Un aperçu de l'ingénierie du logiciel en 12 leçons de L1 Informatique

Mickaël Kerboeuf, Vincent Ribaud

Un aperçu de l'ingénierie du logiciel en 12 leçons de L1 Informatique

Mickaël Kerboeuf, Vincent Ribaud
EA3883, LISyC, Université de Bretagne Occidentale, C.S. 93837 29238 Brest Cedex 3, France
E-mail: {Mickael.Kerboeuf, Vincent.Ribaud} @univ-brest.fr

Résumé

Un cours d'ingénierie du logiciel a été créé cette année en L1 Informatique (deuxième semestre). L'objectif principal est de donner une vision globale du cycle de développement d'un logiciel. Au premier semestre, l'enseignement de l'informatique est commun à la plupart des mentions de la licence Sciences, Technologies, Santé. Au cours de la première année, les élèves ont 48h d'introduction à l'architecture des machines et à l'algorithmique en CAML, et 48h d'initiation à la programmation en Visual Basic et aux outils bureautiques (Word et Excel).
La pédagogie retenue est centrée sur le projet en déroulant le cycle de développement lors de 12 leçons de 2 heures (sur 12 semaines). La répartition des leçons est la suivante : cahier des charges et cycle de développement (1), expression de besoins (2), analyse (3), conception (3), utilisation d'un logiciel correspondant au cahier des charges et non-réalisé par les élèves (1), tests et qualification du logiciel (2). L'accent est volontairement mis sur toutes les activités d'ingénierie du logiciel à l'exception de la programmation.
Sauf pour la conception, le cycle de vie en V permet un apprentissage séquentiel et « par l'exemple » : du cahier des charges à l'expression des exigences avec les cas d'utilisation, des cas d'utilisation à l'analyse avec les diagrammes de classe et de collaboration, de l'analyse aux tests avec un plan de tests et des cas de tests issus des exigences. L'apprentissage de la conception est abordé à l'aide d'une démarche inductive de type rétro-ingénierie. Un modèle de programmation en couches (données, traitements, présentation) est défini et illustré avec Excel, Query et Visual Basic. Ensuite, certains composants déjà programmés sont rétro-conçus pour faire prendre conscience de ce que sont les artefacts de conception (reprenons la définition de l'ISO 15504 : identifier les composants principaux du logiciel et les raffiner en unités plus petites qui peuvent être codées, compilées et testées). A partir de là, il est possible d'effectuer l'activité de conception proprement dite c.à.d. obtenir à partir de l'analyse des modèles de conception : modèle logique de données, vues (ou requêtes), algorithmes de traitement, dialogue homme-machine.
Deux projets sont le support des apprentissages. Le premier, un agenda électronique partagé, est utilisé pour illustrer les produits de sortie (*« deliverables »*) de chaque phase du cycle. Les "deliverables" du premier projet, établis par les enseignants, servent de base et d'exemple aux élèves lors du développement du deuxième projet, un forum de discussion. Afin d'éviter la phase de codage (hors de portée du cours et des élèves), un forum de discussion correspondant à peu près au cahier des charges est fourni. Les élèves ont à leur charge de configurer ce qui est possible, de constater les écarts avec les exigences qu'ils ont formulées dans les cas d'utilisation et de qualifier le logiciel fourni lors des tests.
L'évaluation porte pour 2/3 sur le projet réalisé (1/3 en commun et 1/3 en examen individuel) et pour 1/3 sur les connaissances enseignées.
Ce cours semble bien reçu par les élèves (une douzaine) malgré des difficultés au début de chaque phase due à la complexité et à l'étendue des sujets abordés.

1 Introduction

Dans le cadre de la mise en place du LMD, une UE d'introduction à l'ingénierie du logiciel est dispensée aux élèves du deuxième semestre de la première année de licence informatique, parcours Informatique-Système (ex-DEUG STPI). Au premier semestre, l'enseignement de l'informatique est commun à toutes les mentions de la licence Sciences, Technologies, Santé, à l'exception des sciences de la vie et de l'univers. Dans ce premier semestre, les élèves ont un cours de 48 heures d'introduction à l'architecture des machines et à l'algorithmique en CAML. Au deuxième semestre, un cours de 48 heures dispense une initiation à la programmation en Visual Basic et aux outils bureautiques (Word et Excel).
Il existe une réelle difficulté à enseigner l'ingénierie du logiciel dans le paradigme d'enseignement classique (par matière et organisé en cours magistral, travaux dirigés et travaux pratiques).
Nous reprenons une idée fréquemment utilisée (cf. B. Meyer [Mey0] par exemple) d'employer un projet à long terme (sur le semestre) comme support essentiel de l'apprentissage. La pédagogie retenue rompt avec le paradigme d'enseignement habituel pour reposer sur un paradigme d'apprentissage [Tar98]. En grossissant le trait, on cherche à remplacer les activités liées au verbe « enseigner » par des activités liées au verbe « apprendre ».

Le processus d'apprentissage est guidé par le cycle de développement du logiciel. Le cycle de développement définit notamment le rôle et l'enchaînement des phases d'un projet.
Pour une introduction à l'ingénierie du logiciel, 6 phases ont été retenues :

1. Mise en place du projet à partir d'un cahier des charges et d'un macro-modèle de cycle de développement du logiciel
2. Expression et recueil des exigences à l'aide des cas d'utilisation
3. Analyse des exigences à l'aide du modèle entité-association et de diagrammes de collaboration
4. Conception à l'aide du modèle relationnel et d'algorithmique procédurale
5. Réalisation
6. Tests à l'aide de plans de test

En réalité, deux projets sont utilisés. Le premier projet (un agenda) constitue la base de référence des élèves, il est établi par les enseignants, utilisé régulièrement en classe comme exemple et ses produits de sortie ("déliverables") fournis aux élèves au début de chaque phase. Le deuxième projet (un forum de discussion) est établi par les élèves en groupe de 2 ou 3 au fil du semestre. Pour le projet « Forum », la phase de réalisation n'est pas effectuée ; un logiciel correspondant à la plupart des exigences formulées est fourni aux élèves, à charge pour ceux-ci d'évaluer les écarts avec le cahier des charges.
4 livraisons sont demandées aux élèves à l'issue des phases 2, 3, 4 et 6.

L'environnement pédagogique qui en résulte possède des caractéristiques qui, selon Jacques Tardif, sont en cohérence avec le paradigme d'apprentissage :

- la constance de l'apprentissage et la variation du temps individuel
- le déséquilibre cognitif
- l'authenticité des situations d'apprentissage
- la transdisciplinarité
- les interactions constantes entre la théorie et la pratique
- l'intégration des évaluations aux situations d'apprentissage

La suite de l'article présente, phase par phase, les principaux objectifs d'apprentissage visés. Le tableau ci-dessous donne le nombre de séances de 2h (leçons) passées sur chaque phase. Une leçon mélange cours, exercices et travaux pratiques.

Phases	Nombre de leçons
Mise en place du projet	1
Expression du besoin	2
Analyse des exigences	3
Conception	3
Réalisation (utilisation)	1
Tests	2

2 Mise en place du projet

La première séance définit les objectifs du cours :
- un premier apprentissage d'une méthode de conception et de fabrication de projets logiciels,
- une vue d'ensemble du métier correspondant (« ingénieur du logiciel ») et de ses activités,
- le travail en équipe et par projet.

2.1 Qu'utilise-t-on du cycle de vie

Le cycle de vie du logiciel retenu est en V et les activités d'ingénierie afférentes sont définies conjointement à partir du référentiel-métier de la société Thales Information Services et du Processus Unifié de Jacobson, Booch, Rumbaugh [JBR99]. Afin de donner du sens au vocabulaire utilisé, un modèle général de projet est défini du point de vue des phases (temporelles) le composant et ces phases sont mises en regard avec les *phases du cycle de développement du logiciel* :
- orientation (préciser les finalités et les politiques et choisir les valeurs de référence) : *schéma directeur ;*
- préparation (percevoir un problème, recueillir et traiter l'information nécessaire) : *cahier des charges ;*
- réflexion (constituer une équipe-projet et établir un avant-projet) : *réponse à un appel d'offre ou étude préalable ;*
- élaboration (rédiger le projet et le contrat de projet) : *plan de projet, analyse, conception ;*
- réalisation : *codage, tests, intégration ;*
- appropriation (bilan et suivi) : *qualification, mise en production, maintenance en conditions opérationnelles.*

Les différents intervenants d'un projet sont présentés avec leur rôle : client / maîtrise d'ouvrage, conduite d'affaires / maîtrise d'œuvre, chef de projet / contremaître, ingénieur logiciel / ouvrier spécialisé ...

2.2 Extrait du cahier des charges de l'agenda

L'agenda permettra de créer des réunions, d'inscrire sa participation à une réunion, d'annuler sa participation à une réunion et d'effectuer des consultations des réunions existantes.

...

Une réunion comporte une à plusieurs personnes et se tient dans un seul lieu. Un lieu peut accueillir successivement plusieurs réunions. Une personne peut participer à plusieurs réunions disjointes dans le temps.

...

On peut créer une réunion en saisissant ses caractéristiques, ainsi que le lieu où elle se tient. Le système doit vérifier que le lieu est disponible pendant la période nécessaire à la réunion. On peut inscrire la participation d'une personne à une réunion existante. Le système doit vérifier que la personne est disponible pendant la période nécessaire à la réunion. On peut annuler la participation d'une personne à une réunion existante. On peut rechercher les réunions à partir des critères suivants : personne participant, lieu, sujet et intervalle temporel. Le résultat d'une recherche est affiché à l'écran. On peut modifier certaines caractéristiques d'une réunion. On peut supprimer des réunions du système.

...

3 Expression de besoin

Le point central de l'expression de besoin est d'établir l'importance de la notion d'exigence comme contrat entre le client et le fournisseur. Une exigence est d'abord défini d'après le Larousse « Ce qui est commandé par quelque chose ou quelqu'un ; nécessité, obligation », puis d'après la norme IEEE 729-1983 « Condition ou capacité que doit présenter un système pour satisfaire un contrat, un standard, une spécification ou tout autre document formel imposé ».
Du temps est accordé pour que les élèves réfléchissent sur tous les mots du paragraphe précédent, notamment sur le sens « habituel » qu'on leur prête. On insiste beaucoup sur la nécessité de rédiger un contrat lisible pour le client, c'est-à-dire établi aussi pour un point de vue utilisateur. Enfin, on introduit le modèle des cas d'utilisation.

