Méthodologie de construction du logiciel (M3301-2)

Modélisation et construction des logiciels complexes

J. Christian Attiogbé

Université de Nantes

Plan du cours

Construction du logiciel

Le terme construction de logiciel (*software construction*) fait référence à la création détaillée d'un logiciel opérationnel à travers la combinaison du codage, de la vérification, des tests unitiares, des tests d'intégration et du déboggage.
La construction de logiciels est étroitement liée aux phases de conception et de test.

Positionnement dans les cycles

Le résultat de la phase de conception (*design*) est l'entrée de la phase de construction (*implementation*).
Le résultat de la phase de constrution est l'entrée de la phase de test (*tesing*).

Nécessité de méthodes : conception et construction

Variétés de systèmes et de méthodes

La nature des systèmes complexes oblige le développeur à utiliser
les formalismes,
méthodes,
techniques et
outils appropriés
au bon moment.

Nous allons nous focaliser sur les automates comme base de nombreux formalismes et méthodes

Démarches de conception ou de construction

- Approche TOP-DOWN (descendante)
  On procède par **décomposition**
  - Analyse globale (étude système, ingénierie de système)
  - **Architecture de logiciel** ; (modulaire, fonctionnelle structurée, ...)
    ↓
  - Codage des **composants (modules, fonctions, classes)**
    - Programmation directe ou
    - Développement formel (synthèse, raffinement)

- Approche BOTTOM-UP (ascendante)
  On procède par **composition** de composants élémentaires.
  - Etude des **composants (modules, fonctions, classes)** disponibles
  - **Composition (assemblage)**, réutilisation

Mais il faudra **modéliser et construire** les composants !
Par exemple avec la **Méthode B**.

Software Engineering Models and Methods

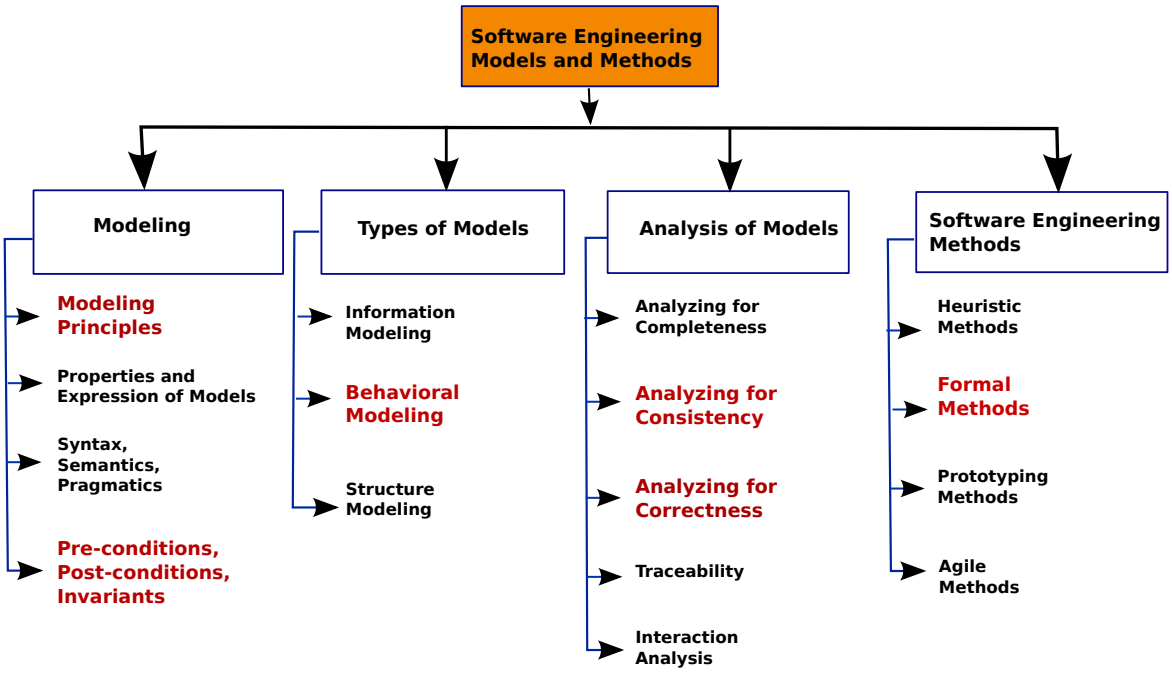

Figure: Software Engineering Models and Methods (source *ieee swebok*)

Avec quelles méthodes développer

Logiciels à développer

Figure: Cahier des charges

Avec quoi le faire ?

