Analyse par Objets
avec UML
(Unified Modeling Language)

Pr. Jean-Marc Jézéquel
IRISA - Univ. Rennes I

Campus de Beaulieu
F-35042 Rennes Cedex
Tel : +33 299 847 192 Fax : +33 299 842 532
e-mail : jezequel@irisa.fr
http://www.irisa.fr/prive/jezequel

1

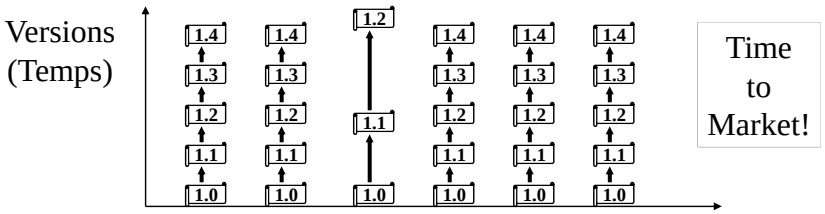
PLAN

- Démarche de modélisation avec UML
- Analyse initiale
  - à l'aide de techniques de brainstorming
  - Elaboration par « recuit simulé »
- Finalisation par critères qualités

2

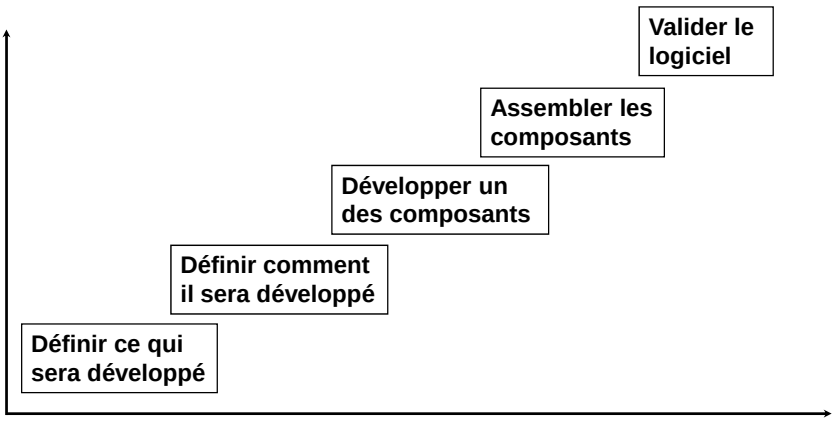
Problématique du logiciel « moderne »

- Importance des aspects non fonctionnels
  - systèmes répartis, parallèles et asynchrones
  - qualité de service : fiabilité, latence, performances...
- Flexibilité accrue des aspects fonctionnels
  - notion de *lignes de produits* (espace, temps)



3

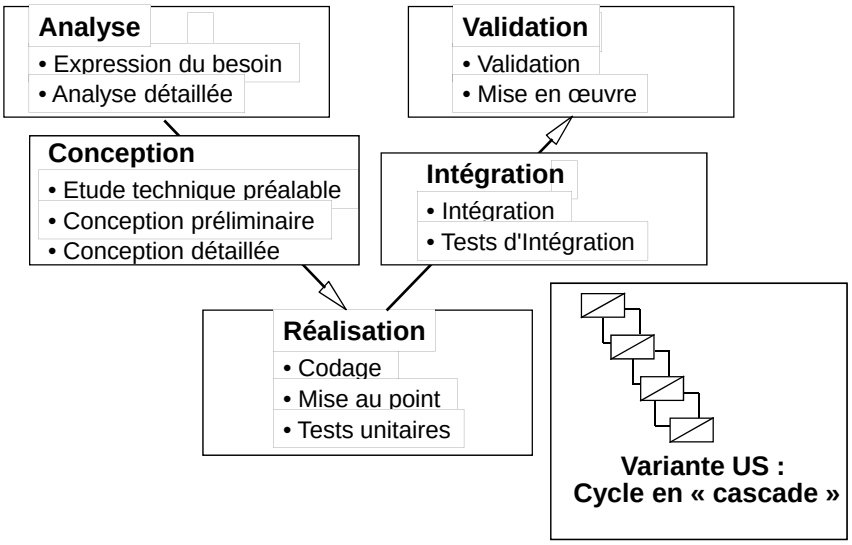
Activités du développement de logiciels



- L'organisation de ces activités et leur enchaînement définit le *cycle de développement* du logiciel

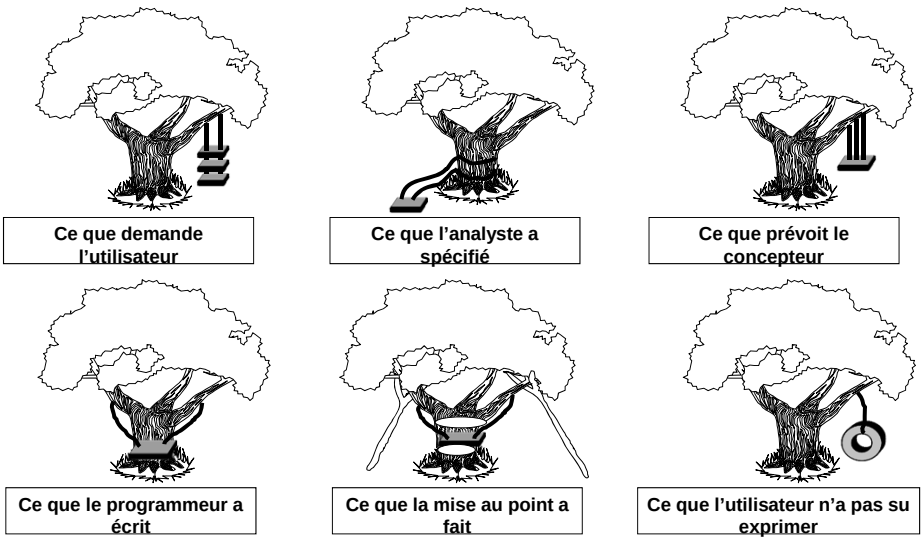
4

Cycle de vie en V normalisé AFNOR



5

Problèmes avec le processus classique...



