Cahier Des Charges Technique

Version 1.0

Membres de l'équipe Hybrid:

Louis ANDRIET

Julien DASTUGUE

Sébastien LAROCHE

Michaël MADEGARD

Cédric MYLLE

Résumé :
Ce document a pour but de décrire les éléments techniques nécessaires à la mise en place du logiciel de planification de soutenances HIP.

Référence du document

Réf.	HYBCDCT001-1

Version du document

DATE	AUTEUR	DESCRIPTION	VERSION
28/10/2007	J. DASTUGUE	Création	0.1a
04/11/2007	M. MADEGARD	Relecture et correction	0.2a
11/11/2007	J. DASTUGUE	Ajout des parties introduction, présentation de l'architecture, description des briques utilisées et conclusion	0.3a
11/11/2007	L. ANDRIET	Ajout des parties diagramme de package et diagramme de séquence.	0.3b
11/11/2007	S.LAROCHE	Relecture et modifications	0.4a
11/11/2007	M. MADEGARD	Relecture et modifications	0.4b
11/11/2007	C. MYLLE	Relecture	0.8
11/11/2007	M. MADEGARD	Approbation de la version 1	0.9

Validation du document

DATE	AUTEUR	DESCRIPTION	VERSION
11/11/2007	C. MYLLE	Validation de la version 1	1.0

Table des matières

Table des matières **3**
Introduction **5**
1 Présentation de l'architecture logicielle **6**
1.1 Architecture logicielle **6**
1.1.1 Présentation d'une architecture 3-tiers J2EE 6
1.1.2 Présentation de l'architecture de HIP 8
2 Composant logiciel utilisé **10**
2.1 La couche d'accès aux données **10**
2.1.1 La base de données 10
2.2 La couche métier **11**
2.2.1 Le serveur d'application 11
2.2.2 L'authentification et l'interaction avec LDAP 13
2.2.3 Implémentation de JPA 14
2.2.4 L'algorithme de génération du planning 14
2.2.5 La lecture de fichier Microsoft Excel 15
2.2.6 La génération des convocations à partir d'un modèle 16
2.2.7 La génération de PDF 16
2.2.8 L'envoi de mail 17
2.3 La couche présentation **17**
2.3.1 Le client léger 17
2.3.2 Le client lourd 19
3 L'Analyse Conceptuelle **21**
3.1 Généralités **21**
3.2 Partie serveur HIP **22**
3.2.1 Paquetages Java 22
3.2.2 Paquetage « model » 22
3.2.3 Paquetage « session » 27
3.2.4 Paquetage « util » 28
3.2.5 Paquetage « web » 28
3.2.6 Paquetage « dao » 28
3.2.7 Paquetage « auth » 30

3.3 Client lourd 32
3.3.1 Paquetage « session » 32
3.3.2 Paquetage « ui » 32
3.3.3 Paquetage « login » 33
3.3.4 Paquetage « internationalisation » 34
3.3.5 Paquetage « util » 34
4 Les Use Case génériques 35
4.1 Généralité 35
4.2 USE CASE générique n°1 : Accéder à la <page> 35
4.3 USE CASE générique n°2 : Utiliser l'application du <client> 37
4.4 USE CASE générique n°3: Ajouter un XXX 39
4.5 USE CASE générique n°4: Modifier un XXX 42
4.6 USE CASE générique n°5: Supprimer un XXX 45
4.7 USE CASE générique n°6 : Visualiser un XXX 48
5 Diagrammes de séquence 51
5.1 Diagrammes de séquence de la version 1 du logiciel 51
5.1.1 Gestion des apprentis 51
5.1.2 Authentification et gestion des droits des utilisateurs 53
5.2 Diagrammes de séquence de la version 2 du logiciel 57
5.2.1 Gestion des Disponibilités 57
5.3 Diagrammes de séquence de la version 3 du logiciel 57
5.3.1 Gestion des plannings 57
5.3.2 Gestion des convocations 58
5.4 Diagrammes de séquence de la version 4 du logiciel 58
5.4.1 Impression 58
5.4.2 Export 58
5.4.3 Internationalisation 58
6 Répartition des rôles pour la V1 59
Conclusion 61
Glossaire 62
Table des illustrations 73

Introduction

Dans le cadre du projet de Génie Logiciel de notre troisième année de formation en Informatique et Réseaux, à l'école Ingénieurs 2000, nous allons réaliser, durant les prochains mois, un travail d'équipe afin d'aboutir à la création d'un logiciel permettant de générer automatiquement les plannings pour les soutenances de dossiers d'alternance et de mémoires d'ingénieurs.

Afin de suivre l'évolution constante du projet, différents rendus s'étalonnant du 10 octobre 2007 au 27 février 2008 seront indispensables. Le premier concerne le cahier des charges initial, servant de base au recensement des besoins du client. Après cela, un dossier de définition des outils de travail permettra de lui présenter les outils avec lesquels nous travaillons. Puis un cahier des charges graphiques sera également nécessaire afin de fournir un aperçu de l'ergonomie du logiciel. Enfin, les rédactions des cahiers des charges fonctionnel et technique permettront de présenter au client l'ensemble des fonctionnalités de son logiciel. La validation de ces documentations débouchera alors sur la création des différentes versions du logiciel, avant d'atteindre la version finale qui sera livrée au client.

Le présent document correspond au cahier des charges technique. Il décrit l'ensemble des éléments techniques nécessaires à la mise en place du logiciel de planification de soutenances HIP.

Dans un premier temps, il présentera l'architecture logicielle utilisée. Puis, chaque brique logicielle utilisée dans le projet sera déterminée. Pour finir, l'architecture de notre logicielle sera détaillée à l'aide diagramme de package.

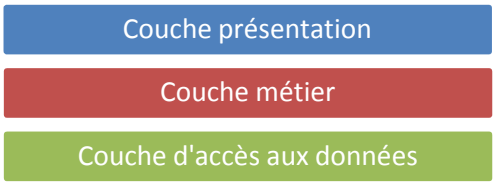
1 Présentation de l'architecture logicielle

Dans cette première partie, nous nous consacrerons à la présentation des technologies qui entreront dans le cadre de la fabrication du logiciel de notre projet.

1.1 Architecture logicielle

1.1.1 Présentation d'une architecture 3-tiers J2EE

L'architecture que nous proposons repose sur les principes d'une architecture 3-tiers. Elle permet de diviser notre logiciel en 3 couches :



En premier lieu, il y a la couche d'accès aux données. Elle correspond aux données qui sont destinées à être conservées.

En second lieu, il y a la couche métier. Elle s'interface entre la couche présentation et la couche d'accès aux données. Elle a pour but d'effectuer le traitement métier.

Pour finir, on trouve la couche présentation. Elle correspond à la partie de l'application visible et avec laquelle les utilisateurs vont interagir. Elle se présente sous la forme de clients - on parle d'interface Homme-Machine - qui permettent de transcrire à l'application les actions que veut mener un utilisateur.

HIP est une application qui est destinée à être déployée dans un environnement J2EE (*Java 2 Enterprise Edition*) définissant une norme proposée par la société Sun. Cette norme est portée par un consortium de sociétés internationales, visant à définir un standard de développement d'applications d'entreprises multi-niveaux, basées sur des composants.

J2EE facilite le déploiement d'architecture 3-tiers en fournissant des composants déjà implémentés à travers un serveur d'application. On peut citer les Servlets pour la couche présentation, les EJBs pour la couche métier et la technologie JPA qui assure la persistance des données dans la base de données.