Figure: Quelles méthodes ?

Il faut connaître pour cela, la nature du logiciel à développer !

Et des normes (*standard*) de développement

Se renseigner sur les normes pour la construction de logiciels

Normes	Domaines	Caractéristiques?
IEC 62304	Systèmes embarqués, médical	
IEC 62279	*Railway applications*	
IEC 61506	*Industrial-process measurement and control*	
...	...	

Catégories ou types des systèmes

Les applications (ou logiciels) sont de différents types.
Des méthodes, outils et technologies sont associés aux types.

Applis	Types
Bases de données, Web, synthèse d'images, calcul numérique, contrôle de procédé industriel, système bancaire, enchères, réservation de ressources, gestion, télécommunication, etc	Systèmes interactifs, réactifs, répartis, transformationnels, embarqués, temps-réels, concurrents, ...

Pour chaque catégorie : des concepts, méthodes, techniques, outils

Catégories ou types des systèmes

Exercice : relier les applis et les types (puis les types et les technos) !

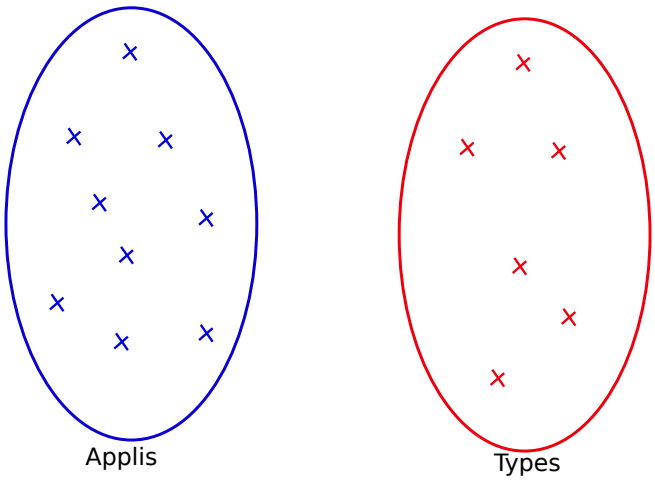

Figure: Relation (à trouver) entre les applis et leurs types

Diverses technologies pour la construction de logiciels

Applis ou paradigmes	Technologies
Programmation Objets, composants	C++, J2EE, Scala, Python, etc
Programmation par assemblage de composants, Architecture	Corba, EJavaBeans, .Net, ICE JavaEE, etc
Programmation multi-paradigme	OCaml, Haskell, JavaScript, Scala, etc
Patrons de conception	Spring (frameworks), MVC, CakePHP, etc
Applications Web	PHP, Javascript, Ajax (client) LAMP, WAMP

Linux Apache MySql PhP - Windows Apache MySql PhP

Diverses technologies pour la construction de logiciels

	Technologies
Architectures orientées services	Axis, Java Web Services Development Pack (JWSDP), JBossWS, Rest, BPMN, etc
Outils supports de développement	les IDE ; ANT, Maven, Sonar (test, qualité), Jenkins, etc
Serveurs d'applications	Java EE, .Net, Tomcat, etc
Ingénierie des modèles	MDE : UML/XML, MOF, QVT, ATL, Acceleo ; MDA, etc
Méthodes et qualité logicielle	agiles (Scrum, ...), CMMI, ITIL, etc
Normes, gouvernance	CMMi, ITIL, etc

Devoir de maison

- Remplissez le tableau suivant,
  en vous servant des définitions données dans la suite.

Catégories des systèmes informatiques/logiciels

Nature	Caractéristiques?	Méthodes/Langages /Techniques ?
Système séquentiel		
Système autonome *autonomous (transformational)*		
Système centralisé *centralised*		
Système réactif *reactive*		
Système temps-réel *real-time system*		
...		

Catégories des systèmes informatiques/logiciels

Nature	Caractéristiques?	Méthodes/Langages /Techniques ?
système parallèle *parallel system*		
Système parallèle et concurrent *parallel and concurrent*		
Système réparti *distributed*		
Système embarqué *embedded*		
Protocoles communication *communication protocols* . . .		