3.1 Qu'utilise-t-on des cas d'utilisation

D'un point de vue technique, un cas d'utilisation établit, entre différents intervenants, un contrat régissant le comportement d'un système.
Voici une synthèse des notions transmises aux élèves :
Un acteur est une entité qui a un comportement.
Un scénario est une suite d'actions produisant un résultat observable qui apporte une valeur à un acteur.

Un cas d'utilisation est une collection de scénarii de réussite ou d'échecs qui décrivent la façon dont un acteur utilise un système pour atteindre un objectif.
Le cas d'utilisation comme contrat de comportement repose sur deux modèles décrits par Cockburn [Coc01]. Le système est le support de l'interaction entre acteurs ayant des objectifs. Il a la responsabilité de protéger les intérêts de tous les intervenants (sur-ensemble des acteurs).

3.2 Méthode de travail

Après une présentation très rapide des concepts ci-dessus, les élèves participent à des ateliers d'écriture de cas d'utilisation, d'abord sur des problèmes de la vie quotidienne (aller boire un café), puis sur des règles de jeux (questions pour un champion). Les cas d'utilisation de l'agenda sont utilisés comme support d'exercices.

3.3 Un exemple de cas d'utilisation

IDENTIFICATION

Titre : Participation à une réunion

DESCRIPTION DES SCENARIOS

Préconditions :

I. Le système est opérationnel.
II. L'utilisateur est connecté.
III. Au moins une réunion est créée.

Scénario stratégique : « Inscription / annulation à une réunion »

1. L'utilisateur recherche une réunion.
2. L'utilisateur sélectionne une des réunions parmi le résultat de la recherche.
3. L'utilisateur peut inscrire la participation d'une personne, annuler la participation d'une personne ou consulter les personnes inscrites à cette réunion.
4. L'utilisateur quitte la gestion de la participation à une réunion.

Scénario principal A : Inscrire la participation d'une personne
Garantie en cas de succès :

I. Une nouvelle personne est inscrite à une réunion.

1. Le système affiche la liste des personnes existantes.
2. L'utilisateur saisit le nom et le prénom de la personne concernée ou choisit la personne dans la liste des personnes existantes.
3. Le système vérifie que la personne est disponible pendant la durée de la réunion.
4. L'utilisateur confirme l'inscription de la personne.

Extensions possibles :

2a. Saisie du nom ou du prénom invalide.

1. Un des champs « nom » ou « prénom » est incorrectement saisi.
2. Le système affiche un message d'erreur
3. Le système efface le(s) champ(s) incorrect(s).

Le scénario reprend en 2.

2b. Personne homonyme.

1. Il existe plusieurs personnes de même nom et de même prénom.
2. Le système précise pour les homonymes des informations spécifiques comme le numéro de téléphone.

Le scénario reprend en 2.

1-4a. Annulation

1. L'utilisateur annule l'inscription de la personne.

Fin du scénario principal A, retour à l'étape 3 du scénario stratégique.

Scénario principal B : Annuler la participation d'une personne
Garantie en cas de succès :

I. L'inscription d'une personne à une réunion est annulée.

1. Le système affiche la liste des personnes existantes.
2. L'utilisateur saisit le nom et le prénom de la personne concernée ou choisit la personne dans la liste des personnes existantes.
3. L'utilisateur confirme l'annulation de participation de la personne.

Extensions possibles :

2a. Saisie du nom ou du prénom invalide.

1. Un des champs « nom » ou « prénom » est incorrectement saisi.
2. Le système affiche un message d'erreur
3. Le système efface le(s) champ(s) incorrect(s).

Le scénario reprend en 2.

2b. Personne homonyme.

1. Il existe plusieurs personnes de même nom et de même prénom.
2. Le système précise pour les homonymes des informations spécifiques comme le numéro de téléphone.

Le scénario reprend en 2.

1-3a. Annulation

1. L'utilisateur ne confirme pas l'annulation de participation de la personne.

Fin du scénario principal B, retour à l'étape 3 du scénario stratégique.

Scénario principal C : Consulter les personnes inscrites à cette réunion

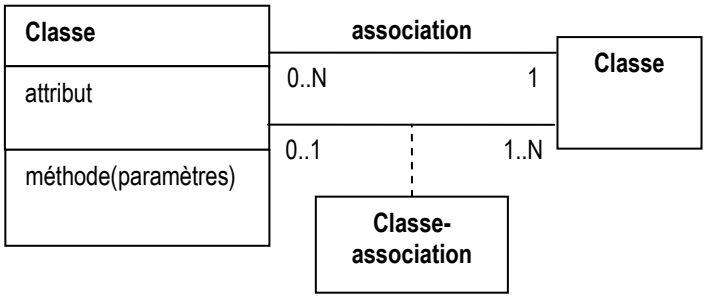
1. Le système affiche la liste des personnes inscrites à cette réunion.

4 Analyse des exigences

Les exigences sont généralement classées en deux familles : exigences fonctionnelles (celles qui sont décrites par les cas d'utilisation) et exigences techniques (quelquefois appelées « non-fonctionnelles »). On se concentre sur les cas d'utilisation.
L'analyse est implicitement descendante, sans vraiment le formaliser, on parlera d'une décomposition du système en niveaux : système, sous-système, composant logiciel ou matériel. De manière générale, l'analyse des exigences d'un niveau consiste à traduire les exigences exprimées globalement en exigences sur le constituant auquel elles sont allouées et de les rendre significatives pour le niveau considéré.
Dans le processus unifié, on retient le modèle de décomposition suivant : le paquetage, le « classifier » (classes, associations, interfaces), l'élément. Le processus unifié présente l'analyse comme le passage d'une vue externe du système (formulée dans le langage du client) à une vue interne (formulé dans le langage du développeur). Dans cette vision, l'analyse est vue comme un premier jet de la conception. Sans être totalement en désaccord avec ce glissement (« les objets d'analyse deviennent des objets de conception »), nous cherchons à établir une démarcation plus tranchée entre analyse et conception. L'objectif est d'avoir une analyse statique du système (sous la forme d'un diagramme de classes simplifié, appelé aussi diagramme de domaine) et une analyse dynamique (sous la forme de diagrammes de collaboration : un par cas d'utilisation).

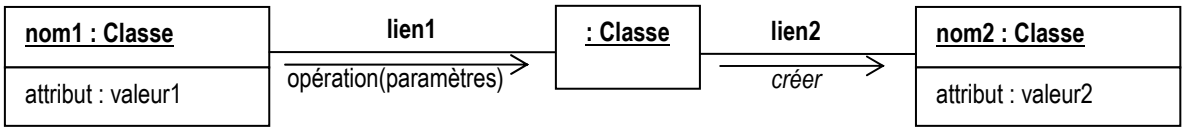
4.1 Qu'utilise-t-on du modèle entité-association

Le minimum du modèle entité-association (représenté dans le méta-modèle ci-dessous) est présenté aux élèves: entité (classe), association, classe-association, rôle, attribut et méthode (non typées).



4.2 Qu'utilise-t-on des diagrammes de collaboration

Les diagrammes de collaboration sont le support de l'analyse dynamique. En UML, il y a 2 niveaux pour un diagramme de collaboration : classe et instance. On considère une version simplifiée des diagrammes de collaboration UML, niveau instance, qui ne font apparaître ni les rôles, ni la chronologie des échanges, ni certaines subtilités comme la possibilité d'isoler une instance dans un paquet d'instances. Ainsi, nos diagrammes de collaboration sont concrètement des diagrammes d'objets sur les liens desquels on indique par une flèche étiquetée un message échangé à un moment non précisé de l'exécution.
Ceci est représenté dans le méta-modèle ci-dessous :



En toute rigueur, il ne s'agit pas tout à fait d'un diagramme de collaboration UML puisqu'on utilise des pointillés au lieu des paquets d'instances et des croix pour signifier la destruction.

On utilise aussi deux stéréotypes : l'acteur (représenté par l'icône « stickman ») et l'interface homme-machine (représentée par une icône représentant une fenêtre).

4.3 Méthode de travail

Après une présentation très rapide des concepts ci-dessus, les élèves analysent les cas d'utilisation de l'agenda suivant la méthode ci-dessous :

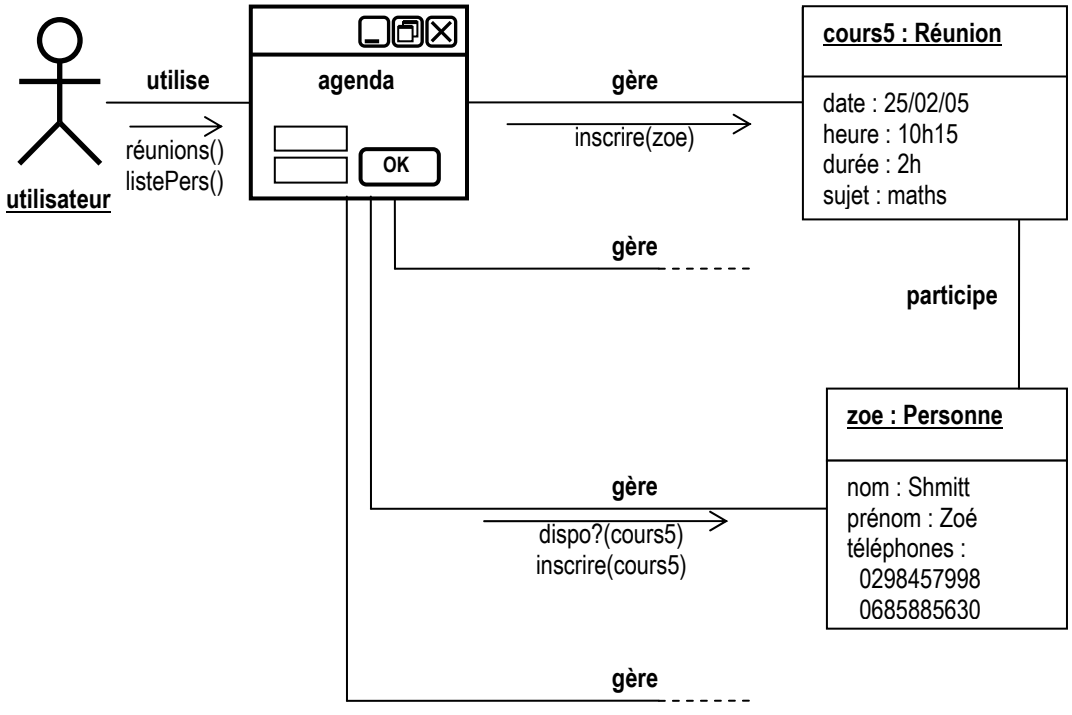
Etape 1 : constructions des schémas d'instances
On essaye de déterminer pour chaque cas d'utilisation :
- les instances qui se dégagent du cas d'utilisation
- les attributs des instances : noms et valeurs associées
- les liens entre les instances
- les attributs des liens : noms et valeurs associées
- le rôle des instances concernées par un lien
- les demandes de service à une instance : une flèche étiquetée par la requête pointe vers l'instance sollicitée ; l'ordre des sollicitations n'a pas d'importance

Par défaut (i.e. sauf mention explicite), le schéma indique l'état à la fin normale du cas d'utilisation.