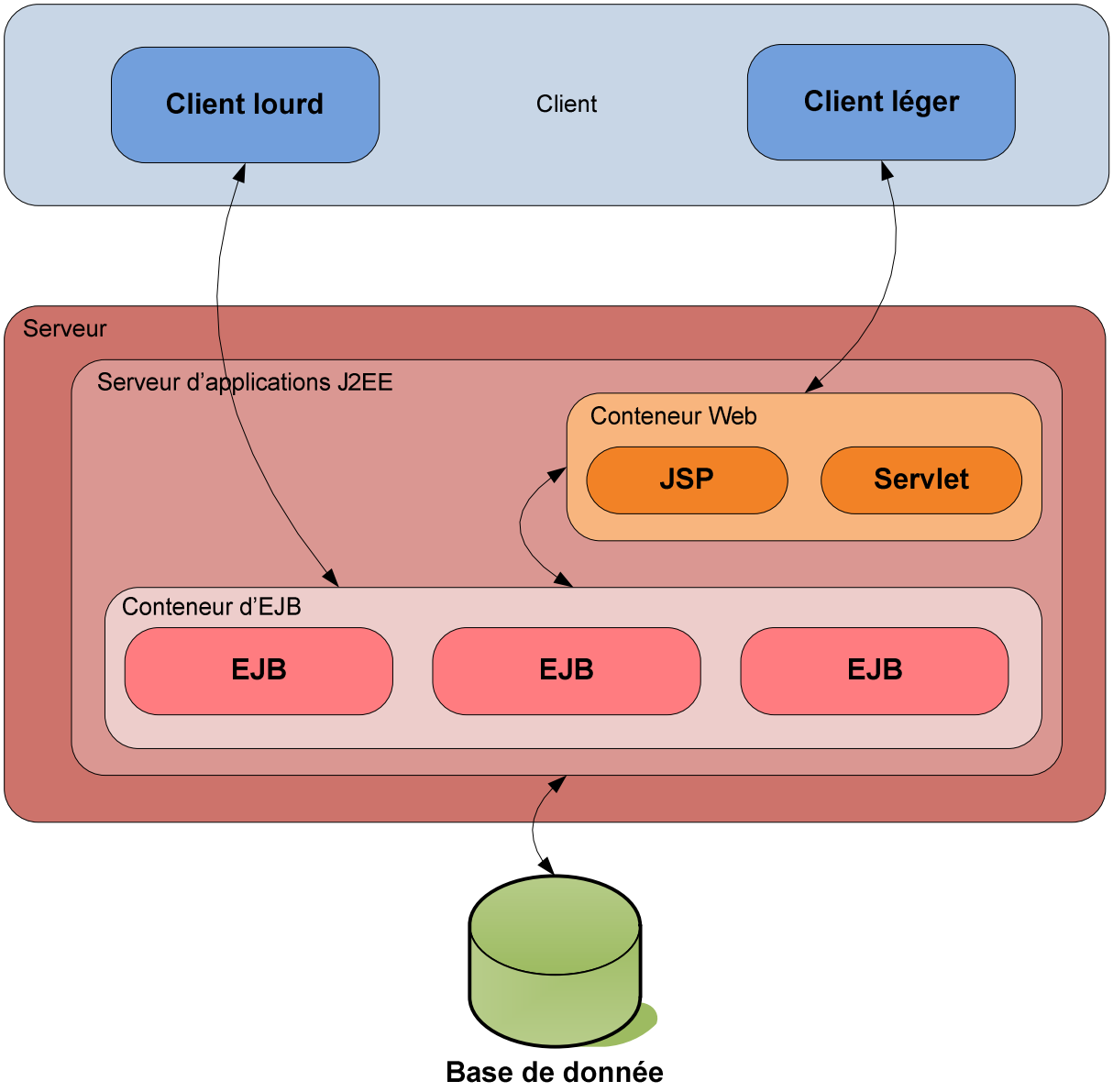


Figure 1 : Architecture J2EE classique

1.1.2 Présentation de l'architecture de HIP

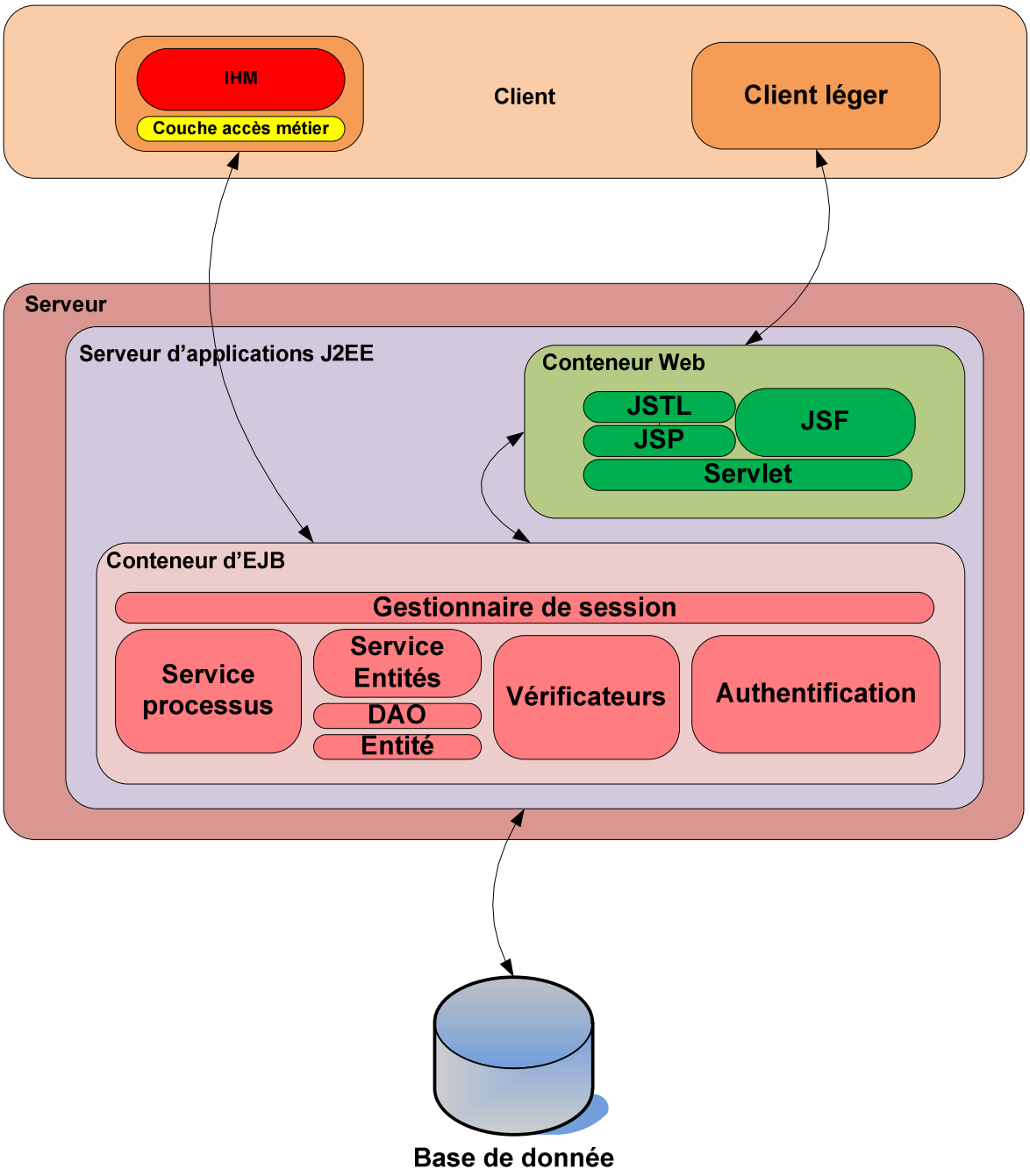


Figure 2 : Architecture Hip

Notre logiciel s'appuiera sur les principes énoncés dans la partie précédente et sera découpé en trois couches : la couche présentation, la couche métier et la couche d'accès aux données.

1.1.2.1 La couche présentation

HIP sera fourni avec deux types de clients : un client léger et un client lourd.

- *Le client léger*

Le client léger est une projection en HTML d'une partie du conteneur web. Contrairement au client lourd, le client léger s'exécute dans le conteneur Web du serveur J2EE, ce qui permet d'accéder à la couche métier directement via les APIs décrites dans la spécification J2EE.

- *Le client lourd*

Le client lourd possède sa propre interface graphique. Il s'exécute en dehors du serveur d'application. Il nécessite donc une couche qui permet l'accès à la couche métier du serveur d'application.

1.1.2.2 La couche métier

La couche métier est assurée par le conteneur EJB 3.0 du serveur d'application. Il permet de fournir des méthodes simples et uniformes quelques soient les méthodes d'accès utilisées (locales avec le client léger ou distantes avec le client lourd). Cette couche comprendra divers modules dont :

- Les gestionnaires de session : responsable de la gestion des utilisateurs.
- Les vérificateurs : Ils permettent de vérifier les données avant qu'elles soient traitées par les processus ou avant qu'elles soient sauvegardées en base.
- La couche authentification : elle permet d'authentifier les utilisateurs de l'application.
- La couche service des processus : elle contient l'ensemble des classes qui permettront le traitement de données telles que la génération de planning ou la génération de convocation par exemple
- La couche service pour les entités : responsable de fournir les actions en base de données à la couche gestionnaire de session.
- La couche DAO : responsable de la manipulation des entités de l'application.
- La couche entité : composée de plusieurs entités qui permettent la sauvegarde en base ou encore la récupération de données.

1.1.2.3 La couche d'accès aux données

La couche d'accès aux données est représentée par une base de données qui accompagne le serveur d'application.

2 Composants logiciel utilisés

Dans cette seconde partie, nous détaillerons l'ensemble des logiciels que nous utiliserons dans le cadre de notre projet.

2.1 La couche d'accès aux données

2.1.1 La base de données

Le choix du SGBD s'est effectué suivant différents critères :

- Le respect de la norme SQL 99 : cette norme permet d'assurer la compatibilité entre le système de gestion de base de données et un serveur d'application JEE 5.O dans le cadre du mapping objet- relationnel.
- La présence d'un driver JDBC : ce driver est nécessaire afin de pouvoir interfacer le SGBD au serveur d'application,
- La maintenance de la base par le service informatique de l'université : l'application doit être déployée au sein du système d'information de l'université, elle doit être administrable par son service informatique.
- Le support de Solaris.

Ensuite, nous avons testé trois SGBD :

- **MySQL :** C'est un SGBDR sous licence GPL qui est utilisé dans beaucoup de logiciels open source. Il respecte la norme SQL99 et il existe un driver JDBC MySQL. Il peut gérer les transactions et les clés étrangères grâce à la technologie InnoDB.
- **PostgreSQL :** C'est un SGBDRO qui offre plus de fonctionnalités que MySQL. Il supporte également le SQL99 et il existe un driver JDBC pour s'interfacer avec lui. Il permet de créer des bases de données plus complexes que MySQL, il gère les sous-sélections et les procédures stockées.
- **Oracle :** Oracle est un SGBD propriétaire fourni par Oracle Corporation. Il est couramment utilisé en entreprise.

SGBD	Norme SQL	Solution gratuite	Support de Solaris	Classement
MySQL	SQL99	Oui	Oui	2
Oracle	SQL99	Oui, il existe une version limitée d'oracle gratuite (Oracle Standard One)	Oui	3
PostgreSQL	SQL99	Oui	Oui	1

La solution de SGBD retenue est PostgreSQL car elle a l'avantage d'être gratuite, contrairement à Oracle, et est plus facile à mettre en place sur nos machines pour le développement. De plus, elle possède un certain nombre de fonctionnalités dont Mysql ne dispose pas.