⇒ plusieurs types de systèmes informatiques, nécessité de méthodes variées

Systèmes autonomes transformationnels

Ils transforment leurs entrées en (résultats en) sortie.
Exemples : tri, édition, calculs, planification, etc

- Concepts : séquence, transition d'états, processeur unique
- Modélisation : Algorithmique, automates, hiérarchies fonctionnelles, UML, modèles relationnels, etc
- Etude : Logique, Automate, ...
- Programmation : langages de programmation classiques (impératifs, fonctionnels, objets, ...)

Systèmes concurrents

Un système dans lequel plusieurs tâches (identiques ou différentes) s'exécutent en même temps.
Exemples : producteurs-consommateurs de ressources

- Concepts : processus, concurrence, synchronisation, multi-processeurs
- Modélisation : automates, réseaux de Petri, algèbres de processus
- Etude : automates, logique, preuve, ...
- Programmation : processus, threads, primitives de synchronisation

Communication entre les tâches, accès concurrents aux données, cohérences des données, ...

Systèmes parallèles

Un système dans lequel plusieurs processeurs contribuent à exécuter la même tâche.
Exemples : imagerie, calculs numériques, parallélisation de divers programmes (tri dichotomique, etc, recherche web, multiplication de matrices ...),

- Concepts : multi-processeurs, synchronisation,
- Modélisation : automates, algèbres de processus,
- Etude : logique, automate, ...
- Programmation : primitives, multitache, processus, threads, multi-threads,

Systèmes répartis (*distributed systems*)

Type de systèmes concurrents où plusieurs ordinateurs coopèrent sans un programme central.
Exemples : réservation de places dans le transport, applications/services web, etc

- Concepts : asynchrone, non-déterminisme, causalité, consensus, messages, événements, mémoire partagée, (cf LAMPORT)
- Modélisation : Algorithmique distribué, Automates communicants, Réseaux de Petri, Algèbres de processus, logiques temporelles
- Etude : automates, réseaux de Petri, algèbres de processus,
- Programmation : Threads, API spécifiques dans les langages

Systèmes temps-réels

Exemples : contrôle de procédés industriels, robotique, aéronautique,

- Concepts : délais, *timeout*, synchronisation, préemption, ordonnancement
- Modélisation : Automates temporisés, moniteurs
- Etude : automates temporisés, réseaux de Petri, ...
- Programmation : *Threads*, C dédié, ...

Systèmes réactifs

Ce sont des systèmes qui réagissent ou répondent continuellement à des événements externes (de leur environnement).
Exemples : système de contrôle (de lumière, présence, etc) ; systèmes de détection d'objets, d'alarme...

- Concepts : événement, signal, réaction
- Modélisation : Automates de Mealy, Statecharts, approche ingénierie système
- Etude : basé sur les automates
- Programmation : langages synchrones (signal, esterel, scade) ; API spécifiques java

Les **systèmes interactifs** sont des cas particuliers de systèmes réactifs.

Systèmes embarqués (*embedded systems*)

Ce sont les systèmes informatiques avec une forte intégration et interaction entre une partie matérielle et une partie logicielle pour effectuer une tâche spécifique.
Caractéristiques : faible quantité de ressources de calcul et de stockage, faible consommation d'énergie, taille réduite pour le système (ils sont dédiés à des tâches spécifiques).
Exemples : assistance dans les véhicules, robotique, domotique, appareils médicaux, objets divers (montres, lecteur MP3, etc)

- Concepts : matériel-logiciel, interruption, préemption, contraintes de temps, performance,
- Modélisation : automates [temporsés], Matlab/simulink, Scilab, MARTE, UPPAAL, AADL, Papyrus
- Etude : Logique, automates, etc
- Programmation : Ada, C spécifique, (Environnements VisSim, etc)

Références

- Heath, Steve (2003). Embedded systems design
- ACM SE 2014 https://www.acm.org/education/se2014.pdf
- Modeling Software Software Behabiour, Jorgensen, CRC Press
- Software Engineering Design, Otero, CRP Press
- Embedded Systems Development, Sangiovanni &al, Springer
- *12 Best Software Development Methodologies with Pros & Cons* http://acodez.in/12-best-software-development-methodologies-pros-cons/

Références

- Software Engineering, Ian Sommerville
- Software Engineering Body of Knowledge (Swebok V3)
- ACM SE 2004 http://sites.computer.org/ccse/
- ACM SE 2014 https://www.acm.org/education/se2014.pdf
- International Council on Soft. and Sys. Engineering (INCOSE), Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities, version 3.2.2, International Council on Systems Engineering, 2012