Why do projects fail so often

- Unrealistic or unarticulated project goals
- Inaccurate estimates of needed resources
- Badly defined system requirements
- Poor reporting of the project's status
- Unmanaged risks
- Poor communication among customers, developers, and users
- Use of immature technology
- Inability to handle the project's complexity
- Sloppy development practices
- Poor project management
- Stakeholder politics
- Commercial pressures

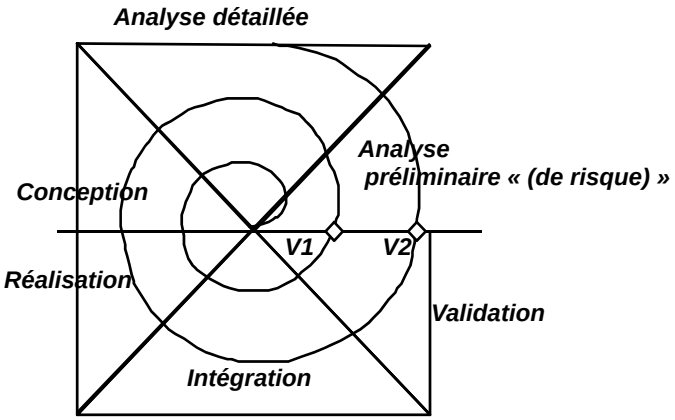
7

Problèmes du processus classique

- Organisation « industrielle » héritée du XIX^{ème} siècle
  - rassurant pour les managers
  - hiérarchie malsaine dans les rôles
  - antinomie : Coplien ’s organizational pattern
    - » *Architects Also Implement*
- cycle management <> cycle développement
- linéarité implicite
  - temps d ’approbation des documents => effet tampon
  - coût de la (non-) modification d ’un document « final »
  - irréaliste pour un projet innovant, donc à risques

8

Cycle de vie en « spirale »



Synergie avec approche par objets

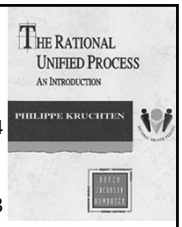
9

Intérêts du cycle de vie en « spirale »

- Bien adapté au développements innovants
  - les progrès sont tangibles : c 'est du logiciel qui « tourne » et pas seulement des kilos de documents
  - possibilité de s 'arrêter « à temps », i.e. avant que l 'irréalisabilité du projet ait créée un gouffre financier
- Moins simple à manager
  - difficile à gérer en situation contractuelle
  - mal contrôlé => on retombe dans le *hacking*
- Production des incréments asservie sur 2 parmi 3 :
  - période (e.g. release toutes les 2 semaines)
  - fonctionnalités (releases découpés suivant use-cases)
  - niveau de qualité (problème de la mesure)

10

Exemple du RUP



RPW ↔ RUP 2000 — 2000 — UML 1.4

RUP 5.5 — 1999 — UML 1.3, Project management
- Realtime ROOM

Rational Unified Process 5.0 — 1998 — Objectory UI design, Data Engineering, UML 1.1
- Performance testing
- Business Engineering
- Configuration & change Mgmt

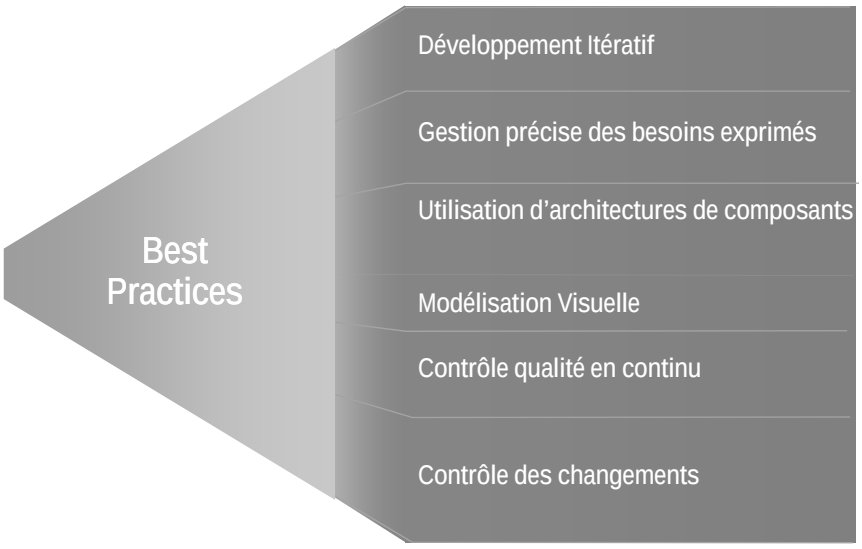
Rational Objectory Process 4.1 — 1997 — SQA Process
- Requirements College

Rational Objectory Process 4.0 — 1996
- OMT Booch
- UML 1.1

1995 — Rational Approach — Objectory 3.8

11

Principes du Rational Unified Process



Point clés

- Développer seulement ce qui est nécessaire
- Minimiser la paperasserie
- flexibilité
  - besoins, plan, utilisation des ressources, etc…
- Apprendre de ses erreurs précédentes
- Réévaluer les risques régulièrement
- Établir des critères de progrès
  - objectifs et mesurables
- Automatiser