2.2 La couche métier

2.2.1 Le serveur d'application

Un serveur d'application est un serveur sur lequel sont installées des applications accessibles aux utilisateurs via le réseau. Le serveur d'application retenu devra s'appuyer sur la spécification J2EE et fournir l'ensemble des fonctionnalités décrites dans cette spécification.

Le choix du serveur d'application utilisé pour héberger notre application s'est effectué suivant ces critères :

- La certification J2EE : J2EE comporte plusieurs versions. Pour pouvoir utiliser un certain nombre de fonctions introduites dans la version 1.5 de Java comme les annotations ou encore la générification, notre serveur doit être spécifié JEE 5.0.

- La présence de la norme EJB 3 : La spécification 3 des EJBs a tout d'abord été élaborée en vue de simplifier la conception d'EJB du côté développeur grâce à l'utilisation de classes Java, pour effectuer la persistance des données, ou encore l'utilisation d'annotations, afin de simplifier le code.

- Le support de Solaris : notre serveur doit être déployé dans un environnement Solaris.

Le marché des serveurs d'applications se découpe en deux grandes catégories, le marché des produits payants (IBM Websphere AS, Borland AS) et le marché des produits gratuits (JBoss, JOnAS…). Une des contraintes du projet est d'utiliser des logiciels gratuits, nous testerons donc uniquement cette catégorie de produit. Parmi les serveurs d'applications gratuits, on dénombre :

- **JOnAS :** JOnAS est un serveur d'application Java EE Open Source, développé par le consortium ObjectWeb. ObjectWeb est un consortium européen à but non-lucratif, fondé par INRIA, Bull et France Télécom. JOnAS est distribué sous la licence open-source LGPL. La dernière version est certifiée J2EE 1.4 et est en train de migrer vers la norme JEE 5.

- **JBoss :** JBoss Application Server est un serveur d'application J2EE libre entièrement écrit en Java, publié sous licence LGPL. Il peut être utilisé sur tout système d'exploitation fournissant une machine virtuelle Java. JBoss est certifié J2EE 1.4.

- **Glassfish :** Glassfish est le nom d'un serveur d'application créé par Sun Microsystems pour la plateforme J2EE. Sun vend cette application en tant que Sun Java System Application Server 9.X. Glassfish est doublement licencié sous les licences CDDL et GPL. Il est supporté sous environnement Solaris.

- **Apache Geronimo :** Apache Geronimo est le serveur d'application de l'Apache Software Foundation. Il respecte la spécification J2EE et est distribué sous licence ASF (Apache licence 2.0). Ce produit est supporté sous environnement Solaris et est certifié JEE 5.0.

Serveurs d'applications	Certification J2EE	EJB	Support de Solaris	Classement
JOnAS 4.8.6	J2EE 1.4	EJB 3.0 en utilisant EasyBeans	Oui	4
JBoss 4.2	J2EE 1.4	EJB 3.0	Oui	3
Glassfish v2	JEE 5.O	EJB 3.0	Oui	1
Apache Geronimo v2.0.1	JEE 5.O	EJB 3.0	Oui	2

La solution retenue est le serveur d'application Glassfish, car il possède la certification JEE 5.O, le support de Solaris, et on peut noter qu'il y a beaucoup plus de retours utilisateurs sur cette solution comparé au serveur Apache Geronimo.

2.2.2 L'authentification et l'interaction avec LDAP

LDAP ou Lightweight Directory Access Protocol est un protocole permettant la consultation et la modification de services d'annuaires. Le serveur LDAP d'une entreprise permet de gérer, par exemple, l'authentification à des services tels qu'un serveur mail, à un intranet ou à une machine physique. Il permet d'utiliser les mêmes informations de connexion (le même couple login/mot de passe). Il est donc nécessaire de trouver une API qui permettra de relier l'application au serveur LDAP de l'université dans le but que chaque personne puisse garder son login/mot de passe afin de s'authentifier. De plus, comme chaque personne voulant accéder à l'application ne disposera pas d'un compte LDAP à l'université, elle devra être capable de gérer des utilisateurs à l'aide de la base de données locale.

Nous avons testé trois bibliothèques de sécurité pour JEE :

- **JLDAP :** JLDAP a été développée par Novell. Elle permet d'écrire des applications qui peuvent accéder, gérer, modifier et chercher des informations stockées dans des répertoires accessibles en utilisant LDAP.

- **JAAS (Java Authentification & Autorisation Service) :** Cette API est fournie en standard avec le kit de développement de J2EE. Elle permet aux services d'authentifier les utilisateurs et d'appliquer les contrôles d'accès. Il est possible de l'interfacer avec un LDAP afin d'assurer l'authentification des utilisateurs.

- **JGuard :** jGuard est un framework de sécurité en Java basé sur JAAS. Ce framework est conçu dans le but de résoudre les problèmes de contrôle d'accès dans les applications Web. Il permet à un utilisateur d'avoir plusieurs rôles en même temps et peut s'interfacer avec un LDAP pour gérer l'authentification.

Nous avons choisi d'utiliser la bibliothèque JAAS. En effet, JGuard est trop lourd à mettre en place dans un environnement client lourd et optimisé pour la mise en place de sécurité dans un client léger. JAAS, quant à lui, a l'avantage d'être directement intégré à la norme JEE 5.0. De plus, JAAS et JLDAP peuvent être complémentaires dans la mise en place de l'authentification LDAP même si JLDAP n'est pas nécessaire.

2.2.3 Implémentation de JPA

JPA permet de manipuler des données en base de données directement à partir d'objets Java sans écrire du code SQL. Il existe plusieurs implémentations de JPA :

- **TopLink :** il permet le mapping objet/relationnel en Java et est développé par Oracle. Il permet également de stocker des objets Java et des EJB (Entreprise Java Been) dans des bases de données relationnelles mais aussi, en implémentant l'API Java JAXB (Java Architecture for XML Binding), de convertir des objets java en documents XML.

- **Hibernate :** Hibernate est un framework de mapping objet/relationnel (ORM) sous GPL gérant la persistance des objets en base de données.
  Ce Framework s'occupe du transfert des classes Java dans les tables de la base de données, de requêter les données et de proposer des moyens de récupérer les données. Hibernate est adaptable en termes d'architecture, il peut donc être utilisé aussi bien dans un développement client lourd, que dans un environnement web léger ou dans un environnement J2EE complet. Il gère l'héritage, le polymorphisme, les collections, etc.

Hibernate est plus puissant que Toplink. Il fournit, en plus de l'implémentation standard de JPA, des classes qui permettent d'ajouter des fonctionnalités supplémentaires pour la gestion de la couche de persistance telles que le Support des curseurs, le Filtrage dynamique des résultats ou le Cascade « delete-orphan ». Ces fonctionnalités ne sont pas nécessaires à l'intérieur de notre projet. TopLink a l'avantage d'être directement intégré au serveur d'application Glassfish. Nous utiliserons TopLink car il fournit l'ensemble des fonctionnalités dont nous avons besoin et il est plus facile à mettre en place que Hibernate.

2.2.4 L'algorithme de génération du planning

Notre logiciel a pour but de générer un planning en prenant en compte diverses contraintes, comme le fait qu'un tuteur enseignant ne puisse pas être présent à deux soutenances en même temps ou que les soutenances d'un tuteur enseignant ou d'un professeur de communication soient fixées dans la même journée. Une façon de résoudre ce problème est d'utiliser la méthode de programmation par contraintes.

La programmation par contraintes (PPC, ou *CP* pour *Constraint Programming* en anglais) consiste à programmer avec des relations appelées contraintes. Un problème comporte un certain nombre de variables, chacune ayant un domaine (l'ensemble des valeurs que peut prendre cette variable), et un certain nombre de contraintes. Une contrainte implique une ou plusieurs variables, et restreint les valeurs que peuvent prendre simultanément ces variables. Trouver une solution à un problème de PPC consiste à décrire l'ensemble des affectations autorisées de chaque variable, de telle sorte que la totalité des contraintes soient satisfaites.

Il existe différente API pour résoudre un problème avec la méthode PPC :

- **JChoco :** Cette API java permet la résolution de problèmes par la méthode de programmation par contraintes. Elle est distribuée sous licence BSD.

- **JaCoP Constraints :** Cette librairie Java permet également la résolution de problèmes par la méthode de programmation par contraintes.

- **JLC :** Java Library Constraints a été l'une des premières librairies à apporter la résolution d'un problème de satisfaction de contraintes. C'est un projet mature distribué sous licence LGPL.

- **Cream :** Cream est une bibliothèque Java qui permet de résoudre un problème par la programmation par contraintes. Cette librairie est distribuée sous licence LGPL.

Nous utiliserons JChoco car il est mieux documenté que tous les autres. On peut notamment trouver des tutoriaux sur son site officiel qui présente la résolution de problèmes tels qu'un sudoku, par exemple.

2.2.5 La lecture de fichier Microsoft Excel

Les coordonnées des apprentis et de leurs tuteurs enseignants sont stockées dans un fichier Microsoft Excel. Il est donc nécessaire de trouver une API qui peut fournir des fonctionnalités de lecture de ce type de fichier :

- **Apache Jakarta POI :** Le projet POI fournit une API servant à manipuler de multiples formats de fichiers Microsoft. Elle fournit des méthodes afin de lire ou d'écrire dans des fichiers Microsoft Excel ou Microsoft Word, des versions 97 à 2003.

- **Java Excel API :** Java Excel API est une API Java Open Source sous licence LGPL permettant de lire, d'écrire ou de modifier des fichiers Microsoft Excel. Cette API contient un mécanisme qui permet à des applications Java de lire ou de créer des feuilles de calcul, de modifier des cellules et de les éditer.