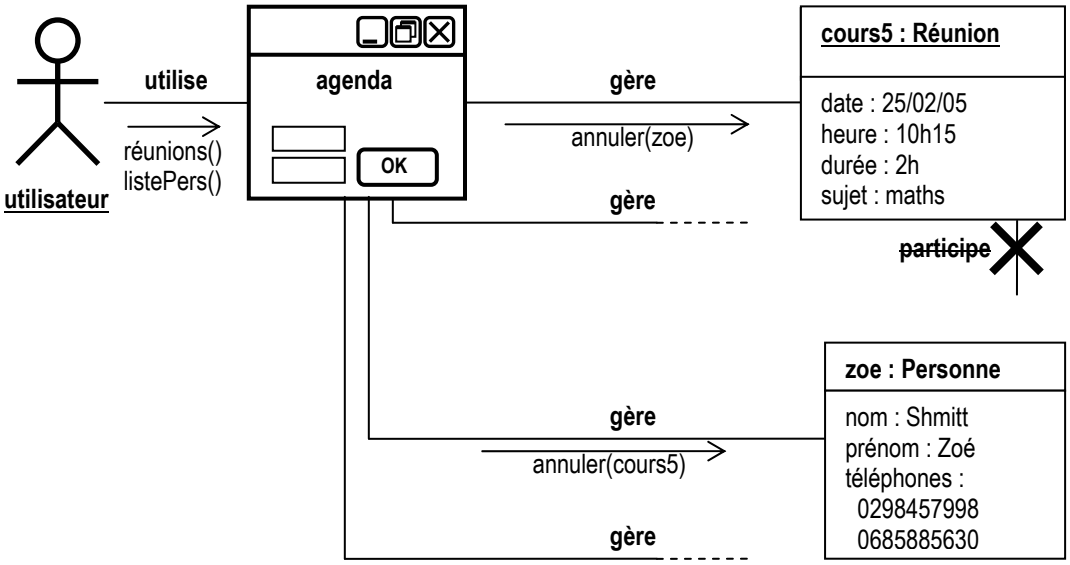
Etape 2 : construction du diagramme de classe
On abstrait les schémas d'instances et on les rassemble dans un diagramme unique :
- abstraction des instances : classes
- abstraction des valeurs d'attributs : noms et types des valeurs associées
- abstraction des liens entre les instances : associations, multiplicités et rôles
- abstraction des attributs de liens : classes-associations
- abstraction des demandes de service :
  - la classe sollicitée se retrouve munie d'une opération correspondant à la requête
  - cette opération est caractérisée par d'éventuels paramètres (quand la requête concerne une tierce instance), et par un éventuel type de résultat (quand la requête implique une réponse)
  - si la sollicitation concerne le lien (création, suppression du lien), on peut envisager de mettre l'opération dans une classe-association

4.4 Un exemple d'analyse

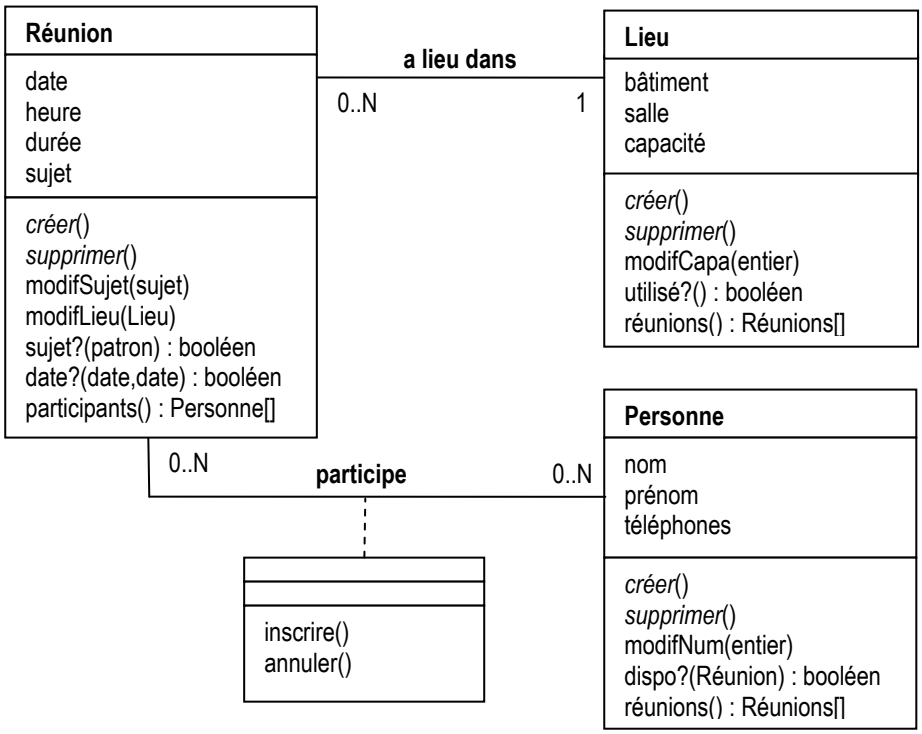
4.4.1 Cas d'utilisation « Participation à une réunion : inscription »



4.4.2 Cas d'utilisation « Participation à une réunion : annulation »



4.4.3 Diagramme de classes (diagramme du domaine) de l'agenda



5 Conception

« La conception d'un produit est composée de deux grandes phases : conception préliminaire et détaillée. La conception préliminaire établit les capacités du produit et son architecture, ce qui comprend la décomposition du produit, l'identification des composants, les états et les modes du système, les interfaces majeures entre composants et les interfaces externes du produit. La conception détaillée définit précisément les capacités et la structure des composants » [CMMI02].

L'activité de conception est de loin la plus difficile à appréhender. Deux points de vue sont abordés :

- un modèle d'architecture en trois couches. Ces couches prennent respectivement en charge les interactions avec les utilisateurs, les traitements métiers de l'application et l'accès aux sources de données (fichiers, bases de données, …) ;
- la décomposition hiérarchique du système en paquetages (composants), en insistant sur les critères de décomposition (par domaine fonctionnel, par acteur, par unité de lieu, par enchaînement temporel, ...).

Chaque composant pourrait être conçu selon les trois niveaux architecturaux et les interfaces entre composants lieraient à la fois les couches intra et inter-composants. Cependant, la conception est simplifiée comme suit :

- toutes les données du système sont regroupées dans un paquetage « Base de données » ;
- les dépendances entre paquetages sont réduites au minimum, à la limite aucun paquetage n'expose de services aux autres paquetages et tous les paquetages dépendent du paquetage « Base de données » ;

- la conception opère globalement pour les données : du diagramme de classes au modèle logique des tables, du modèle logique à l'implantation ; alors qu'elle opère localement pour les autres paquetages : un modèle de traitements et un modèle de présentation est établi par paquetage.

5.1 Qu'utilise-t-on en conception de données

Le modèle retenu est le modèle relationnel dans sa forme la plus simple: les données sont stockées en lignes et en colonnes. Une table a un nom et un ensemble de colonnes, chaque colonne ayant un nom et un type. Un n-uplet (ligne) est une collection de champs, chaque n-uplet est identifié par sa clé primaire. Une « clé primaire » présente dans une autre table est appelé clé étrangère. L'égalité des valeurs d'une clé primaire et d'une clé étrangère permet la jointure de deux lignes.
On n'utilise pas de système de gestion de base de données, mais Microsoft Excel. Un classeur est une base de données, une feuille est une table, une ligne est un n-uplet.

5.2 Qu'utilise-t-on en conception de traitements

Requêtes (en SQL) et procédures (en Basic) forment les constituants élémentaires de traitements. Ces constituants sont directement conçus à partir des diagrammes de collaboration issus de l'analyse.
Le modèle de conception sous-jacent des requêtes est l'algèbre relationnelle, au travers de Microsoft Query. Query fournit un environnement de requêtes capable de travailler avec Excel comme source de données (aussi bien qu'Access ou Oracle). Query permet la définition et l'exécution de requêtes au moyen d'un langage mixant :
- une notation textuelle issue de QBE (Query By Example) pour les opérations de projection et de restriction ;
- une notation graphique « style Access » pour le produit cartésien et les jointures ;
- des items de menus et des boîtes de dialogue pour les opérations de tri, de groupe et les fonctions de calcul.

Le SQL généré peut être visualisé, ce qui permet à certains élèves de l'apprendre « en faisant ».
Le modèle de conception des procédures écrites en Microsoft Visual Basic (VB, bien interfacé avec Excel et Query) est l'algorithmique procédurale, en commentant les instructions VB.