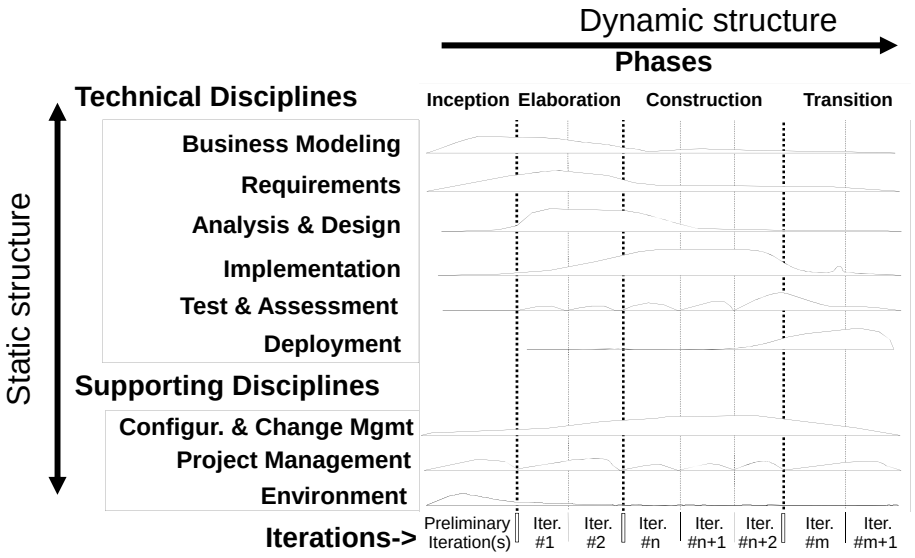
13

Architecture du processus

- 2 structures orthogonales
- Structure statique
  - Workers, artifacts, activities, workflows
  - authoring et configuration du processus
    - ☞ SPEMsi, ingénierie des méthodes et des processus
- Structure dynamique
  - Structure du cycle de vie : phases, itérations
  - Mise en oeuvre du processus : planification, exécution
    - ☞ gestion des activités, suivi de projet

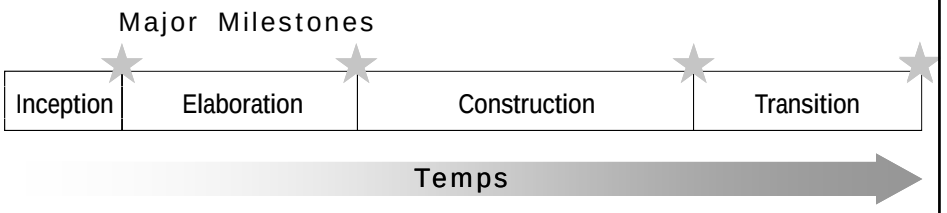
14

Les 2 dimentions du processus



15

Phases du développement itératif



Inception: définition de la porté du projet

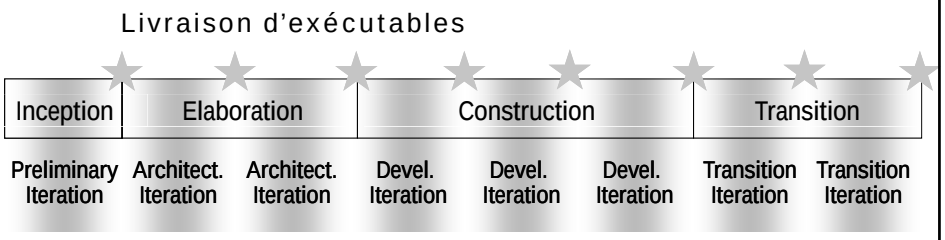
Elaboration: planification du projet, spécification des fonctionnalités, architecture de base

Construction: réalisation du produit

Transition: transfert du produit vers les utilisateurs

16

Itérations



Une itération est une séquence d'activités avec un plan bien établi et un critère d'évaluation, résultant en la livraison d'un logiciel *exécutable*.

17

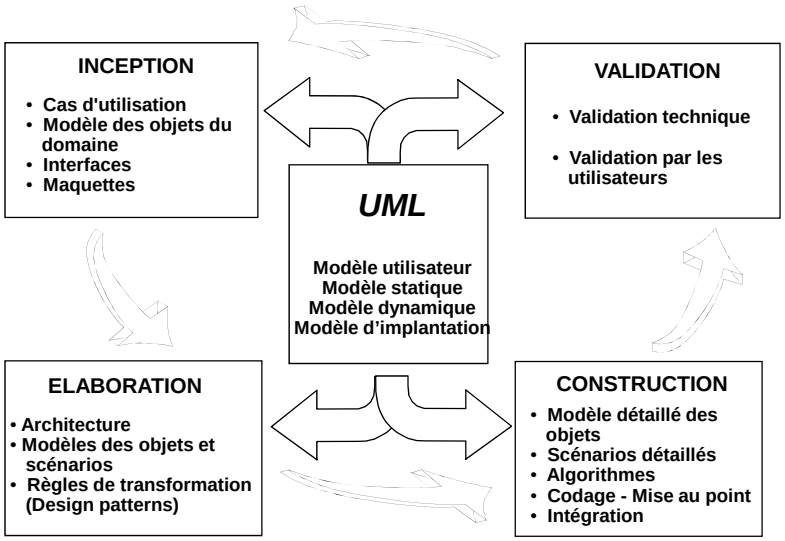
Phases et itérations : 2 exemples

- Petit projet de commerce électronique
  - Intégration à un mainframe
  - 5 personnes
- Grand projet d'infrastructure
  - Gros travail d'architecture nécessaire
  - 20 personnes