- **OpenOffice SDK :** Cette API permet de manipuler l'ensemble des formats que peut supporter la suite OpenOffice.org. Elle permet de manipuler notamment des fichiers tableurs Microsoft Excel comme le ferait OpenOffice Calc, le tableur de la suite OpenOffice.org.

Nous utiliserons la bibliothèque Java Excel API car celle-ci permet les opérations de bases que nous souhaitons effectuer, comme la lecture de valeurs dans une cellule. Elle est aussi plus simple à utiliser qu'Apache POI.

2.2.6 La génération des convocations à partir d'un modèle

Les convocations sont générées à partir d'un modèle, qui doit être éditable. Pour cela, il existe :

- **JasperReports :** JasperReports est une librairie Java open source permettant la génération de rapports. Ces rapports peuvent ensuite être exportés dans des fichiers PDF, HTML, etc. Il est entièrement écrit en Java et peut être utilisé dans des applications Java, telles que des applications Web ou J2EE pour générer du contenu dynamique.

- **JODReports :** JODReports, Java OpenDocument Reports, est une librairie Java qui permet de générer des documents dynamiques en fusionnant un modèle avec des données. Ses principales fonctionnalités sont qu'il est basé sur le format texte OpenDocument, que les modèles peuvent être entièrement créés de façon visuelle grâce à un logiciel de traitement de texte comme OpenOffice.org et les données peuvent provenir de multiples sources comme du XML, des POJOs ou des Maps. JODReports est un logiciel open source distribué sous les termes de la licence LGPL.

Nous utiliserons JODReports car celui-ci est simple à utiliser et s'appuie sur les formats de Microsoft ou d'OpenOffice pour rédiger le modèle.

2.2.7 La génération de PDF

Lorsque le planning est validé, les convocations peuvent être générées. Ces convocations doivent pouvoir être exportées au format PDF afin de les archiver. Pour cela, il est nécessaire de trouver une API permettant de manipuler le format PDF.

- **iText :** iText est une librairie qui permet la génération de fichiers PDF. Elle est développée sous licence LGPL.

- **PDFBox :** Cette librairie distribuée sous la licence BSD permet la création de fichiers PDF à partir de fichiers textes.

- **Gnujpdf :** Gnujpdf est une librairie java licenciée sous LGPL. Elle fournie une API simple afin de créer des fichiers PDF à partir d'objet Graphics.

- **JODConverter :** JODConverter est une librairie java gratuite qui automatise les conversions qui peuvent être réalisées par OpenOffice.org. Cela signifie que si on peut convertir un format ABC vers un format XYZ depuis OpenOffice.org alors on peut effectuer le même mécanisme avec JODConverter. Elle peut donc transformer un fichier Microsoft Word ou OpenOffice Writer en fichier PDF.

Nous utiliserons Gnujpdf car cette librairie nous permettra d'exporter le planning très facilement à l'aide des objets Graphics.

2.2.8 L'envoi de mail

Pour avertir les apprentis et leurs tuteurs enseignants sur l'organisation des soutenances, notre application doit gérer l'envoi de mails, contenant les convocations aux soutenances. L'envoi sera réalisé à l'aide de Java Mail API qui est une API directement intégrée à la spécification JEE 5.0. De plus, elle est capable de gérer l'envoi de mails pour différents protocoles (IMAP et POP3).

2.3 La couche présentation

2.3.1 Le client léger

Le client léger s'appuiera sur les technologies Servlet et JSP. Une servlet est une application Java qui permet de générer dynamiquement des données au sein d'un serveur HTTP. Ces données sont le plus généralement présentées au format HTML. Un fichier Java Server Pages est une page contenant majoritairement du code HTML, ou autre code Web, et du code Java. L'inclusion du code Java permet de générer des pages HTML dynamiques. Par exemple, les pages pourront avoir un contenu différent selon les informations stockées en base de données ou encore selon les préférences de l'utilisateur.

Avec ces deux technologies, la création d'interfaces graphiques ergonomiques et complexes est très difficile. Pour résoudre, ce problème il existe des frameworks Web qui permettent de simplifier certain processus comme la création de composants tels que des tableaux ou formulaires ou encore la transmission des données à la partie métier.

On peut citer :

- **Apache Struts :** Apache Struts est un framework open source utilisé pour faciliter le développement des applications web J2EE. Son but premier est de permettre la mise en place d'une architecture MVC (modèle-vue-contrôleur) plus aisément. Il utilise pour cela l'API des servlets en les étendant et en donnant accès à des objets qui améliorent l'approche de ces dernières. Cela débouche sur une meilleure subdivision et structuration du code d'une application web. Cette structuration permet ainsi une meilleure maintenabilité et modularité pour des développements futurs.

- **JSF :** JSF signifie "Java Server Faces" et est un framework pour les applications web respectant l'architecture J2EE. Le but premier de JSF est de procurer un environnement de développement permettant de construire une interface de type web sans devoir toucher aux codes HTML et JavaScript. Ceci est réalisé par la mise en place d'un mapping entre l'HTML et les objets concernés (Managed Bean).

- **Spring :** Spring est un framework J2EE. Il est l'un des plus appréciés des développeurs car il a une particularité importante par rapport à bien d'autres des frameworks existant tels que Hibernate ou encore Struts. Son implication ne se limite pas à un tiers de l'architecture trois-tiers mais touche l'ensemble de l'architecture. Le framework Spring est considéré comme un conteneur léger, c'est-à-dire qu'il permet de se baser sur une architecture avec des objets simples. Les objets peuvent dès lors être utilisés avec Spring sans que ces derniers ne doivent implémenter une interface ou hériter d'une classe. En fait des objets totalement indépendants du framework dans lequel ils sont exploités. Les relations entre les objets passent par des fichiers de configuration qui permettent, une fois instaurés, une grande liberté et flexibilité. Spring ne se présente pas comme un concurrent face aux autres frameworks existants, il se place comme un cadre permettant de faciliter l'utilisation justement des autres frameworks existants. Le premier exemple se situe au niveau de la couche de présentation dans une architecture MVC. Cette couche peut par exemple être implémentée via le framework Struts. Le deuxième exemple se situe au niveau de la couche de persistance toujours dans une architecture MVC où l'un des frameworks les plus répandus est Hibernate. Spring permet aussi d'intégrer aisément ce framework et lui apporte des aides comme la déclaration et gestion de transactions.

- **Apache Tapestry :** Apache Tapestry est un framework J2EE qui permet la création d'applications Web dynamiques à la manière d'Apache Struts ou JSF. Il se démarque de ses concurrents par une approche totalement différente. Pour Apache Tapestry, la création de page HTML se fait en assemblant des composants. Il se repose donc sur un concept simple et relativement proche du développement d'interface graphique en Java avec Swing.

Nous utiliserons le framework JSF et ses implémentations Apache Myfaces et Apache Tomahawk. En effet, JSF est très bien documenté, c'est une spécification qui fait partie de JEE 5.0 et dispose d'un certain nombre d'implémentations qui met à disposition de multiples composants déjà créés. Par exemple, Apache Tomahawk dispose d'un composant calendrier déjà implémenté. De plus, contrairement à Apache Struts et Spring, JSF se met en place assez facilement et demande moins de temps de formation car les concepts qui forment la base de JSF sont simples.

2.3.2 Le client lourd

Come dit précédemment, le client lourd possède une interface graphique propre. Plusieurs technologies s'offrent à nous pour réaliser celle-ci :

- **AWT :** AWT (Abstract Widget Toolkit) est une bibliothèque graphique pour Java. Cette bibliothèque a été introduite dès les premières versions de Java.

- **Swing :** Swing est une bibliothèque graphique pour Java. Elle est fournie avec le kit de développement par défaut Java. Swing offre la possibilité de créer des interfaces graphiques identiques quelque soit le système d'exploitation sous-jacent. De plus, elle dispose de plusieurs choix d'apparence pour chacun des composants standards. Le pattern MVC est directement intégré au composant Swing. Pour finir, Swing est l'API graphique que nous avons l'habitude d'utiliser pour nos projets à Ingénieurs 2OOO et peut être enrichie avec des APIs telles que SwingX ou JGoodies afin de fournir des composants plus complexes ou de simplifier la validation de données dans des champs de saisie, par exemple.

Figure 3 : exemple SwingX

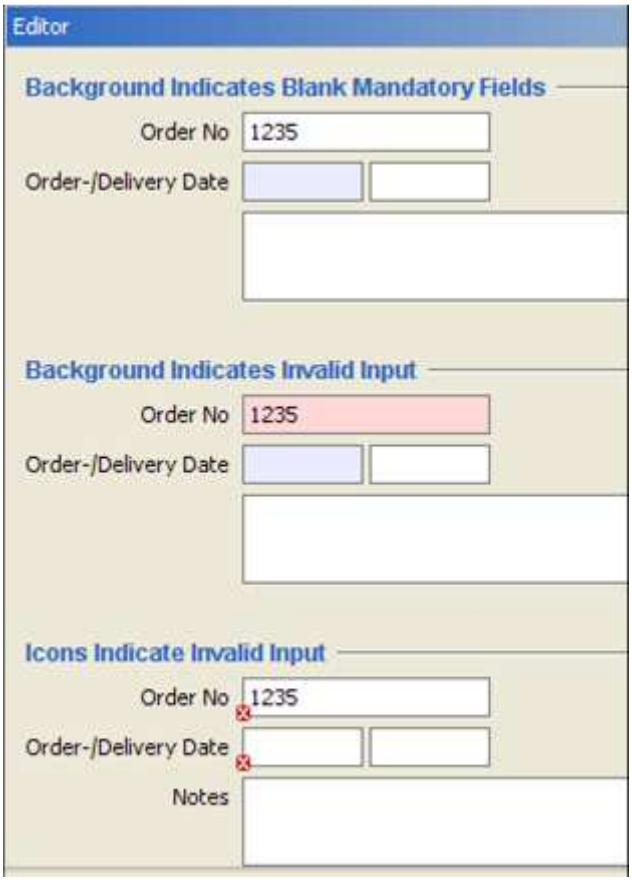


Figure 4 : Exemple d'utilisation de SwingX (calendrier, auto-complétion et validation de champ de saisie)

- **SWT :** (pour *Standard Widget Toolkit*) a été conçu par les développeurs d'Eclipse comme une alternative à AWT et Swing, considérés comme trop lents pour une application Java qui requiert des performances optimales. En effet, SWT repose sur les fonctions et objets systèmes de la plateforme sur laquelle l'application s'exécute.

