**Module : Méthodes et Spécifications Formelles**
**(Approche orientée *modèle*)**

**Slide 1**

Christian Attiogbé
UFR Sciences Nantes
Dpt. Informatique

Christian.Attiogbe@univ-nantes.fr
maj. janvier 2006

**Motivations**

**Slide 2**

Niveau MASTER ⇒ Conduite de projets industriels informatiques

Domaines variés, tailles variables (assez grandes tailles)

Projets informatiques grande taille ⇒ Méthodes, Techniques, Outils

– Méthodes d'analyse,
– Méthodes de conception,
– Méthodes de développement.

## Motivations (suite)

Exemples de méthodes

- Analyse fonctionnelle (SADT par exemple),
- Analyse structurée (SA, SSADM), SA-RT (Temps-Réel),

**Slide 3**

- Entités/Associations, Merise, Axiale,
- JSD/JSP,
- Analyse Orientée Objet, OMT, UML,
- Architecture de logicielle,
- etc

$\Rightarrow$ méthodes semi-formelles

## Motivations (suite)

Besoin de méthodes plus rigoureuses pour certains domaines :

- Sécurité, Certification, Coût, Maintenance

**Slide 4**

- ITSEC (Information Technology Security Evaluation Criteria) exigent l'usage de méthodes **formelles**

- Echec (d'un vol) de ARIANE !

– Milieux hostiles à l'homme (nucléaire, chimie, marin, etc)

– Systèmes embarqués (véhicules, équipements, etc)

**Slide 5**

– Automates (domaine médical, etc)

– etc

Les méthodes formelles ⇒

– garantie de **correction des logiciels**,

**Slide 6**

– **diminuent/éliminent les erreurs**, les disfonctionnements,

– **facilitent la maintenance**.

## Méthodes formelles : introduction

**Méthodes de développement des systèmes informatiques.**

Quelques analogies :

**Génie civil**

Slide 7

→ Architecture, plans (conception), calculs, construction (réalisation)

**Physique**

→ Observations, Modélisation, études sur les modèles, réalisation

**Informatique**

⇒

Analyse des besoins (observations ?)

Slide 8

Modélisation,

études des modèles,

réalisation du système

Différentes **approches d'utilisation des méthodes** formelles :

– **à postériori :** On développe (programmation) puis on vérifie que le produit est correct
  → Systèmes de preuve, systèmes de test

**Slide 9**

– **à priori :** On développe correctement le produit
  → Méthodes de développement (raffinement, synthèse), Systèmes de preuve

Plusieurs méthodes formelles (langages, systèmes de preuve, méthodes)

⌟

– **Approche** TOP-DOWN **(descendant)**
  On procède par **décomposition**
  – Analyse globale (étude système, ingénierie de système)
  – Architecture de logiciel

**Slide 10**

  ↓

  – Codage des composants
    – Programmation ou
    – Développement formel

⌟

**Slide 11**

– **Approche** BOTTOM-UP **(ascendant)**
On procède par **composition** de composants élémentaires.
– Etude des composants disponibles
– Composition, Réutilisation

**Slide 12**

**Eléments d'architecture de logiciel**

– **Composants** : entités logicielles élémentaires

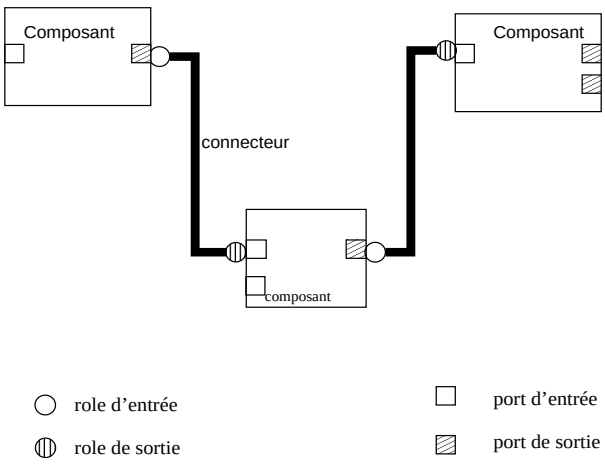
– **Connecteurs** : mécanismes pour relier les composants

– **Configurations** : topologies des composants et des connecteurs

**Slide 13**

- **Interfaces** (associés aux composants et connecteurs)
  - **ports (interface d'un composant)** : définissent les points de communication d'un composant avec son environnement
  - **rôles (interface d'un connecteur)** : identifient les participants à une interaction
- **Attaches** : définissent les équivalence entre deux points d'interface.