	No. of Iterations				Project Length	Iteration Length
	Inception	Elaboration	Construction	Transition		
e-business	0.2	1	3	1	3-4 months	2-3 weeks
infrastructure	1	3	3	2	9-12 months	5-7 weeks

18

Vision «générique» d'un cycle UML



19

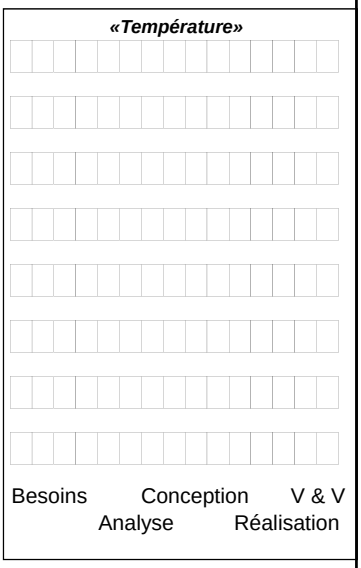
Processus de développement avec UML

- Approche itérative, incrémentale, dirigée par les cas d'utilisation
  - Expression des besoins
  - Analyse
    - » Elaboration d 'un modèle « idéal »
  - Conception
    - » passage du modèle idéal au monde réel
  - Réalisation et Validation

20

Température des diagrammes UML

- Diagramme de cas d'utilisations
- Diagramme de classes
- Diagramme de paquetages
- Diagramme de séquences
- Diagramme de collaborations
- Diagramme d'états-transitions
- Diagramme d'activités
- Diagramme d'implantation



21

Objectif d'une analyse avec UML

- Construction d 'un modèle d'analyse
- Modèle unique mais quadri-dimentionnel
  - Aspects utilisation du système
  - Aspects statiques du système
  - Aspects dynamiques du système (comportemental)
  - Aspects déploiement et implantation

22

Construction d'un modèle d'analyse

- Hypothèse d'un monde idéal
  - processeur suffisamment rapide
  - mémoire suffisamment grande et rapide
  - communications rapides et fiables
  - pas de défaillances
- Sélection des aspects cruciaux d'un problème
- Pas de solution unique :
  - modélisation = interprétation
  - « *Quoiqu'on en dise, dans la vie scientifique, les problèmes ne se posent pas d'eux-mêmes. Rien ne va de soi. Rien n'est donné. Tout est construit.* » G. Bachelard

23

Modélisation UML

- Modélisation selon 4 points de vue principaux :
  - Vision utilisateur du système
    - » Cas d'utilisation
  - Aspects statiques du système
    - » Description des données et de leurs relations
    - » Structuration en paquetages
  - Aspects dynamiques du système (comportemental)
    - » Diagramme de séquences (scénarios)
    - » Diagramme de collaborations (entre objets)
    - » Diagramme d'états-transitions (Harel)
    - » Diagramme d'activités
  - Vision implantation
    - » Diagramme de composants et de déploiement

24

Etude de cas

- Un serveur de réunions virtuelles
  - Adaptation du concept d'IRC à un contexte de réunions de travail au sein d'une entreprise géographiquement dispersée
- Cette étude de cas va nous servir de fil conducteur pour décrire un processus d'analyse et de conception avec UML

25

Cahier des charges (1/2)

- Il s'agit de réaliser la partie serveur d'une application client-serveur permettant de faire des réunions virtuelles multimédia sur Internet. L'objectif de cette application est de permettre d'imiter le plus possible le déroulement de réunions de travail classiques. Cependant, dans la première version de ce projet, les interventions des participants se feront en mode mono-média seulement (i.e. échanges en forme textuelle).
- Le serveur devra permettre de planifier et de gérer le déroulement de plusieurs réunions simultanées. Des programmes clients existeront dans l'avenir pour plusieurs plate-formes (Mac, Windows, Unix) afin de permettre à des personnes désirant organiser des réunions virtuelles ou y participer de dialoguer avec le serveur en utilisant un protocole ad hoc développé au dessus de IP.

26

Cahier des charges (2/2)

- Après s'être connecté au serveur (à l'aide d'un nom de login et d'un mot de passe mémorisé par le système), une personne a la possibilité de planifier des réunions virtuelles (choix d'un nom, définition du sujet, date de début et durée prévue, ordre du jour), de consulter les détails d'organisation d'une réunion, de les modifier (seulement l'organisateur), d'ouvrir et de clôturer une réunion (seulement l'animateur), d'entrer (virtuellement) dans une réunion précédemment ouverte, et d'en sortir. En cours de réunion, un participant peut demander à prendre la parole. Quand elle lui est accordée, il peut entrer le texte d'une intervention qui sera transmise en ``temps-réel'' par le serveur à tous les participants de la réunion.
- Plusieurs sortes de réunions doivent pouvoir être organisables :
  - Réunions standards, avec un organisateur qui se charge de la planification de la réunion et désigne un animateur chargé de choisir les intervenants successifs parmi ceux qui demandent la parole.
  - Réunions privés, qui sont des réunions standards dont l'accès est réservé à un groupe de personnes défini par l'organisateur
  - Réunions démocratiques, qui sont planifiées comme des réunions standards, mais où les intervenants successifs sont choisis automatiquement par le serveur sur la base d'une politique premier demandeur-premier servi.