- **JFace :** SWT est une API de bas niveau. Elle propose des objets qui permettent la création d'interfaces graphiques mais qui nécessitent aussi énormément de code. JFace propose d'encapsuler de nombreuses opérations de base et de faciliter ainsi le développement des interfaces graphiques reposant sur SWT. JFace est une bibliothèque qui utilise SWT afin de faciliter son utilisation dans le développement d'application. Il encapsule un certain nombre de traitements afin de faciliter l'utilisation de SWT notamment en réduisant la quantité de code à produire.

Nous avons choisi d'utiliser la technologie Swing associée à SwingX. En effet, chaque personne de notre groupe sait développer une interface graphique en Swing. De plus, SwingX et JGoodies nous permettront de réaliser facilement la validation des données et l'auto-complétion dans les champs de saisie, par exemple.

3 L'Analyse Conceptuelle

L'analyse conceptuelle consiste à analyser le domaine d'application pour en extraire les règles de comportement auxquelles sont soumis les objets du domaine. Il faut pour cela modéliser ces objets, définir leurs comportements et modéliser les mécanismes de traitement associés.

3.1 Généralités

Nous allons présenter par la suite une description de l'architecture logicielle grâce à une description de chaque paquetage, accompagnée d'un diagramme de paquetage, ou diagramme de classe.

Ceci n'est cependant pas une vue final de l'application, celle-ci sera probablement modifiée et affinée par la suite, durant le développement des versions futures.

3.2 Partie serveur HIP

3.2.1 Paquetages Java

Le diagramme ci-dessous représente l'ensemble des paquetages de la partie serveur de l'application HIP.

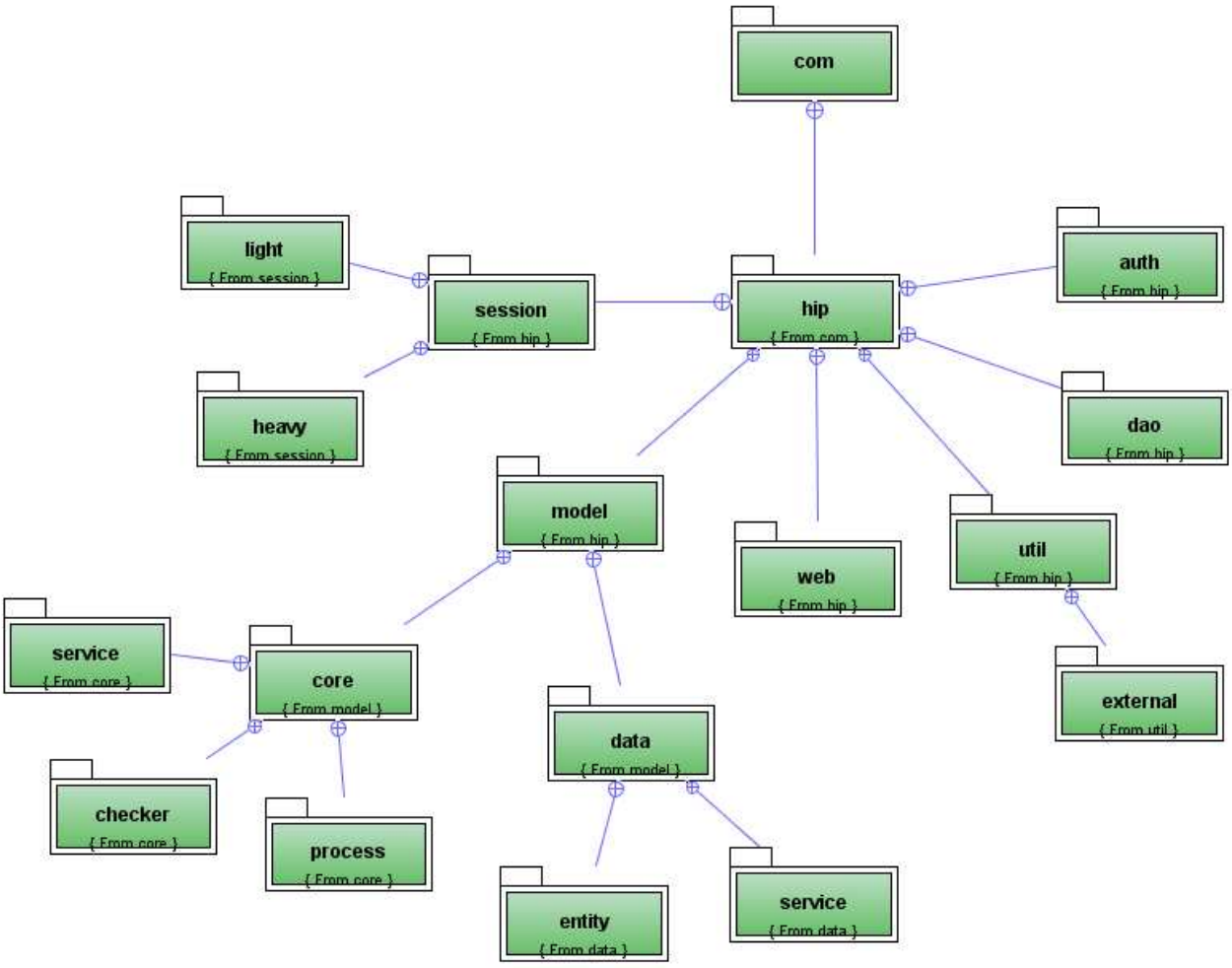


Figure 5 : Diagramme des paquetages du serveur HIP

3.2.2 Paquetage « model »

Le paquetage « model » correspond à la couche métier du serveur qui constitue la partie fonctionnelle de l'application. Elle implémente la logique et décrit les opérations que l'application effectue sur les données en fonction des requêtes des utilisateurs.

3.2.2.1 Paquetage « core »

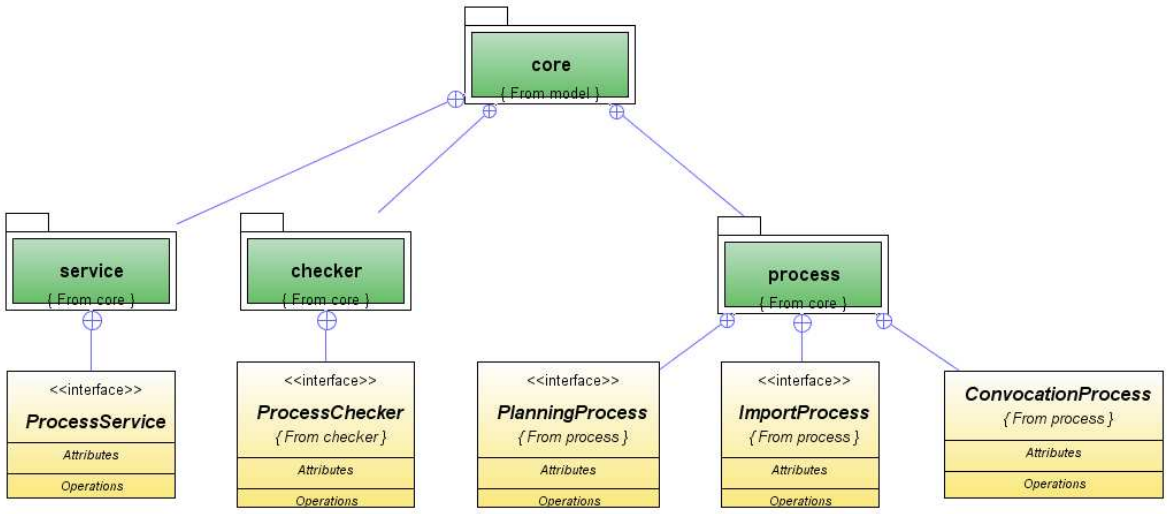


Figure 6 : Diagramme de classes du paquetage "core"

✓ **Paquetage « core.process »**

Le paquetage « process » correspond aux traitements métiers lourds qu'il faut effectuer sur le serveur. On retrouve, entre autres, la génération des plannings, des convocations et de l'import des données à partir du fichier Microsoft Excel.

✓ **Paquetage « core.checker »**

Le paquetage « core.checker » gère la vérification des informations reçues de la couche service par le client avant de traiter les données et d'effectuer des traitements du paquetage « process ».

✓ **Paquetage « core.service »**

Le paquetage « core.service » contient les méthodes d'accès aux traitements métiers permettant ainsi de cacher l'implémentation de ceux-ci.

3.2.2.2 Paquetage « data »

✓ **Paquetage « data.entity »**

Le paquetage « entity » représente les objets métiers dans un mécanisme de stockage persistant. Les entités sont des POJOs ordinaires qui possèdent un certain nombre d'annotations permettant de définir une représentation objet de la base de données. Les principales méthodes sont les getters et setters qui permettent d'accéder aux champs.

Pour une plus grande visibilité, nous allons découper les diagrammes de classes du paquetage « entity » en plusieurs parties.