5.3 Qu'utilise-t-on en conception de présentation

Le modèle de présentation est celui de Windows à base de formulaires ou contrôles OCX (composants avec représentation visuelle). Un formulaire est une portion d'écran employée pour présenter des informations à l'utilisateur ou pour lui permettre de saisir des données. La façon la plus simple de définir l'interface utilisateur d'un formulaire consiste à placer des contrôles OCX sur sa surface. Pour la plupart, les contrôles OCX exposent des propriétés qui permettent de définir leur apparence (couleur d'avant-plan et d'arrière-plan, taille, emplacement, police d'affichage du texte, etc.). Les contrôles OCX et les formulaires peuvent également exposer des propriétés qui définissent la manière dont le contrôle OCX interagit avec l'utilisateur, ainsi que les informations sur lesquelles il doit opérer au moment de l'exécution.

5.4 Qu'utilise-t-on en conception d'architecture

Un modèle en couches est un modèle d'architecture. MVC (Model-View-Controller) et ses variantes reste l'architecture la plus employée pour relier les couches. Le modèle d'architecture enseigné est en réalité un mélange d'analyse et de conception.
Le modèle d'analyse est issu du Processus Unifié, où toute classe appartient à un des trois stéréotypes de base : « frontière », « entité », « contrôleur ». Une classe « frontière » modélise l'interaction du système et de ses acteurs et représente souvent des abstractions de fenêtres, de formulaires, ... Une classe « entité » sert à modéliser les informations et le comportement d'un phénomène ou d'un concept ; dans la plupart des cas elles sont dérivées du diagramme de classe (ou diagramme de domaine). Une classe « contrôleur » représente la coordination, le séquencement, les transactions et le contrôle d'autres objets, ainsi que les calculs complexes ne pouvant être liés à aucune entité spécifique.
Le modèle de conception établit une correspondance transparente entre les artefacts de conception et les constructions des outils d'implémentation utilisés : les frontières sont des contrôles OCX et des formulaires VB, les entités sont des tables Excel, les contrôleurs sont des requêtes Query ou des procédures VB. La spécification d'un artefact de conception utilise le même langage que l'outil d'implémentation, si bien que les opérations, paramètres, types, etc., sont spécifiés dans la syntaxe de l'outil employé.
Gérer la décomposition du système en relation avec les couches est hors de portée du cours. On se contente de respecter la décomposition en paquetages issue des cas d'utilisation. On élude ainsi une des grandes difficultés de la conception : allouer réellement les exigences à des composants logiciels indépendamment de la manière dont les exigences ont été attribuées aux artefacts d'analyse.

5.5 Apprentissage de la conception

5.5.1 Données

La base de la conception relationnelle est enseignée de manière magistrale, illustrée par l'agenda :
- une entité (classe d'analyse) donne naissance à une table (feuille Excel) ;
- une relation 1-N donne naissance à une table de liens (feuille Excel) ou à une clé étrangère (colonne) ;
- une relation M-N donne naissance à une table de liens (feuille Excel);
- une classe-association donne naissance à une table (feuille Excel).

Les copies d'écran ci-dessous représentent le modèle des tables de l'agenda.

TablesAgenda.xls:1

	A	B	C	D	E	F
1	idP	nom	prenom	tel		
2	1	COUPET	Grégory	681016958		
3	2	LANDREAU	Mickaël	681016210		
4	3	ABIDAL	Eric	681018061		

Personne | Lieu | Reunion | Participe

TablesAgenda.xls:2

	A	B	C	D	E	F	G
1	idL	batiment	salle	capacite			
2	1	LC	1	45			
3	2	LC	2	25			
4	3	LC	3	30			

Personne | Lieu | Reunion | Participe

TablesAgenda.xls:3

	A	B	C	D	E	F	G
1	idR	date	heure	duree	sujet	idL	
2	1	14/03/2005	14:30	2h	Préparation	1	
3	2	22/03/2005	14:30	2h30	Entrainement	1	
4	3	23/03/2005	8:00	1h	Bilan	3	

Personne | Lieu | Reunion | Participe

TablesAgenda.xls:4

	A	B	C
1	idP	idR	
2	1	1	
3	1	2	
4	1	3	
5	2	1	
6	4	4	
7	20	1	
8	4	6	
9	4	7	
10	5	5	
11	5	7	
12	6	2	
13	10	1	
14	10	2	
15	10	6	
16	10	7	
17	12	1	
18	13	1	
19	13	2	
20	14	1	
21	14	2	
22	14	3	

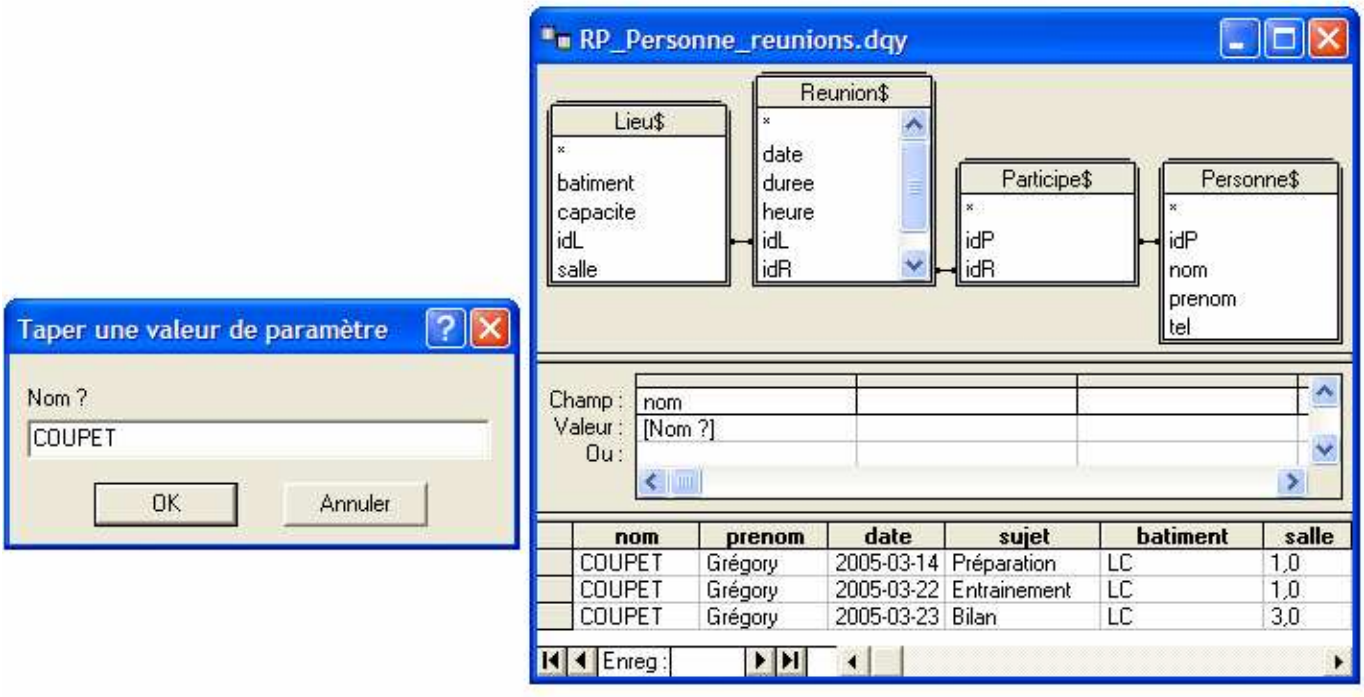
Participe

5.5.2 Traitements

La conception (et l'exécution) des requêtes sont enseignées à partir des diagrammes de collaboration de l'agenda à l'aide de petits exercices utilisant :

- la projection et la restriction (ex : Quelles sont les personnes qui portent le prénom David ou Olivier ?)
- la gestion des requêtes (sauvegarder, modifier, exécuter, sauvegarder le résultat, etc.)
- le paramétrage des requêtes (ex : Quels sont les lieux dont la capacité est comprise entre X et Y personnes ?)
- les requêtes portant sur plusieurs tables (ex : *Quelles sont les personnes qui participent à la réunion de numéro X* ?)

L'apprentissage est intuitif et assez aisé. A titre d'exemple, la requête ci-dessus (*Recherche des réunions auxquelles participe une personne*), la saisie de son paramètre et son résultat sont présentés dans les copies d'écran ci-dessous.