27

Démarche de modélisation avec UML

- Construction du diagramme de cas d'utilisations
  - Grande découpe fonctionnelle (10% de l'effort)
- Construction du diagramme de classes
  - à partir des noms des données du problème (30%)
- Construction de diagrammes de séquences et de collaborations
  - instances des cas d 'utilisation (25%)
- Généralisation à l'aide de diagrammes d'états-transitions
  - à partir des diag. Séquences (15%)
- Affiner et préciser la solution (20%)

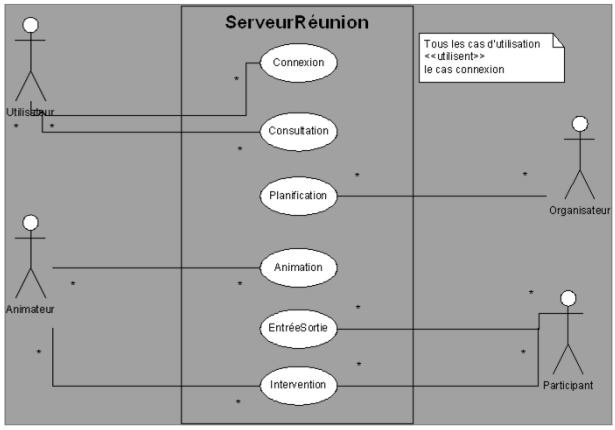
28

Cas d’utilisation

- Grande découpe fonctionnelle
  - pilote les incréments dans la spirale
  - chaque incrément correspond à la réalisation d ’un cas d ’utilisation
- 1 diagramme global + texte 10 lignes / cas

29

Diagramme de cas d'utilisations



30

Exemple de cas d'utilisation : Planification

- La *planification* d'une réunion virtuelle est effectuée par une personne jouant le rôle d'organisateur pour cette réunion. Ceci consiste à créer une nouvelle réunion dans le système (ou à la mettre à jour si elle existe déjà) en faisant le choix d'un nom, la définition du sujet, de la date de début et la durée prévue, ainsi que l'ordre du jour.

31

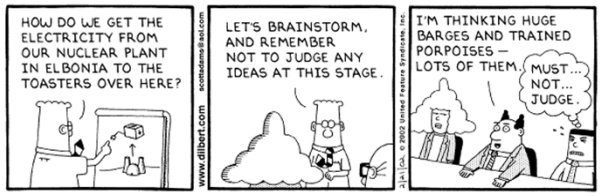
Processus de construction du diagramme de classes

- Identifier les classes d'objets
  - Garder les bonnes classes
  - constitution du dictionnaire de données
- Identifier les associations.
  - Garder les bonnes associations
- Identifier les attributs.
  - Garder les bons attributs
- Raffiner au moyen de l'héritage
  - Généralisations et raffinages
- Itérer la modélisation
- Grouper les classes en modules

32

Identification des classes

- A l'aide des noms des données du problème
  - processus de remue-méninges (brainstorming)



- Eliminer les classes :
  - redondantes (noms synonymes)
  - non pertinentes (vs. le modèle)
  - trop vagues (non réifiable facilement)
  - attributs, opérations, rôles de relations
  - constructions liées à l'implantation

33

Souligner les noms dans le cahier des charges

- Il s'agit de réaliser la partie serveur d'une application client-serveur permettant de faire des réunions virtuelles multimédia sur Internet. L'objectif de cette application est de permettre d'imiter le plus possible le déroulement de réunions de travail classiques. Cependant, dans la première version de ce projet, les interventions des participants se feront en mode mono-média seulement (i.e. échanges en forme textuelle).
- Le serveur devra permettre de planifier et de gérer le déroulement de plusieurs réunions simultanées. Des programmes clients existeront dans l'avenir pour plusieurs plate-formes (Mac, Windows, Unix) afin de permettre à des personnes désirant organiser des réunions virtuelles ou y participer de dialoguer avec le serveur en utilisant un protocole ad hoc développé au dessus de IP.

34

Classes potentielles

– Serveur	Implantation
– Application	Redondant serveur
– Réunion	OK
– Internet	Implantation
– Objectif	Non pertinent
– Déroulement	Action
– Version	Implantation
– Projet	Non pertinent
– Intervention	OK (?)
– Participant	Rôle relation Personne-Réunion
– Mode	Implantation
– Programme	Implantation
– Plate-formes	Implantation
– Personne	OK
– Protocole	Implantation
– …	

35

Identification des relations entre classes

- Recherche des phrases verbales
- Eliminer les relations :
  - entre classes éliminées
  - non pertinentes ou liées à l'implantation
  - qui sont en fait des actions
  - pouvant être dérivées d'autres relations
  - Raffiner la sémantique des relations
  - ajouter les rôles
  - qualifier les relations (sélecteur)
  - spécifier la multiplicité

36

Identification des attributs

- Propriétés d'objets individuels
  - à rechercher à l'aide des adjectifs (couleur, poids...) ou propositions substantives du problème
- Eliminer les attributs non nécessaires ou incorrects
  - s'ils sont en fait des objets
  - sélecteurs de relations
  - identificateurs (clef de BD)
  - attributs de relations
  - valeurs internes ou détails d'implantation

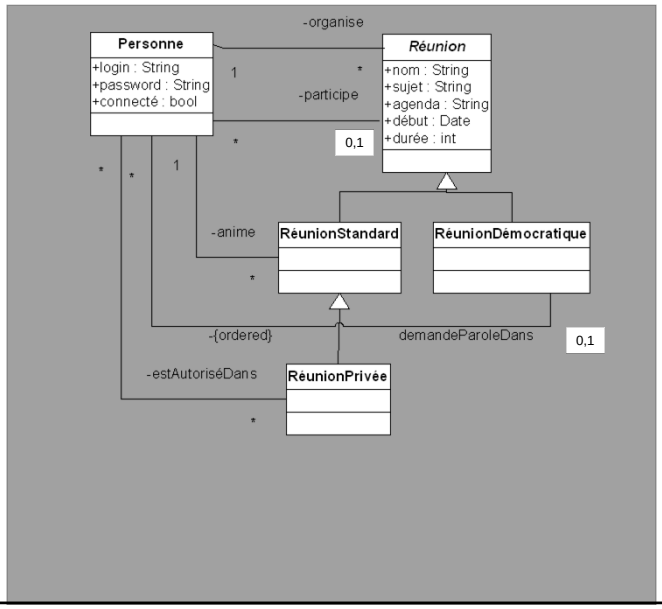
37

Raffiner au moyen de l'héritage

- Généralisation à l'aide de super-classes
  - Recherche de classes avec des attributs, relations ou opérations similaires
- Spécialisation à l'aide de sous-classes
  - Différentes variantes d'une même classe
    - » Réunion privée, démocratique …
  - sous-classe ou attribut pour distinguer ?