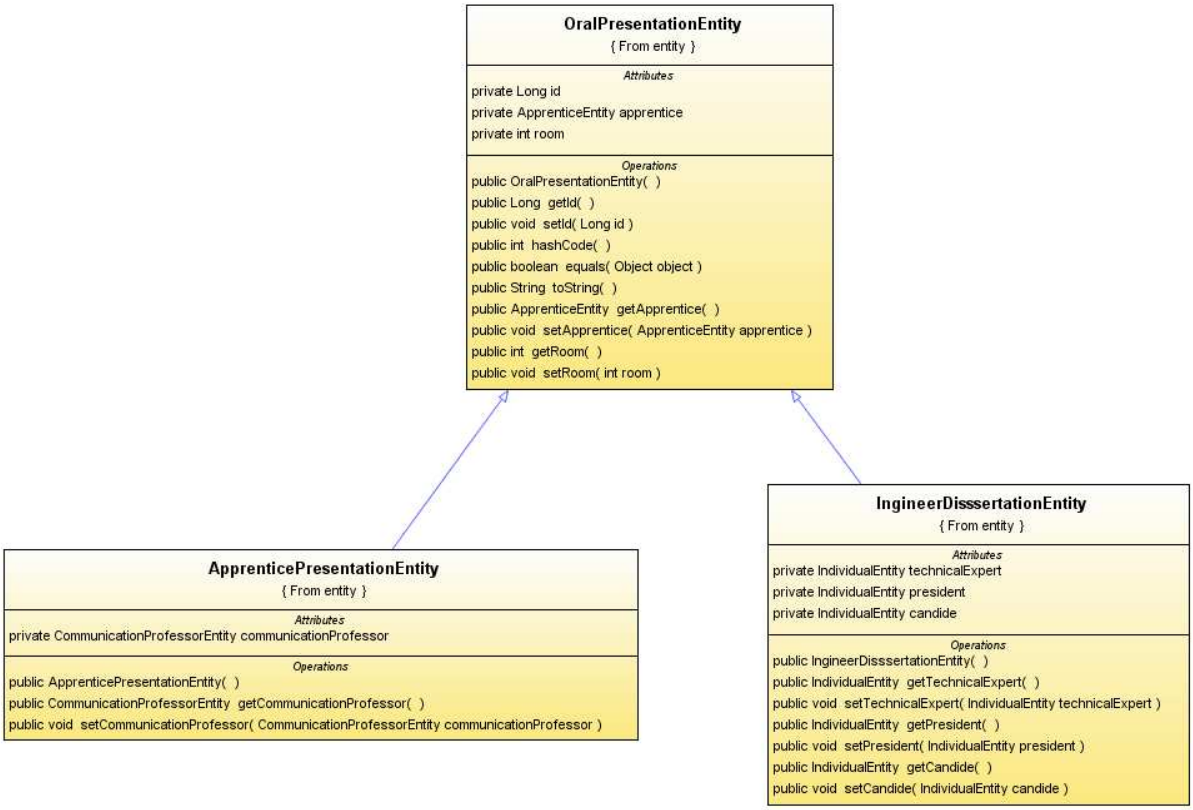


Figure 7 : Diagramme de classes du paquetage "data"

- **OralPresentationEntity :**
  Classe abstraite modélisant une soutenance générale et contenant des informations (apprenti, tuteur enseignant, …).

- **AppreticePresentationEntity :**
  Classe modélisant les soutenances des dossiers d'alternances de première et deuxième année qui étend de la classe OralPresentationEntity et contenant des informations supplémentaires (professeur de communication).

- **InigeerDissertationEntity :**
  Classe modélisant les mémoires d'ingénieur de troisième année qui étend la classe OralpresentationEntity et contenant des informations supplémentaires (Expert technique, président de jury, …).

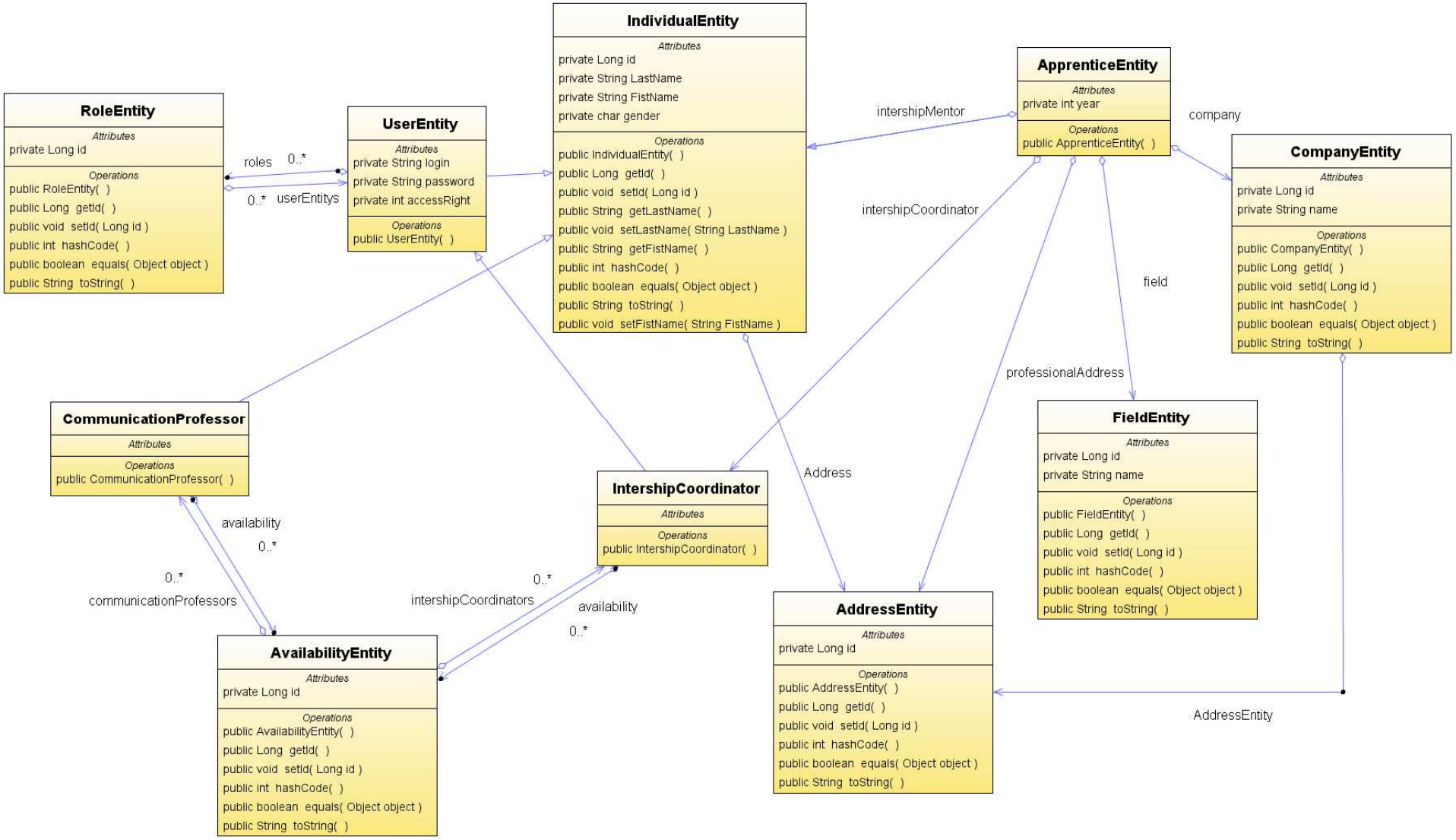


Figure 8 : Diagramme de classes du paquetage « entity »

- **IndividualEntity :**
  Classe modélisant une personne physique qui assiste à une soutenance avec diverses informations (nom, prénom, sexe, coordonnées).

- **ApprenticeEntity :**
  Classe modélisant un apprenti, étendant la classe IndividualEntity et contenant des informations supplémentaires (entreprise, tuteur ingénieur, …).

- **CommunicationProfessorEntity** :
  Classe modélisant un professeur de communication, étendant la classe IndividualEntity et contenant comme informations supplémentaires ses disponibilités.

- **UserEntity :**
  Classe modélisant un compte permettant de se connecter au logiciel, étendant la classe IndividualEntity et contenant des informations supplémentaires (login, mot de passe, …).

- **IntershipCoordinator :**
  Classe modélisant un tuteur enseignant, étendant la classe UserEntity et contenant comme informations supplémentaires ses disponibilités.

- **AddressEntity :**
  Classe modélisant les coordonnées d'une personne et contenant des informations (adresse, ville, téléphone, …).

- **CompanyEntity :**
  Classe modélisant l'entreprise d'un apprenti ou d'un tuteur entreprise et contenant des informations (nom, adresse).

- **FieldEntity** :
  Classe modélisant la filière d'un apprenti et contenant des informations (nom).

✓ **Paquetage « data.service »**

Le paquetage « data.service » contient les méthodes d'accès aux données persistantes à partir du paquetage « dao ». Il joue le rôle de « Façade ».

✓ **Paquetage « data.checker »**

Le paquetage « data.checker » contient toutes les classes responsables de la vérification et validation des entités reçues à partir du client. Il est primordial de faire ceci afin de ne pas corrompre la base de données.