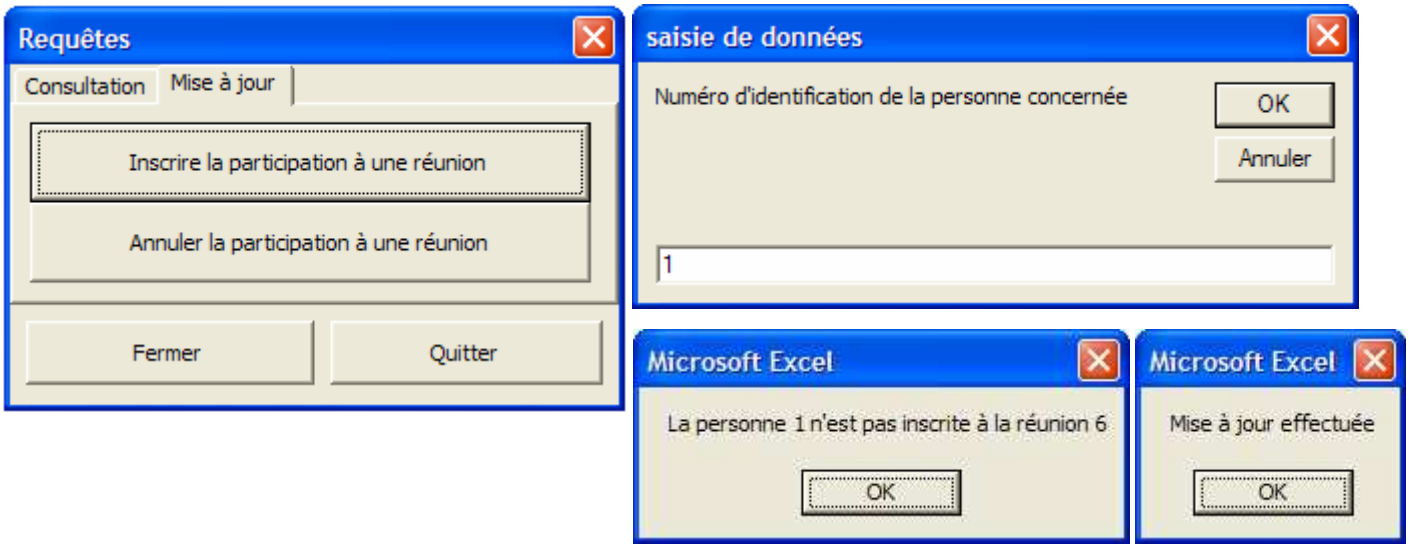


L'apprentissage de l'algorithmique est effectuée dans d'autres UE (une en CAML au 1er semestre, une en VB au 2ème). On se contente de noter quels cas d'utilisation ou quels diagrammes de collaboration ne peuvent être mis en œuvre avec Query, et ensuite de décrire l'algorithme général en pseudo-VB puis en VB commenté.
Voici, en exemple, la mise en œuvre du diagramme de collaboration « Participation à une réunion : annulation » (cf. 4.4.2) :

```
Private Sub CommandButton10_Click()
    'ouverture du classeur :
    Workbooks.Open "TablesAgenda.xls"
    'saisie du couple (idP,idR) à supprimer de Participe :
    numpchaine = InputBox("Numéro d'identification de la personne concernée", _
                          "saisie de données")
    numRchaine = InputBox("Numéro d'identification de la réunion", _
                          "saisie de données")
    nump = Val(numpchaine) 'transformation en nombre
    numR = Val(numRchaine) 'transformation en nombre
    Index = 0 'indice de parcours dans Participe
    indexLignePR = -1 'indice de la ligne à supprimer
    Do
        Index = Index + 1
        codeP = Worksheets("Participe").Range("A1").Offset(Index)
        codeR = Worksheets("Participe").Range("B1").Offset(Index)
        If codeP = nump And codeR = numR Then indexLignePR = Index
    Loop While codeP <> "" 'On cherche à atteindre la première ligne vide
    If indexLignePR = -1 Then
        MsgBox("La personne "+numpchaine+" n'est pas inscrite à la réunion "+numRchaine)
        Exit Sub
    End If
    If indexLignePR > 1 Then 'il y a plus d'une ligne dans la table
        'on remplace la ligne à supprimer par la dernière ligne :
        Worksheets("Participe").Range("A1").Offset(indexLignePR) = _
            Worksheets("Participe").Range("A1").Offset(Index - 1)
        Worksheets("Participe").Range("B1").Offset(indexLignePR) = _
            Worksheets("Participe").Range("B1").Offset(Index - 1)
    End If
    'on supprime la dernière ligne :
    Worksheets("Participe").Range("A1").Offset(Index - 1).Clear
    Worksheets("Participe").Range("B1").Offset(Index - 1).Clear
    'on ferme et on sauvegarde le classeur :
    Workbooks("TablesAgenda.xls").Save
    MsgBox ("Mise à jour effectuée")
    Workbooks("TablesAgenda.xls").Close savechanges = False
End Sub
```

5.5.3 Présentation

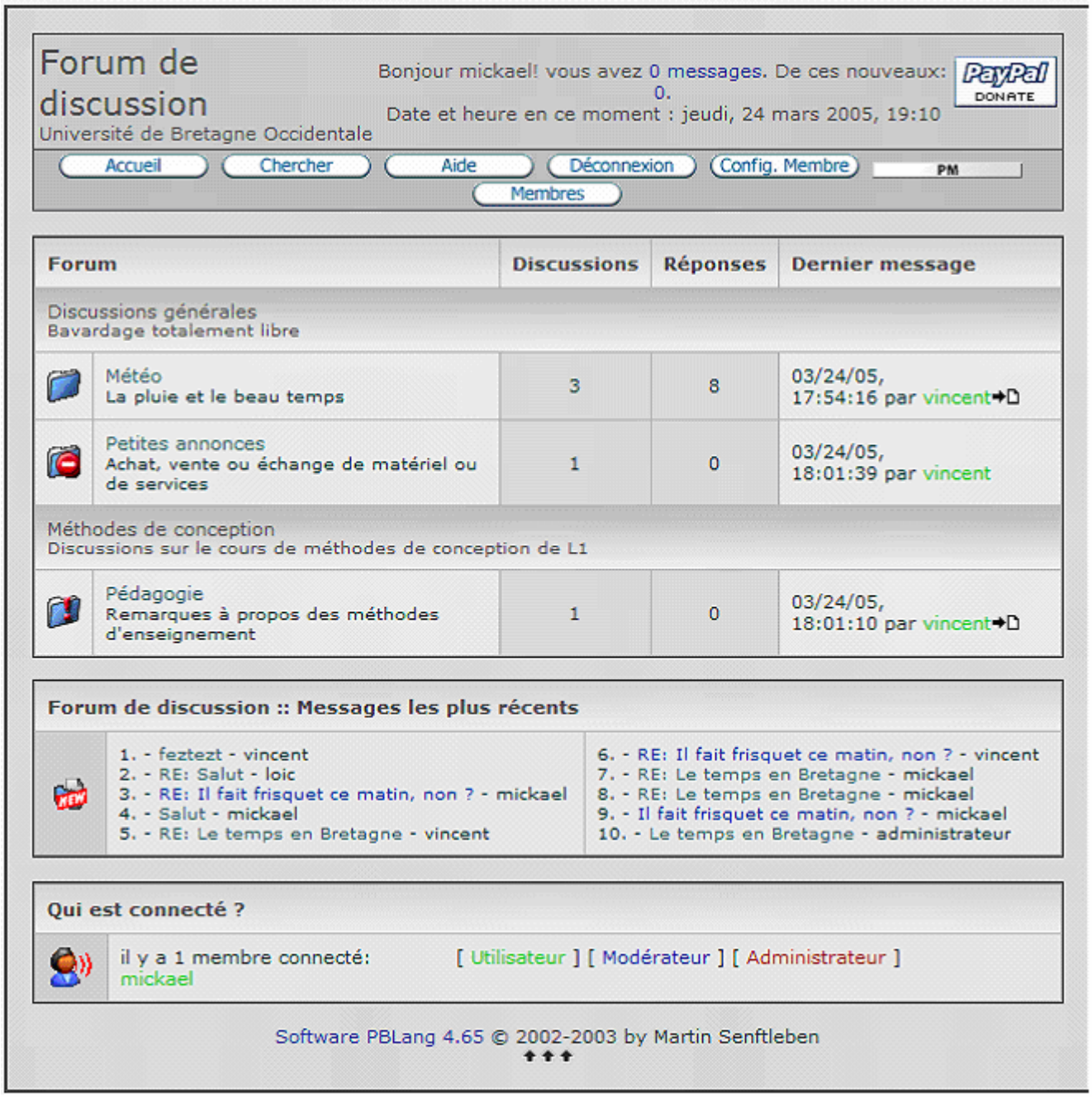
Le maquettage d'IHM est aisé en VB. A partir d'ébauches d'interface établies lors des cas d'utilisation ou des diagrammes de collaboration, les classes « frontières » sont réalisées sous la forme de formulaires ou de contrôles OCX dans le Concepteur VB. La liaison entres classes « frontières » et classes « contrôleurs » est établie à l'aide de boîtes de dialogues (cas des requêtes ou des procédures) ou avec des feuilles Excel (cas d'un résultat multi-ligne d'une requête).
Les copies d'écran ci-dessous illustrent une classe « frontière » (la fenêtre Requêtes), le paramétrage en entrée (la fenêtre saisie de données) et les différents résultats possibles (erreur : la personne n'était pas inscrite à la réunion ; succès : la mise à jour est effectuée) pour le diagramme de collaboration « Participation à une réunion : annulation » (cf. 4.4.2).