38

Diagramme de classes d'analyse



39

Itérer la modélisation

- Classes manquantes ou en trop
  - asymétries : ajout de classes par analogies
  - scinder les classes disparates en classes plus élémentaires
  - A quoi sert une classe si pas d'attributs ou d'opérations?
- Relations manquantes, en trop ou mal placées
  - si aucune opération ne traverse une relation...
- Attributs
  - accéder à un objet par un attribut -> relation qualifiée
- Il faut parfois attendre d'avoir fait les autres vues pour pouvoir itérer

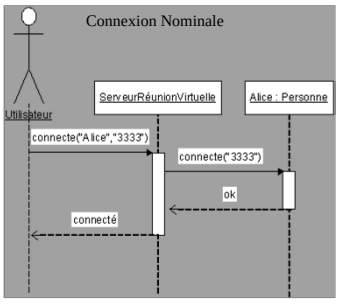
40

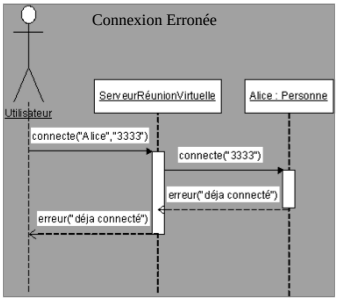
Diagrammes dynamiques

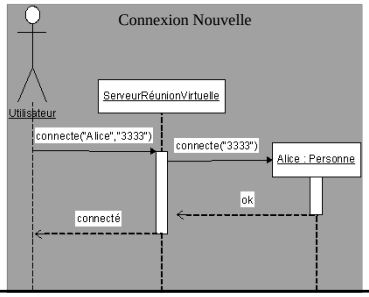
- Pour chaque cas d ’utilisation
  - – un scénario nominal (diagramme de séquence)
    - » tout se passe bien
  - – quelques scénarios exceptionnels
    - » montrent des variations sur le scénario optimal
- Alternativement (Catalysis)
  - – description par pre/post de l ’état du système avant/après chaque occurrence d ’événement