**Slide 14**



**Eléments d'architecture de logiciel (suite)**

– Environnements centralisés

COTS, BEANS, EJB, etc,

**Slide 15**

– Environnements distribués

CORBA, bus logiciel sur le réseau, composants, langage IDL

– Environnements mobiles

⌟

**Besoin des méthodes formelles**

Dans tous les cas (approches) recours au méthodes formelles pour

– Etude des systèmes

**Slide 16**

– Etude des composants

– Cadre formel pour le raisonnement, analyse, développement

⌟

**Slide 17**

## Qu'y a-t-il dans les méthodes formelles ?

– Logique
– Algèbre
– Mathématiques discrètes
– Théorie des ensembles
– Théorie des automates
– Théorie des types
– Théorie du raffinement
– ...

⌟

**Slide 18**

## Exemples d'applications industrielles

**avec les formalismes/méthodes $Z$, $VDM$, $CSP$**

IBM, INMOS, ...

CICS : Système transactionnel interactif (1983, $Z$)

Conception de circuits, Transputer ($Z$, $CSP$)

et de nombreux autres systèmes

⌟

## Exemples d'applications industrielles

**avec la méthode B (J-R. Abrial)**

GEC ALSTHOM, SNCF et MATRA Transport

**Slide 19**

- Système de controle de vitesse de train (KVS pour SNCF)
- Ligne A du RER - SACEM (signalistion, contrôle de vitesse)
- Metro de Calcutta (CTDC)
- Metro de Montreal(CTDC), Marseilles, Bel horizonte
- Météor (ligne de Métro sans conducteur)

⌟

**Slide 20**

- Assurance vieillesse, Sécurité sociale
- CICS de IBM (restructuration majeure du logiciel de gestion de transaction, 800000 lignes)
- B et VDM dans le domaine des logiciels de finance, BULL UK

⌟

Système de contrôle de vitesse (Metro)

– acquisition de données (capteurs, détecteurs, etc),

– 'calcul'/prise de décision,

**Slide 21**

– envois de commandes aux dispositifs physiques (ralentissement, freinage)

– embarquement du logiciel

⌟

**D'autres approches formelles utilisées aujourdhui**

certaines (outillées) sont industrialisées

RAISE (Résultat d'un projet ESPRIT)

Approche algébrique + *processus communicants*

**Slide 22**

LOTOS, SDL (Standard européen)

Approche algébrique + *processus communicants*

PVS (USA)

MEC, AltaRica (Université de Bordeaux + industriels)

⌟

Logiques classiques

Logique du premier ordre, Logique de Hoare, etc

**Slide 23**

Logiques non classiques, logiques modales

Logiques d'ordre supérieur, Théorie des types

⌟

**Usage des méthodes formelles**

- Marteau pour tuer une mouche
  - Selon la nature des besoins
- Environnement professionnel
  - compétences disponibles ?

**Slide 24**

- Contexte industriel
  - Délais, coûts, productivité
- Certification
  - Obligations des donneurs d'ordre

⌟

**Quelle approche utiliser ?**

→ plusieurs paramètres :
Concepteur/développeur de gros systèmes
Concepteur/développeur de petits systèmes "maison"

**Slide 25**

Nature des systèmes à développer

Compétences disponibles,
...

**Cas de la méthode B**

**Le système développé correct par construction**

Initialement, Modèle d'exécution séquentielle - Systèmes séquentiels

**Slide 26**

Pas de non-déterminisme de comportements

Systèmes autonomes et réactifs
Systèmes centralisés et distribués

## Cas de LOTOS

### Analyse et vérification du système développé

Modèle d'exécution séquentielle et parallèle
Systèmes séquentiels et concurrents

**Slide 27**

Possibilité de non-déterminisme de comportements

Systèmes autonomes et réactifs

Systèmes centralisés, systèmes distribués

⌟

## L'avenir ?

Formation aux méthodes formelles

Elaboration de méthodes et d'outils performant autour des langages et systèmes actuels,

**Slide 28**

Meilleure couverture du développement selon la nature des systèmes

Intégration des méthodes

⌟

**L'avenir ? (suite)**

Développement de standards (interopérables) pour gagner la confiance des industriels.

**Slide 29**

Utilisation à plus grande échelle des approches du type $B$ par la mise sur le marché des outils de travail de plus en plus performants.

Renforcement des approches hybrides, spécification/programmation + vérification à postériori

⌟

**L'avenir ? (suite)**

Développement et utilisation à grande échelle de l'approche synthèse de programmes pour des systèmes de taille importante.

**Slide 30**

⌟