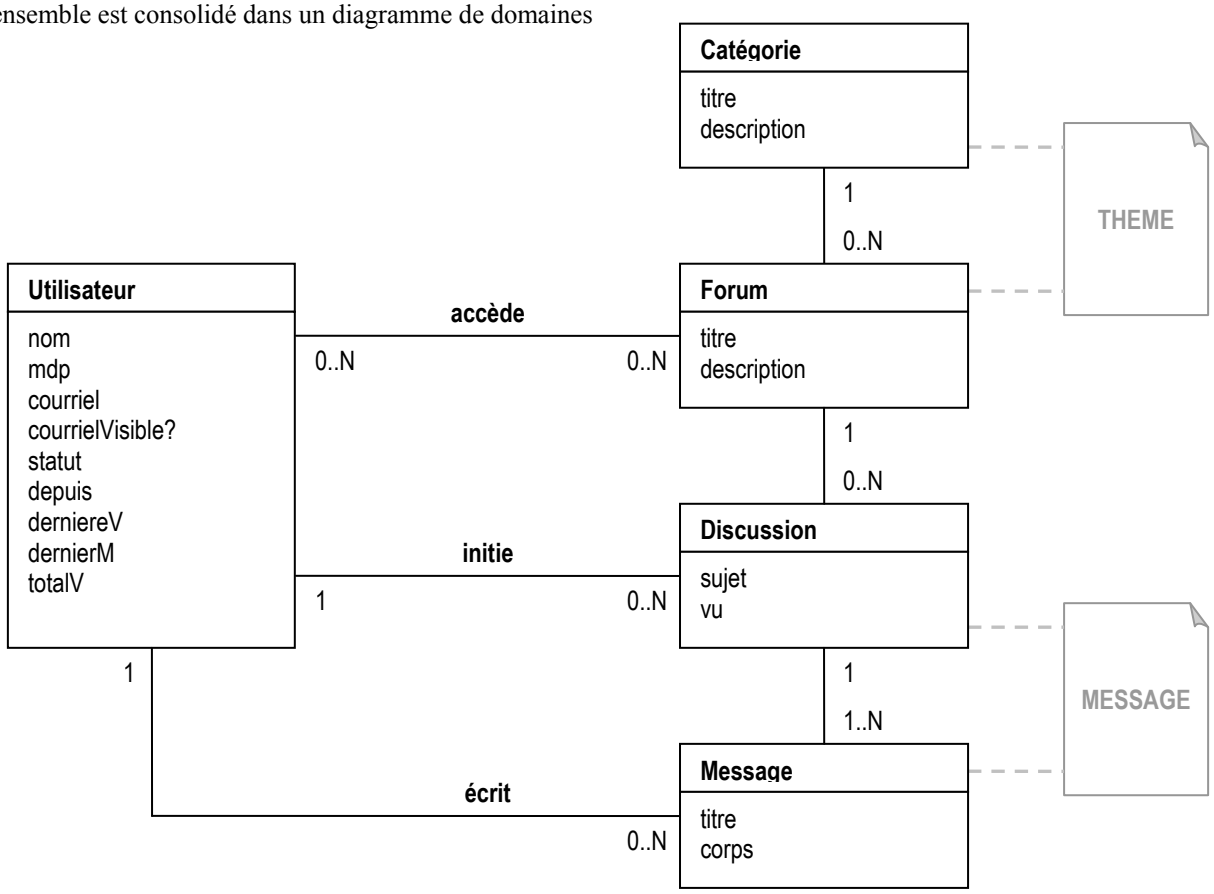
5.6 De l'usage de la rétro-activité

Les activités de conception présentée dans les § précédents, même si elles donnent aux élèves une « idée globale » de la conception, sont perçues comme indépendantes. De plus, face à un problème complexe, l'expérience que nous avons d'enseigner la conception montre que les élèves (du L1 au M2) sont assez démunis pour établir une conception (alors que les autres phases du cycle de vie posent moins de problèmes).
Conception et réalisation sont étroitement liées : il faut réaliser à partir d'une ébauche de conception pour pouvoir raffiner la conception à partir des problèmes rencontrés en réalisant, ceci de manière cyclique. Le paradoxe est le suivant : « On ne conçoit bien que ce qu'on a déjà réalisé ».
Par ailleurs, ce cours omet volontairement les aspects liés à la programmation (de toutes façons, le temps manquerait pour réaliser le projet Forum tel qu'il a été spécifié, analysé et conçu par les élèves).
La réponse apportée à ces deux difficultés utilise la pédagogie inductive de type rétro-ingénierie de Steven Pinker, qui revient à analyser, décomposer pas à pas une activité afin de se demander comment elle est mise en œuvre dans une situation donnée [Pin00].
Un forum de discussion (correspondant à peu près au cahier des charges du projet à réaliser) est fourni aux élèves. L'apprentissage de la conception passe par la rétro-conception par les élèves de ce forum (qu'ils n'ont pas réalisés). Il s'agit de partir de l'exploitation du concret pour expliciter les démarches techniques et théoriques.

Le travail est réalisé de la manière suivante. Les élèves explorent librement le forum pendant une demi-séance (1 heure environ). Les grandes fonctionnalités du forum sont regroupées en paquetages, ce qui forme la décomposition hiérarchique. Les interfaces du forum (la matérialisation des classes « frontières ») sont regroupées par paquetage.
A titre d'illustration, la copie d'écran ci-dessous synthétise une partie des interfaces du forum.



A partir de ces interfaces, les élèves induisent les classes « entités » et leurs associations.
L'ensemble est consolidé dans un diagramme de domaines

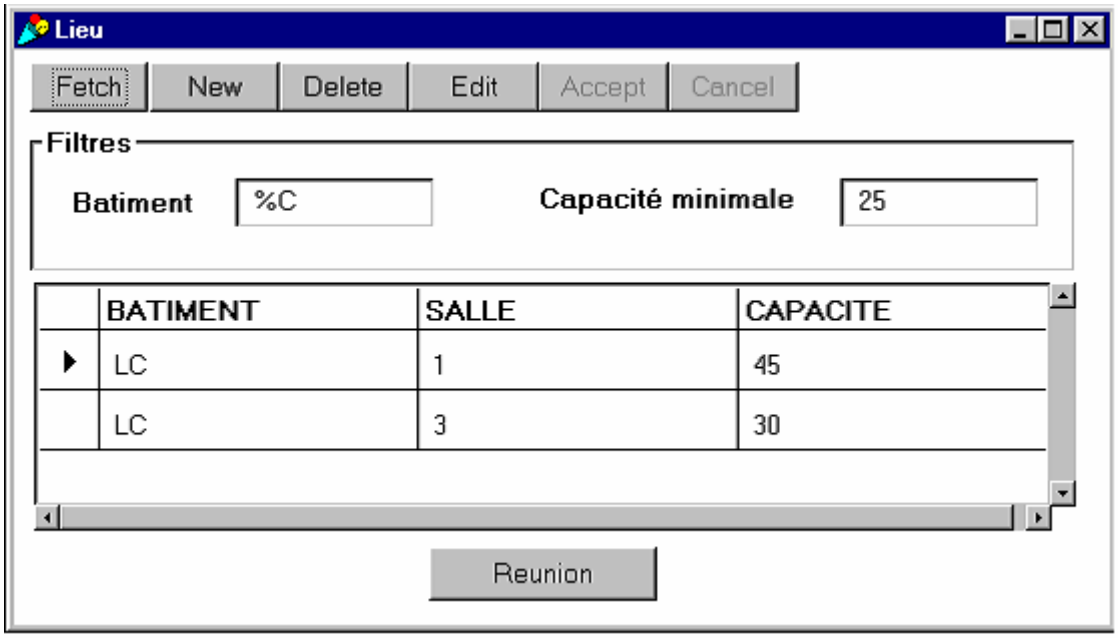


Ensuite, à partir de ce diagramme de domaine, les requêtes et les procédures correspondant aux fonctionnalités (celles découvertes lors de l'exploration) sont définies (quelques-unes d'entre elles sont cependant réalisées, car faire rétro-réfléchir a ses limites ...).

Pour réaliser ces opérations, la démarche suivante est proposée et illustrée sur l'agenda, puis expérimentée par les étudiants sur le forum :

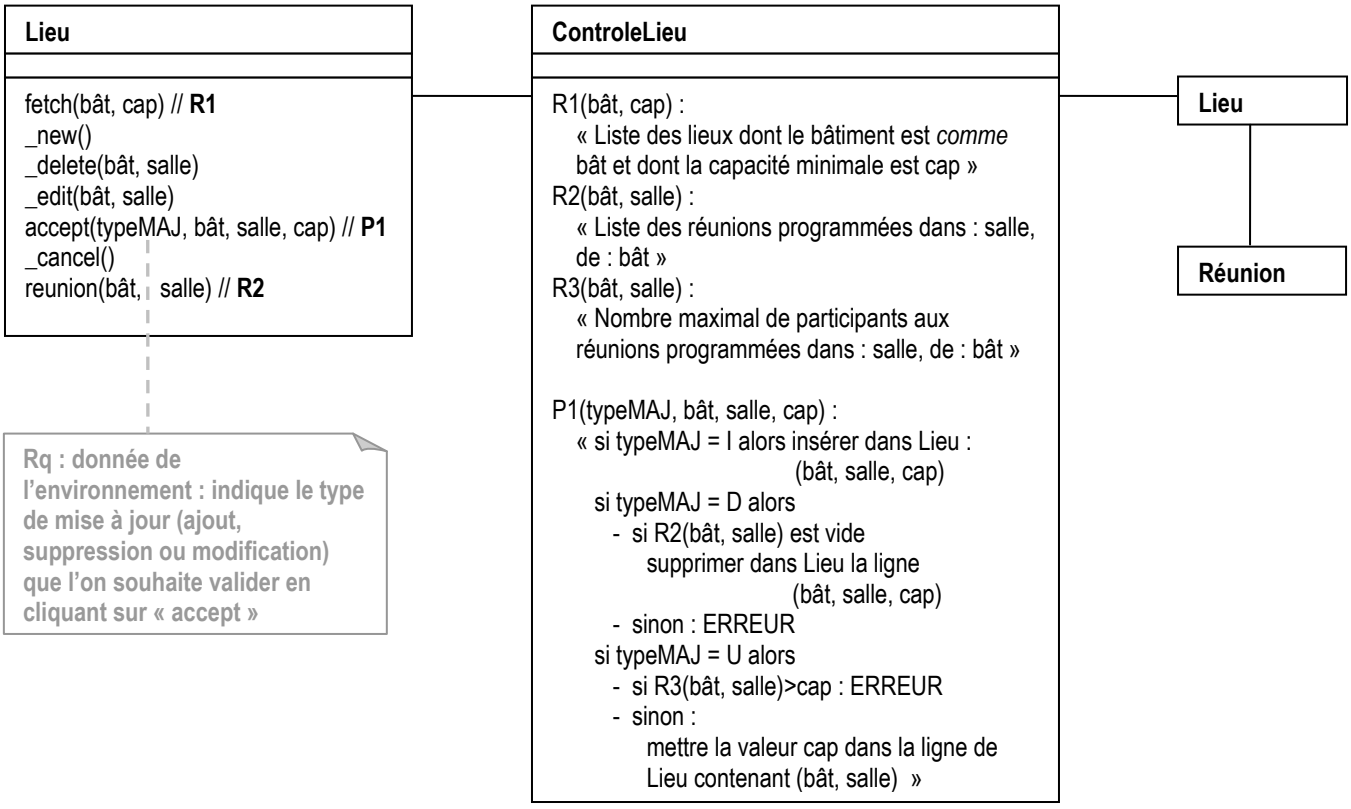
- Nommer les classes « frontières » à partir des éléments d'interface, comme une fenêtre ou un panneau de fenêtre
- Caractériser l'environnement d'exécution comme la valeur d'un identifiant d'utilisateur, ou la date courante
- Identifier les opérations et leurs paramètres éventuels :
    - Identifier les zones actives qui déclenchent des opérations : lien hypertexte, bouton, onglet,…
    - Identifier les zones de saisie/consultation qui correspondent à des paramètres d'opérations : champ, zone de texte, bouton radio, check box, item de liste sélectionné,…
- Déterminer la nature des opérations
    - Traitements : opérations nécessitant un accès aux données
    - Opérations de navigation : opérations ne nécessitant pas la manipulation de données autres que celles contenues dans l'environnement :
        - Opérations activant des instances d'autres classes frontières
        - Opérations mettant à jour l'état de l'instance de classe frontière courante
    - Convention de nommage : le nom des opérations de navigation commence par « _ »
- Pour les traitements : identification des entités sollicitées
- Description des contrôleurs
    - Identifier les contrôleurs qui font le lien entre les classes frontières et les entités sollicitées
    - Identifier et spécifier (en français) les requêtes (simples consultations des données) et les procédures de mises à jour (ajout, suppression, modification des données)
- Réalisation des traitements par les requêtes et procédures des contrôleurs
    - Dans les classes frontières : mettre en commentaire des traitements le nom des requêtes et procédures des contrôleurs qui permettent de les réaliser

L'illustration de cette démarche de rétro-conception sur l'agenda s'appuie sur une série de copies d'écran d'un outil existant. Par exemple, la fenêtre de gestion des lieux dans l'agenda permet d'effectuer la mise à jour des lieux : création, modification et destruction.
Elle permet également d'effectuer des recherches sur les critères Bâtiment et Capacité. L'exemple suivant illustre la recherche de toutes les salles dont le bâtiment se termine par la lettre 'C' et dont la capacité est supérieure à 25 personnes.