41

Exemples de scénarios : Cas d'utilisation Connexion

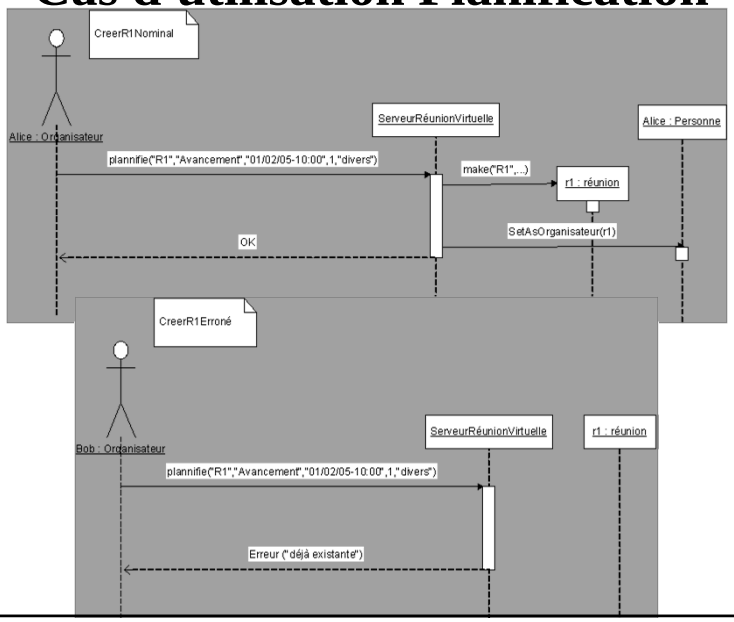






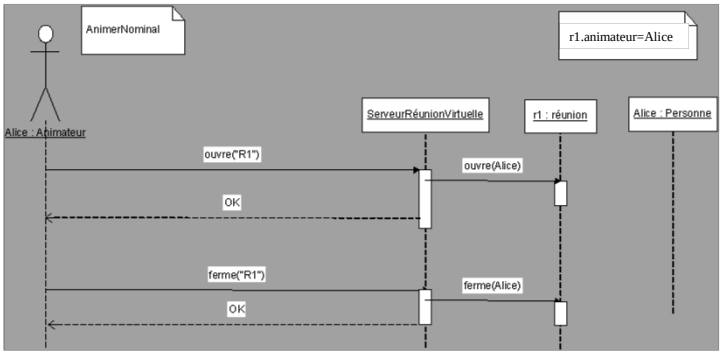
42

Exemples de scénarios : Cas d'utilisation Planification



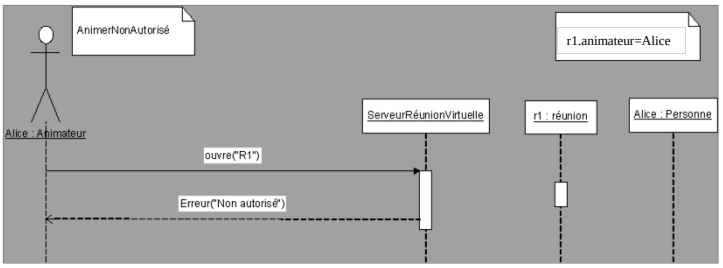
43

Exemples de scénarios : Cas d'utilisation Animation 1/3



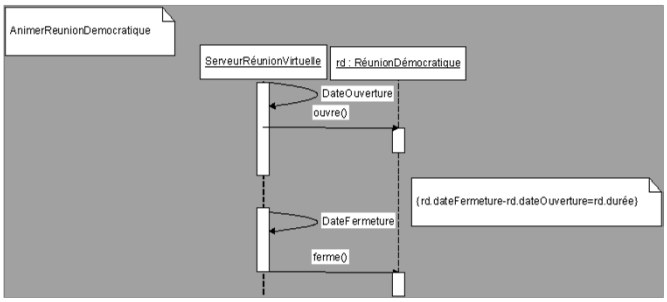
44

Exemples de scénarios : Cas d'utilisation Animation 2/3



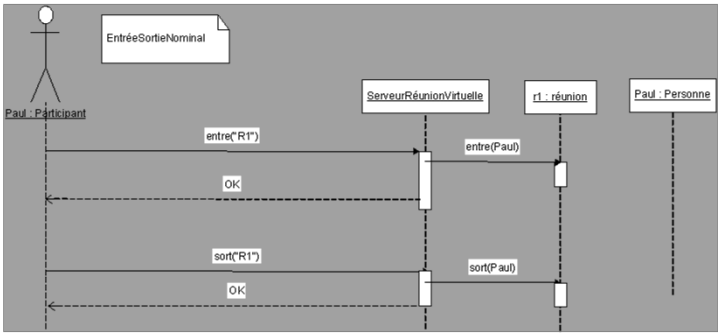
45

Exemples de scénarios : Cas d'utilisation Animation 3/3



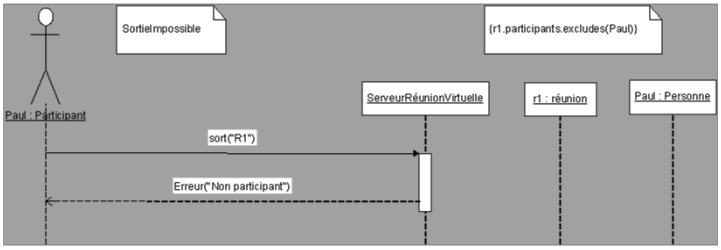
46

Exemples de scénarios :
Cas d'utilisation EntréeSortie 1/4



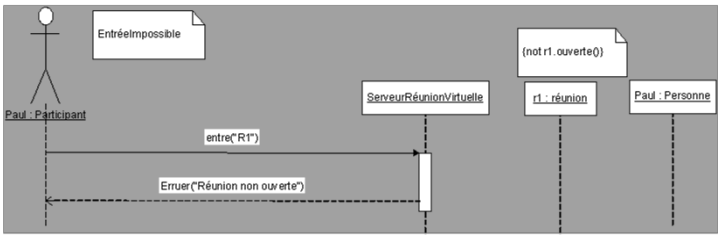
47

Exemples de scénarios :
Cas d'utilisation EntréeSortie 2/4



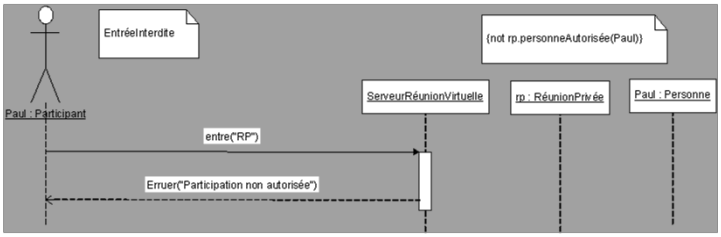
48

Exemples de scénarios : Cas d'utilisation EntréeSortie 3/4



49

Exemples de scénarios : Cas d'utilisation EntréeSortie 4/4



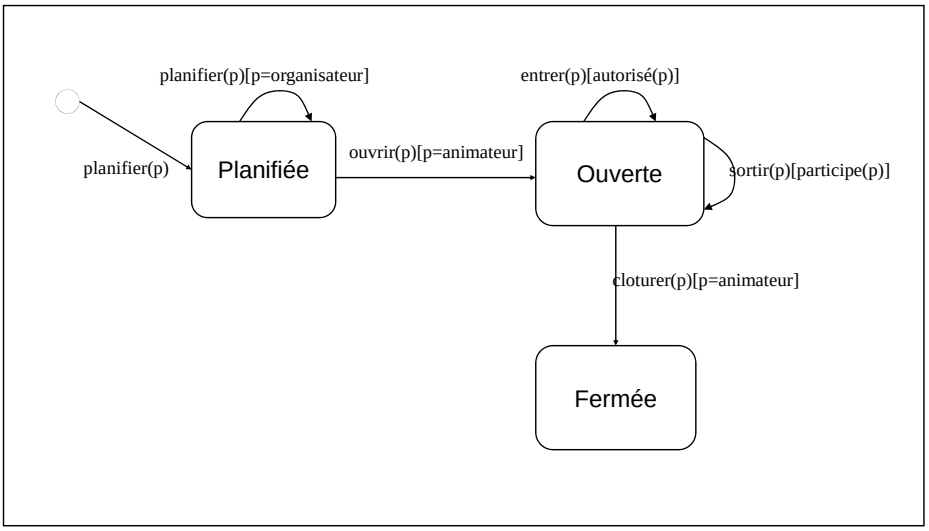
50

Construction des diagrammes d’états

- Généralisation pour une classe donnée de l ’ensemble des scénarios qui mettent en jeu ses instances
  - en suivant les transition de l ’automate, on doit pouvoir retrouver tous les scénarios