3.2.3 Paquetage « session »

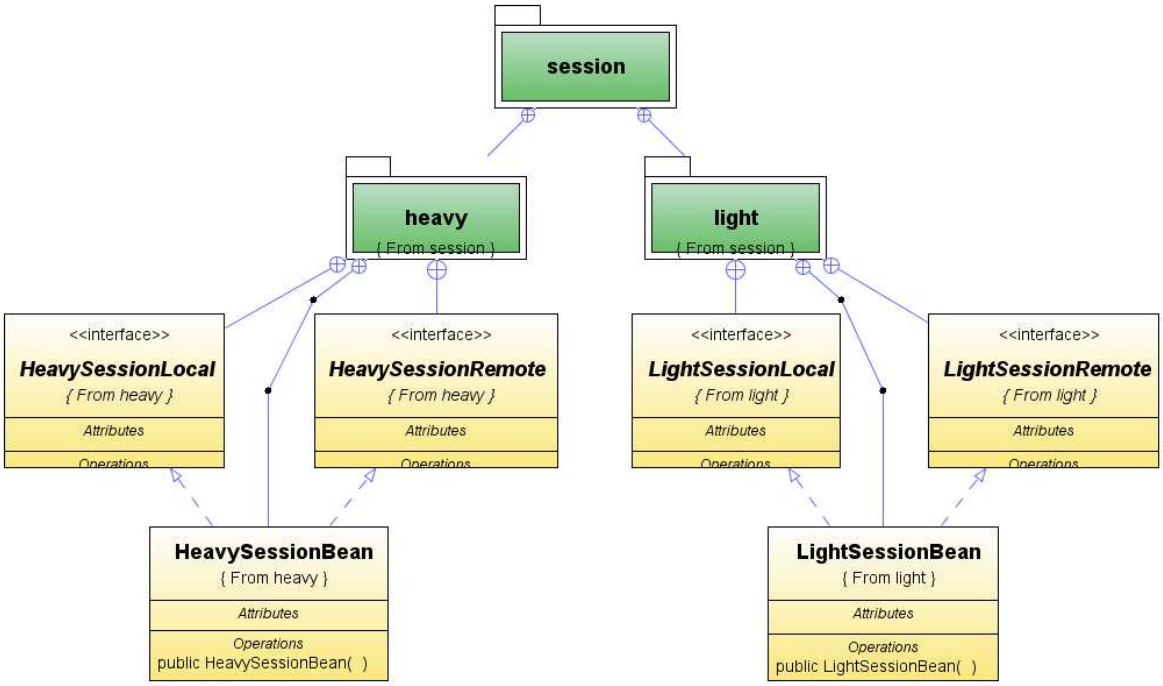


Figure 9 : Diagramme de classes du paquetage « session »

Ce paquetage « session » possède deux autres paquetages, « heavy » et « light ».

✓ **Paquetage « heavy »**

Le paquetage « heavy » fournie les opérations principales accessibles depuis le client lourd. Il s'agit d'un EJB session Stateful. Il assure la communication entre le client lourd et le serveur.

✓ **Paquetage « light »**

Le paquetage « light » fournit les opérations principales accessibles depuis le client léger. Il s'agit d'un EJB session Stateless. Il assure la communication entre le client léger et le serveur.

3.2.4 Paquetage « util »

Le paquetage « util » contient les outils et services nécessaires au fonctionnement de l'application Hip.

✓ **Paquetage « util.external »**

Le paquetage « util.external » fournit les services permettant d'utiliser certaines API existantes et ajoutées dans notre application.

On y trouve entre autres les services permettant d'utiliser l'API JExcel pour l'import du fichier Microsoft Excel ou encore l'API gnujPDF pour la sauvegarde des plannings au format PDF.

3.2.5 Paquetage « web »

Le paquetage « web » correspond à l'implémentation du client léger grâce à JSP qui permet de générer dynamiquement les pages Web.

3.2.6 Paquetage « dao »

Le paquetage « dao » représente la couche d'accès aux données persistantes via l'intermédiaire d'Entity Bean. Il joue le rôle de « Proxy » entre la couche service et la base de données persistante (ou plus précisément avec JPA).

L'interface ***DAO*** contient l'ensemble des méthodes génériques qui vont permettre d'effectuer les opérations nécessaires sur la base de données afin de créer, rechercher, modifier ou supprimer les objets métiers.

Une classe abstraite ***AbstractToplinkDAO*** implémente toutes les méthodes de l'interface DAO.

A chaque objet métier, on associe un **XXXDAO** qui contient l'implémentation des méthodes propres aux fonctionnalités de l'objet.

Les DAO sont instanciés à l'aide d'une classe ***DAOFactory*** qui a pour but de retourner une instance d'une classe en fonction d'un type demandé.

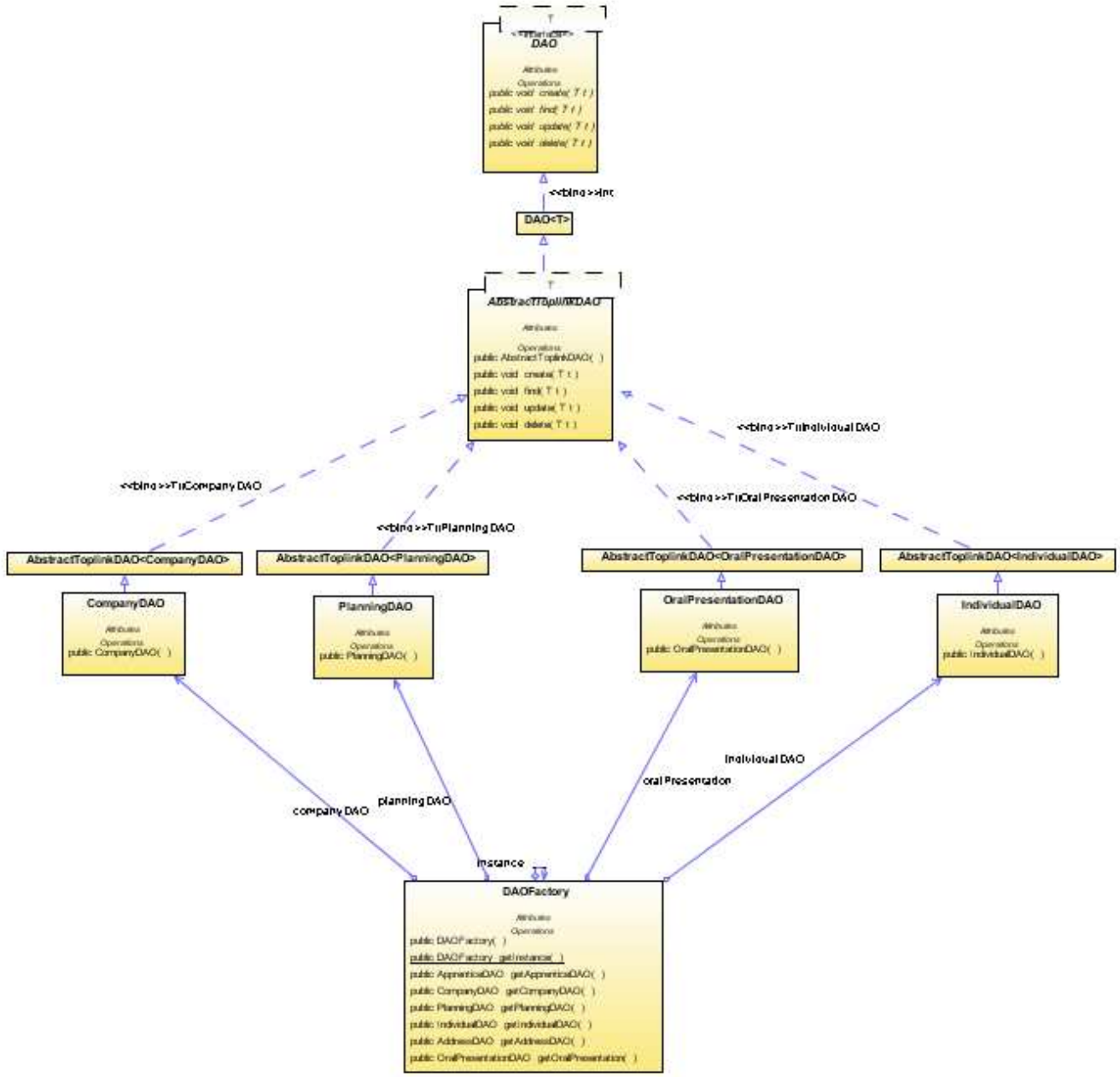


Figure 10 : Diagramme de classes du paquetage « dao »

3.2.7 Paquetage « auth »

Ce paquetage contient toutes les classes nécessaires à l'authentification d'un utilisateur sur Hip. Il repose sur les principes de JAAS.