Le bouton « Reunion » permet d'accéder à la liste des réunions qui se déroulent dans le lieu sélectionné. Aucune mise à jour (ajout, suppression, modification) n'est effective tant que l'utilisateur n'a pas cliqué sur « Accept ». De plus, si une mise à jour n'a pas été validée par « Accept » et que l'utilisateur tente une autre mise à jour, un message d'erreur l'invite à décider d'abord s'il veut valider la mise à jour (« Accept ») ou l'annuler (« Cancel »).

L'analyse de cette fenêtre selon la démarche proposée amène au diagramme suivant :



A partir de copies d'écran de l'agenda, les élèves effectuent progressivement une partie de la rétro-conception.
Dans la suite du cours (et de l'article), le travail des élèves se poursuit sur le forum fourni qui, rappelons-le, présente des écarts avec le cahier des charges initial. L'exemple de l'agenda est donc laissé de côté.

6 Tests

L'objectif est de sensibiliser à la démarche de test du développement d'un logiciel construite sur le modèle suivant :
- les unités de réalisation du logiciel sont testées individuellement lors de tests unitaires ;
- les unités de réalisation du logiciel sont assemblées progressivement, cet assemblage est testé lors de tests d'intégration ;
- le logiciel constitué est testé lors de tests de qualification (appelé aussi validation).

Le point fondamental est d'utiliser les tests pour faire ressortir l'importance des exigences, de leur respect et la traçabilité.

6.1 Qu'utilise-t-on ?

On n'abordera ni les types de tests (comportement, sûreté, performances, ...) ni les techniques de test (analyse, observation, ...). On cherche seulement à sensibiliser aux tests fonctionnels dans l'optique d'une vérification des capacités du logiciel, c'est à dire de vérifier si le composant testé remplit bien les missions qui lui sont attribuées. Le test définit :

- les données d'entrées nominales,
- les actions opérateur prévues,
- les résultats attendus,
- l'ordonnancement des fonctions.

L'organisation des tests repose sur un plan de test, organisé comme suit :

- La qualification est divisée en différentes opérations. On l'appelle aussi recette du logiciel.
- Une opération de qualification consiste à valider un ensemble de fonctionnalités, de services, de documentation ou de contraintes du système. Une opération est structurée en étapes.
- Une étape est décomposée en actions. Les actions définissent les fonctionnalités à valider pour chaque étape.
- Chaque action est constituée d'essais. Les essais de chaque action doivent vérifier la conformité des résultats avec les exigences.

Le plan de tests est fourni aux élèves. Il est directement construit à partir de leur cahier des charges. Il sert de support à la présentation des concepts. La décomposition des opérations en étapes et en actions correspond généralement aux étapes d'assemblage dans lesquels les composants sont testés ; elle doit faire ressortir l'architecture du logiciel.

6.2 Méthode de travail

A partir du plan identifiant les opérations, les étapes, les actions et les essais (ce qui définit tous les chemins possibles de tests), les élèves doivent vérifier les différents chemins.

La chronologie des activités d'exécution de tests est la suivante :

- exécution des tests, conformément aux prévisions, avec détection et collecte des défauts,
- formalisation et conservation des résultats,
- rapport de test.

L'exécution des tests selon le plan a pour rôle de faire comprendre la nécessité d'une organisation (d'une stratégie) de tests. La description des tests n'est pas fournie. Pour certains tests, elle sera rédigée par les élèves. Afin de rendre saillant la relation entre exigences et tests, les cas d'utilisation fourniront la matière pour la description des tests.

Les élèves exécutent le plan et rédigent les description de tests selon les règles suivantes :

- lorsque la fonctionnalité demandée dans le cahier des charges est validée, on se contente d'approuver l'action correspondante dans le plan de test ;
- lorsque la fonctionnalité est manquante ou défectueuse, les élèves doivent rédiger la description des essais, incluant des essais qui prouveraient la validité comme des essais qui démontrent l'incapacité.

6.3 Un exemple de tests

6.3.1 Opérations

Qualification **Titre officiel :** Forum Electronique	**Approbation** **(nom et signature)**	**Date**	**Report pour remarques éventuelle(s)**
Opération 1 : Validation du mode utilisateur			
Opération 2 : Validation du mode administrateur			
Opération 3 : Validation de la documentation			
Opération 4 : Validation des produits			
Opération 5 : Validation des services			

6.3.1.1 Validation du mode utilisateur (O1)

Titre officiel : Forum Electronique **Nom de l'opération à recetter :** *Validation du mode utilisateur (O1)*	**Approbation** (mettre une croix)	**Date**	**Report pour remarques éventuelle(s)**
Etape 1 :Validation de la connexion type utilisateur			
Etape 2 :Validation de la navigation type utilisateur			
Etape 3 :Validation de la création type utilisateur			
Etape 4 :Validation de la recherche type utilisateur			
Etape 5 :Validation de la consultation type utilisateur			
Etape 6 :Validation de la sauvegarde type utilisateur			
Etape 7 :Validation de la modification type utilisateur			
Etape 8 :Validation de la suppression type utilisateur			

6.3.1.1.1 Validation de la recherche type utilisateur (E4)

Etape n° : 4 Validation de la recherche type utilisateur	Approbation (mettre une croix)	Date	Report pour remarques
Action 1 : Validation de la recherche sur les noms de thème			
Action 2 :Validation de la recherche sur les titres de message			
Action 3 :Validation de la recherche sur les auteurs de message			
Action 4 :Validation de la recherche sur les dates de création de message			

7 Evaluation

L'évaluation porte pour 2/3 sur le projet réalisé (1/3 en commun et 1/3 en examen individuel) et pour 1/3 sur les connaissances enseignées (lors d'un examen sur table). Ces évaluations seront complétées par un questionnaire dont les résultats seront disponibles lors du colloque.
Afin d'illustrer le travail réalisé, un exemple de production d'élève (parmi les meilleures) est donné pour chacune des livraisons demandées.

7.1 Cas d'utilisation

IDENTIFICATION

Titre : Connexion

DESCRIPTION DES SCENARIOS

Préconditions :

I. Le système est opérationnel.

Scénario stratégique : « connexion d'un utilisateur/administrateur »

1. L'utilisateur ouvre une session de travail sur un PC du département.
2. L'utilisateur peut se connecter en tant qu'administrateur ou non.
3. L'utilisateur se déconnecte.

Scénario principal A : Connexion d'un utilisateur/administrateur
Garantie en cas de succès :

I. Un nouvel utilisateur est connecté.

1. L'utilisateur entre son identifiant.
2. Le système vérifie que l'identifiant n'est pas déjà utilisé.
3. Le système autorise la connexion.

Extensions possibles :

2a. Identifiant réservé.

1. Le champ « identifiant » est incorrectement saisi.
2. Le système affiche un message d'erreur
3. Le système efface ce champ.

Le scénario reprend en 2.

1-3a. Annulation

1. L'utilisateur annule la connexion.

Fin du scénario principal A, retour à l'étape 2 du scénario stratégique.

Scénario principal B : Connexion d'un administrateur
Garantie en cas de succès :

I. L'administrateur est connecté.

1. L'administrateur sélectionne le « mode administrateur ».
2. Le système affiche un nouveau champ.
3. L'utilisateur entre son identifiant et son mot de passe.
4. Le système autorise la connexion en tant qu'administrateur.

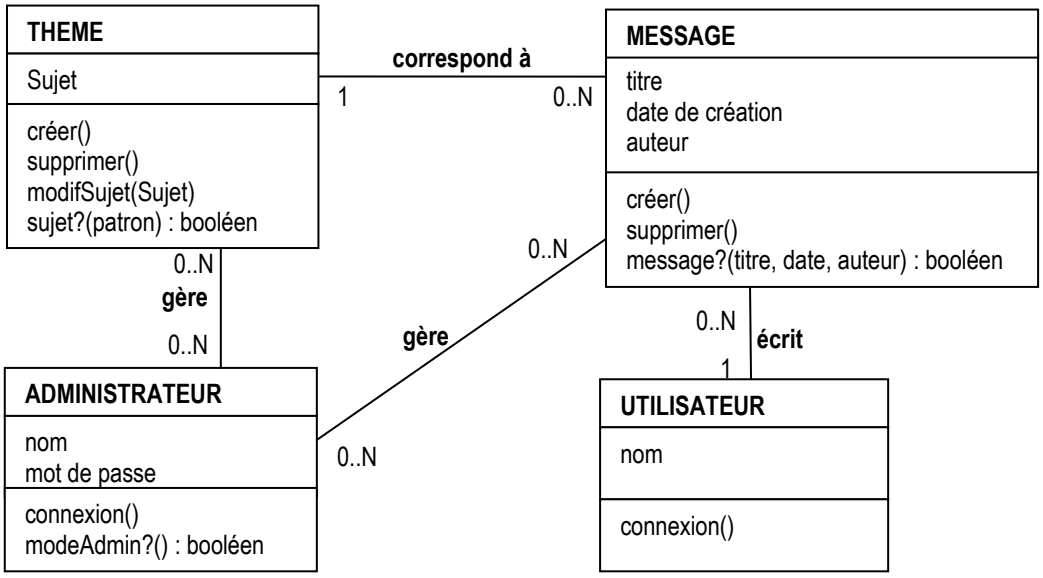
Extensions possibles :

3a. Identifiant ou mot de passe incorrect.

