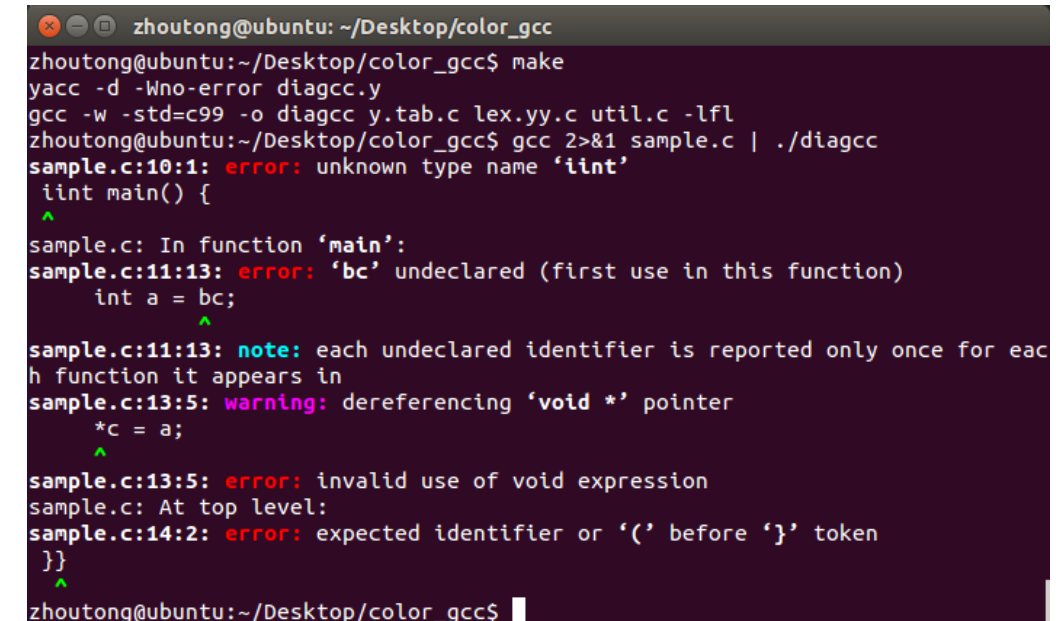
- Ecriture du code source => **Editeur**
- Compilation => **Compilateur / Edition de lien**

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <arpa/inet.h>

void serveur1(portServ ports)
{
    int sockServ1, sockServ2, sockClient;
    struct sockaddr_in monAddr, addrClient, addrServ2;
    socklen_t lenAddrClient;

    if ((sockServ1 = socket(AF_INET, SOCK_STREAM, 0)) == -1) {
        perror("Erreur socket");
        exit(1);
    }
    if ((sockServ2 = socket(AF_INET, SOCK_STREAM, 0)) == -1) {
        perror("Erreur socket");
        exit(1);
    }

    bzero(&monAddr, sizeof(monAddr));
    monAddr.sin_family = AF_INET;
    monAddr.sin_port = htons(ports.port1);
    monAddr.sin_addr.s_addr = INADDR_ANY;
    bzero(&addrServ2, sizeof(addrServ2));
```



The screenshot shows a terminal window with the following commands and output:

```
zhoutong@ubuntu: ~/Desktop/color_gcc
zhoutong@ubuntu:~/Desktop/color_gcc$ make
yacc -d -Wno-error diagcc.y
gcc -w -std=c99 -o diagcc y.tab.c lex.yy.c util.c -lfl
zhoutong@ubuntu:~/Desktop/color_gcc$ gcc -g2 sample.c | ./diagcc
sample.c:10:1: error: unknown type name 'iint'
    iint main() {
    ^
sample.c: In function 'main':
sample.c:11:13: error: 'bc' undeclared (first use in this function)
    int a = bc;
             ^
sample.c:11:13: note: each undeclared identifier is reported only once for each function it appears in
sample.c:13:5: warning: dereferencing 'void *' pointer
    *c = a;
    ^
sample.c:13:5: error: invalid use of void expression
sample.c: At top level:
sample.c:14:2: error: expected identifier or '(' before '}' token
    }}
    ^
zhoutong@ubuntu:~/Desktop/color_gcc$
```

Avec quels outils

- Test et corrections => **Débogueur**

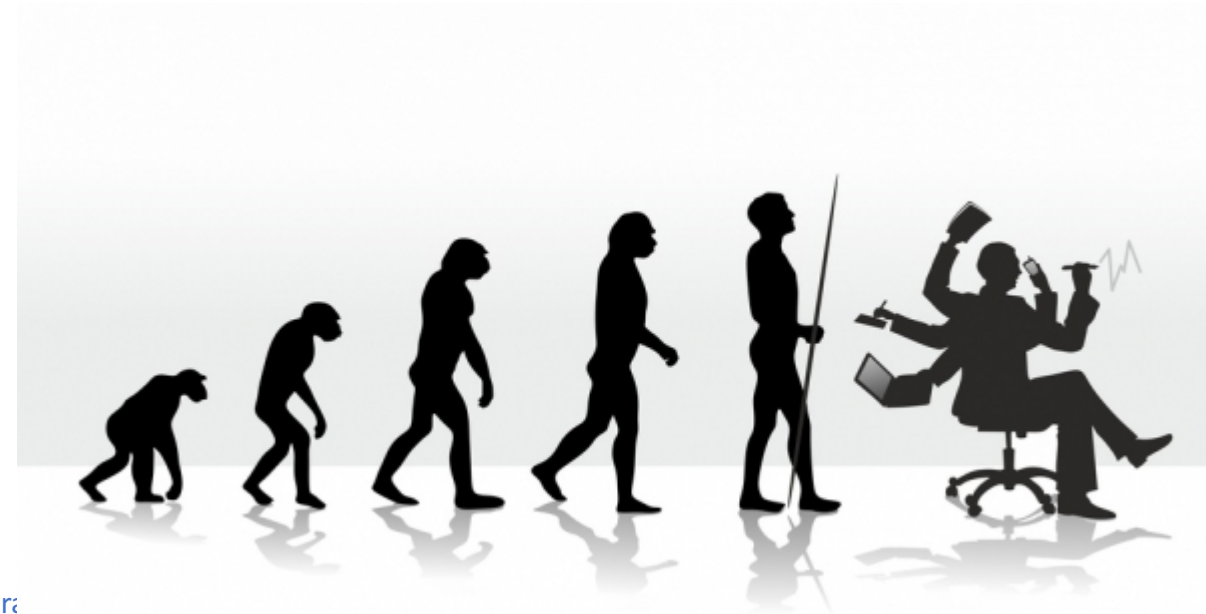
```
63     }
64     mch_errmsg("\nusage:");
65     for (i = 0; ; ++i)
66     {
67         mch_errmsg(" vim [options] ");
68         mch_errmsg((char *)use[i]);
69         if (i == (sizeof(use) / sizeof(char_u *)) - 1)
70             break;
71         mch_errmsg("\n  or:");
72     }
73
74     mch_errmsg("\n\nOptions:\n");
75     mch_errmsg("  -\t\t\tEnd of options\n");
76 #ifdef USE_GUI
77     mch_errmsg("  -g\t\t\tRun using GUI (like \"gvim\")\n");
```

- Livraison => Code, binaire et **documentation** (Algorithmes, commentaires, ...)

```
1  /*
2  Ci-dessous, ce sont des directives de préprocesseur.
3  Ces lignes permettent d'ajouter des fichiers au projet, fichiers que l'on appelle "bibliothèques".
4  Grâce à ces bibliothèques, on disposera de fonctions toutes prêtes pour afficher par exemple "Bonjour".
5  */
6
7  #include <stdio.h>
8  #include <stdlib.h>
9
10 /*
11 Ci-dessous, vous avez la fonction principale du programme, appelée "main". C'est par cette fonction que le programme va commencer à s'exécuter.
12 Ici, ma fonction se contente d'afficher "Bonjour" à l'écran.
13 */
14
15 int main()
16 {
17     printf("Bonjour"); // Cette instruction affiche Bonjour à l'écran
18     return 0;          // Le programme renvoie le nombre 0 puis s'arrête
19 }
```

Objectif

- Connaissance des **mécanismes de bonne programmation**
- Connaissance et maîtrise des **outils** permettant l'implémentation, le débogage et l'exécution de programmes
- Aborder la programmation plus **professionnellement**
- Maîtrise du **langage C**



Me joindre

- Contact
 - adresse : Faculté des Sciences, Bât B3, 2e étage
 - mail : nicolas.belloir@univ-pau.fr
- Organisation
 - Bien vérifier sur l'EdT !!!

Contenu du module

- Volume horaire
 - 12h de cours (8*1,5h)
 - 27h de TP (9*3h)
- Programme
 - Outils et techniques pour programmation
 - Un langage de programmation : le langage C
- Evaluation
 - Examen : 70 %
 - Contrôle continue : 30 % :
 - Un projet (binôme aléatoire)
 - Un contrôle de TP
 - Une note de participation?