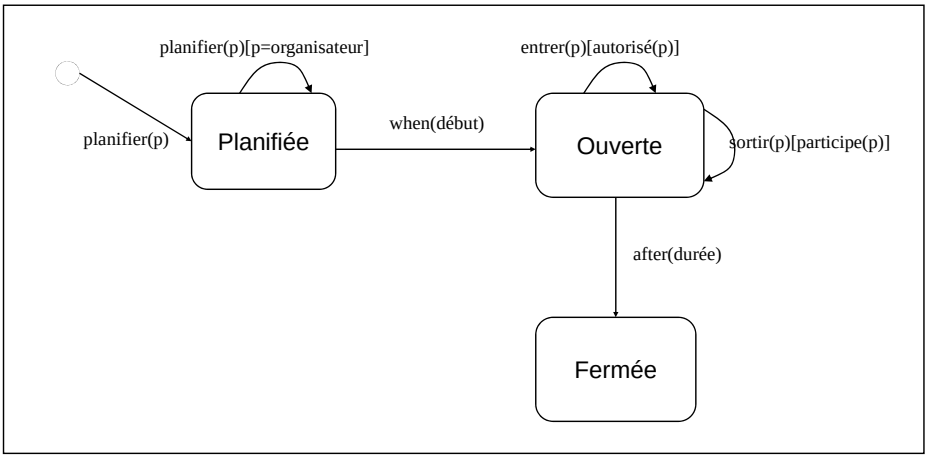
51

Diagramme d'états : Réunion



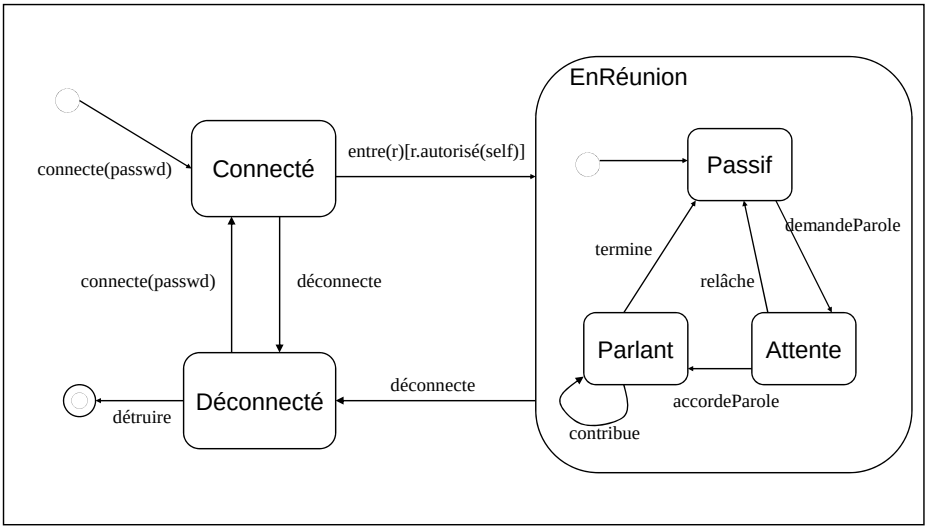
52

Diagramme d'états : Réunion démocratique



53

Diagramme d'états : Personne



54

Conseils pratiques

- Réfléchir au problème avant de commencer
  - Soigner le nommage, insister sur le nommage des relations et des rôles
- Faire simple!
  - «*Things must be as simple as possible, but no simpler*». A. Einstein
  - éviter toute complication nuisible
    - » utiliser les qualifieurs
    - » éviter les relations ternaires, quaternaires (trop complexe)
    - » se dégager de l'implémentation : raisonner objets, classes, messages, relations, attributs, opérations
  - ne pas s'inquiéter si les possibilités de la notation ne sont pas toutes exploitées

55

Conseils pratiques (suite)

- Approche incrémentale
  - Itérer
  - Confronter ses modèles aux autres
  - Savoir s'arrêter avant d'atteindre la perfection...
    - » prise en compte qualité (niveau de précision), coûts, délais...
    - » asservissement au processus de développement
- Faire simple (encore)
  - *Il semble que la perfection soit atteinte, non quand il n'y a plus rien à ajouter mais quand il n'y a plus rien à retrancher* - Antoine de Saint-Exupéry, *Terre des hommes*

56

Critères de qualité d'un bon modèle d'analyse avec UML (1/2)

- Cas d 'utilisation
  - environ 6 cas, nommage correct des acteurs (noms) et des cas (actions)
  - pour chaque cas, présence d 'un court texte d 'explication
  - complétude vs. Cahier des charges
- Diagramme de classes
  - nommage correct des classes (noms)
  - définitions présentes dans un dictionnaire
  - nommage de toutes les relations, présence des cardinalités
  - attributs seulement de types simples
  - non duplication attributs/relations (utilisation héritage)

57

Critères de qualité d'un bon modèle d'analyse avec UML (2/2)

- Diagrammes de séquence
  - attachés à un cas d 'utilisation
  - pour chaque cas d 'utilisation,
    - » présence d 'un scénario nominal
    - » quelques scénarios exceptionnels
  - chaque opération (message reçu par un objet) définie dans la classe correspondante du diagramme statique
- Diagrammes d 'états
  - chaque automate est attaché à une classe
  - chaque attribut/opération utilisée sur l 'automate doit être défini dans la classe englobante
  - en suivant les transition de l 'automate, on doit pouvoir retrouver tous les scénarios

58