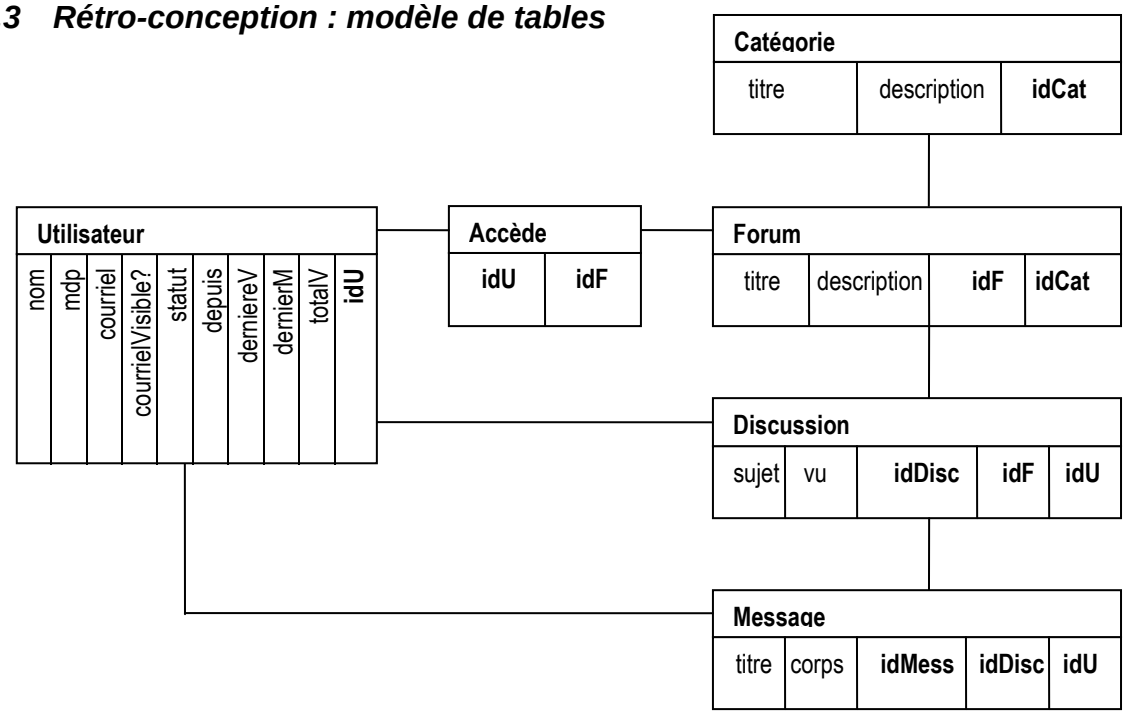
1. Un des champs est incorrectement saisi, ou les deux ne correspondent pas.
2. Le système affiche un message d'erreur
3. Le système efface le(s) champ(s) incorrect(s).

Le scénario reprend en 3.

7.2 Analyse : diagramme de classes



7.3 Rétro-conception : modèle de tables



7.4 Tests

O1	Validation du mode utilisateur	E4	Validation de la recherche type utilisateur	A2	Validation de la recherche sur les titres de message

FORUM Essai n° 1

Description :
Rechercher des messages à partir d'un mot contenu dans leurs titres, avec le type utilisateur

Conditions initiales :
Il existe plusieurs messages dans le forum

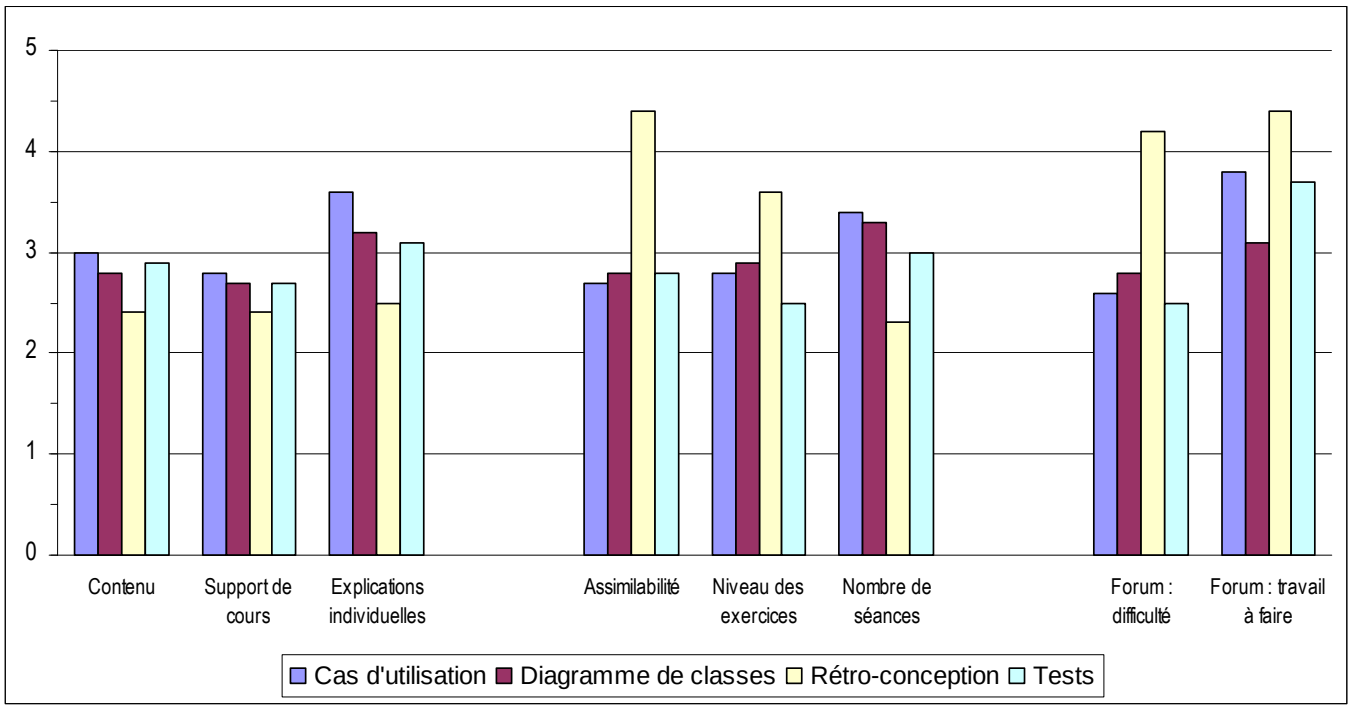
Vérifications à effectuer	Résultats attendus[1]	T	V
Recherche à partir d'un mot effectivement contenu dans le titre d'au moins un message	Affichage de la liste des messages dont le titre contient le mot recherché	X	X
Recherche à partir d'un mot qui n'apparaît dans aucun message (titre et corps)	Affichage d'une liste vide	X	X
Recherche à partir d'un mot qui n'apparaît dans aucun titre de message mais qui apparaît dans le corps d'au moins un message	Affichage d'une liste vide	X	

8 Premier bilan

Le bilan à chaud (à l'issue des 12 séances) semble assez bon. La majorité des élèves est assidue, et après quelques semaines se mettent à travailler en groupe. Les élèves jouent les rôles demandés et, dans l'ensemble, produisent les « déliverables » dans les délais. Le niveau des productions est très satisfaisant, surtout quand on considère que ce sont des élèves de première année, ayant peu d'UEs d'informatique.

Un questionnaire d'évaluation a été remis à l'issue de l'examen. Les élèves ont répondu à quatre séries de huit questions concernant respectivement les cas d'utilisation, les diagrammes de classes, la rétro-conception et les tests.
Dans chaque série, les trois premières questions concernent le contenu, le support de cours et les explications individuelles. Les réponses possibles sont codées par des chiffres qui vont de 1 (faible) à 5 (fort).
Les trois questions suivantes concernent l'assimilabilité et le niveau des exercices (de 1 : facile, à 5 : difficile), et le nombre de séances (de 1 : insuffisant, à 5 : trop nombreux).
Les deux dernières questions concernent le projet Forum et permettent d'évaluer la difficulté (de 1 : facile, à 5 : difficile) et le travail à faire (de 1 : trop court, à 5 : trop long).

Le graphique suivant présente la synthèse de cette évaluation :



[1] T : testé, V : validé

Une dernière série de questions permet de faire un bilan de l'enseignement. Le tableau suivant en présente la synthèse :

Bilan		1	2	3	4	5		Moyenne
Impression générale	mauvaise		1	5	4		bonne	3,3
Investissement personnel	faible	1	2	3	3	1	fort	3,1
Volume global horaire	insuffisant		1	6	2	1	excessif	3,3

Dans une rubrique du questionnaire, les étudiants ont pu exprimer leurs impressions. Le commentaire suivant résume bien l'opinion générale :

> « La partie la plus difficile est la rétro-conception. De plus le travail demandé par cette partie est trop long parce qu'il arrive pendant la période de révision des partiels. Beaucoup de travail pour seulement 2,5 crédits. Peut-être que 5 crédits seraient plus justes. »

La vraie évaluation ne pourra être effectuée que les années suivantes : ce cours est dispensé aux seuls élèves du parcours dit C de la licence informatique (l'ancien DEUG STPI). En L2 et L3, les cours d'algorithmique, d'objet, de base de données et de systèmes d'information (les plus proches du domaine) seront communs avec les autres parcours. On percevra alors peut-être l'intérêt de ce cours pour développer une vision plus globale (et espérons-le à long terme) du « développement » du logiciel.

9 Références

[Coc01] Alistair Cockburn, Rédiger des cas d'utilisation efficaces, Eyrolles, 2001

[CMMI02] CMMI for Systems Engineering and Software Engineering (CMMI-SE/SW, V1.1) Continuous Representation, http://www.sei.cmu.edu/pub/documents/02.reports/pdf/02tr011.pdf

[JBR99] Ivar Jacobson, Grady Booch, James Rumbaugh, The Unified Software Development Process, Addison-Wesley Longman, 1999

[Mey01] Bertrand Meyer, Software Engineering in the Academy, IEEE Computer, May 2001

[Pin00] Steven Pinker, Comment fonctionne l'esprit, Odile Jacob, 2000

[Tar98] Jacques Tardif, Intégrer les nouvelles technologies de l'information – Quel cadre pédagogique ?, ESF, 1998

[TIS02] TEMPO, la maîtrise du développement de systèmes informatiques, Thales Information System, 2002