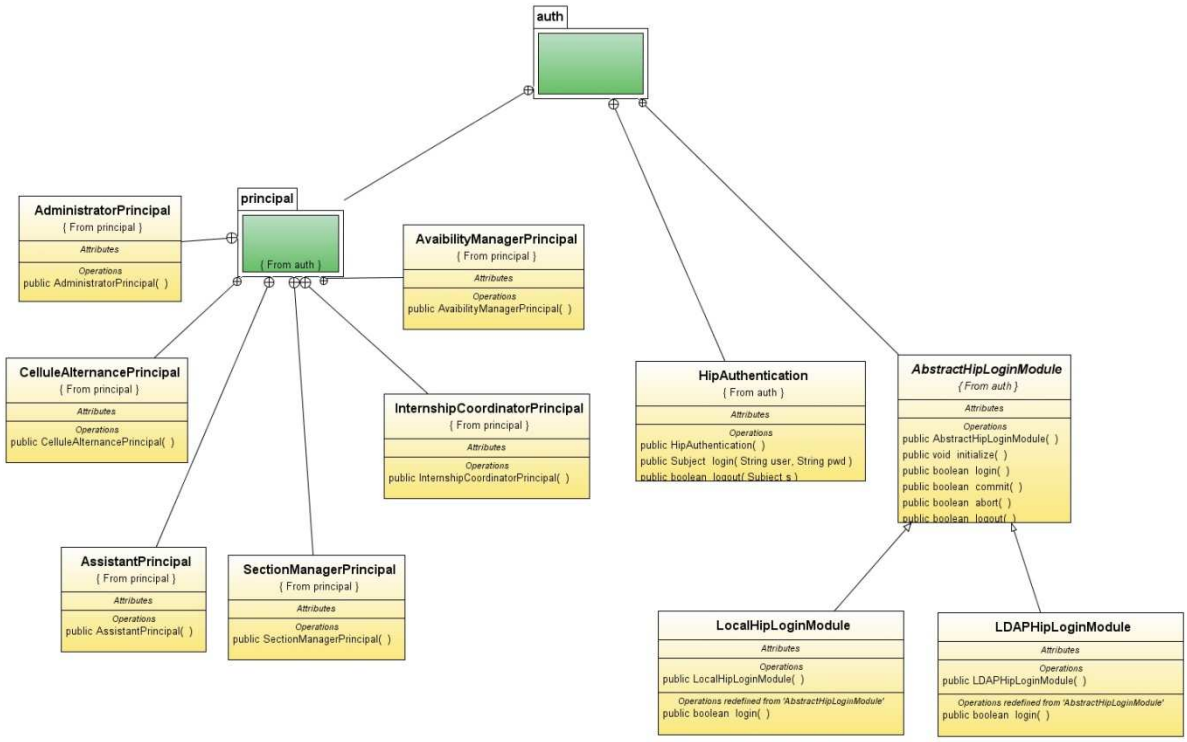


Figure 11 : Diagramme de classe du paquetage auth

HipAuthentication : Cette classe permet à un utilisateur de se loger ou de se déloger. Elle possède une méthode login() qui, dans un premier temps, essaied'authentifier l'utilisateur à travers les deux LoginModule de l'application (LocalHipLoginModule et LDAPHipLoginModule). Elle renvoie ensuite un Subject représentant l'utilisateur avec ces rôles respectifs. Cette classe contient une deuxième méthode qui se nomme logout() qui permet la déconnexion de l'utilisateur.

AbstractHipLoginModule : Cette classe est une classe abstraite qui implémente l'interface LoginModule de JAAS. LoginModule est une interface qui décrit comment un utilisateur s'authentifie. AbstractHipLoginModule implémente les méthodes initialize(), commit(), abort() et logout() de LoginModule. La méthode initialize() permet de faire du traitement lorsque le LoginModule est créé. commit() permet de récupérer les rôles des utilisateurs qui seront stockés en base de données une fois que ceux-ci sont authentifiés par la méthode login(). abort() est appelée lorsque l'authentification avec login() échoue et logout() permet de se déconnecter.

LocalHipLoginModule : Cette classe étend AbstractHipLoginModule et surcharge la méthode login() afin d'authentifier un utilisateur à l'aide de la base de données locale.

LDAPHipLoginModule : Cette classe étend AbstractHipLoginModule et surcharge la méthode login() afin d'authentifier un utilisateur à l'aide des annuaires de l'université.

Ce paquetage contient un paquetage nommé « principal ». Dans ce paquetage, on retrouve les classes qui représentent les rôles que chaque utilisateur peut avoir :

- ✓ **CelluleAlternancePrincipa**l pour le rôle cellule alternance,
- ✓ **AssistantPrincipal** pour le rôle secrétaire,
- ✓ **AdministratorPrincipal** pour le rôle Administrateur,
- ✓ **InternshipCoordinator** pour le rôle tuteur enseignant,
- ✓ **AvaibalityManagerPrincipal** pour le rôle responsable des disponibilités des TE,
- ✓ **SectionManager** pour le rôle responsable de filière.

Chacune de ces classes implémente l'interface Principal fournie par la bibliothèque JAAS.

3.3 Client lourd

Le client lourd représente l'interface graphique servant à afficher et modifier les informations fournies par le serveur.

Le diagramme ci-dessous représente les paquetages de la partie serveur de l'application HIP.

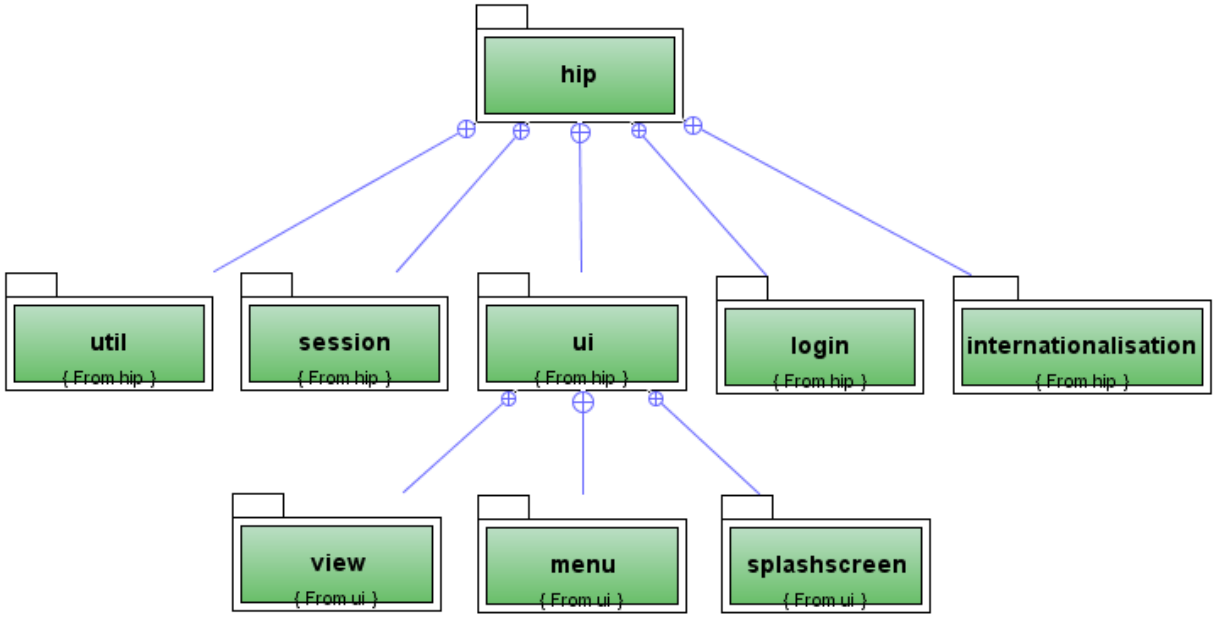


Figure 12 : Diagramme des paquetages du client lourd HIP

3.3.1 Paquetage « session »

Le paquetage « session » contient les classes permettant le dialogue entre le client lourd et le serveur.

3.3.2 Paquetage « ui »

Le paquetage « ui » contient toutes les classes relatives à l'interface graphique du client lourd. Il est subdivisé en plusieurs paquetages représentant différentes parties.

3.3.2.1 Paquetage « view »

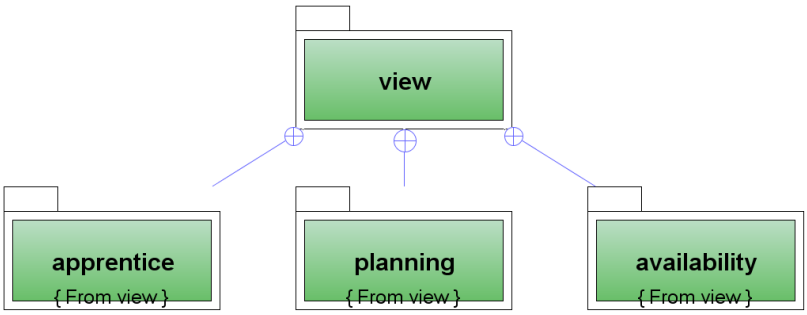


Figure 13 : Diagramme des paquetages du paquetage « view »

Le paquetage « view » contient les différentes vues disponibles pour l'utilisateur. Il est divisé en trois sous paquetages :

- ✓ Le paquetage « view.apprentice »

  Ce paquetage contient toutes les vues relatives à la gestion des apprentis.

- ✓ Le paquetage « view.planning »

  Ce paquetage contient toutes les vues relatives à la gestion des plannings ainsi que la génération des convocations.

- ✓ Le paquetage « view.availability »

  Ce paquetage contient toutes les vues relatives à la gestion des disponibilités des tuteurs.

3.3.2.2 Paquetage « menu »

Le paquetage « menu » contient les classes consacrées aux menus et à la page d'accueil du client lourd HIP.

3.3.2.3 Paquetage « SplasScreen »

Le paquetage « SplashScreen » contient les classes consacrées au fonctionnement de l'écran de chargement du client lourd HIP.

3.3.3 Paquetage « login »

Le paquetage « login » contient les classes gérant le système de logement de l'application.

3.3.4 Paquetage « internationalisation »

Le paquetage « internationalisation » contient les classes utilisées pour traduire les textes de l'application dans la langue souhaitée.

3.3.5 Paquetage « util »

Le paquetage « util » contient les classes outils utilisées par le client lourd.

4 Les Use Case génériques

La prochaine partie présente les Use Case génériques avec leurs diagrammes de séquences, pour l'ensemble des fonctionnalités.

4.1 Généralité

Plusieurs cas d'utilisation génèrent des diagrammes de séquences de même forme. La suite des opérations effectuées dans l'application sont les mêmes, seules certaines classes ou objets changeront.

Les cas d'utilisations dit « génériques » sont la création, la suppression, la modification et la visualisation des informations, mais aussi l'utilisation de l'application et l'accès aux vues du système.

Les Use Cases et diagrammes qui sont présentés comprennent des sections notées XXX. C'est à ce niveau des diagrammes que les différences existent.

4.2 USE CASE générique n°1 : Accéder à la <page>

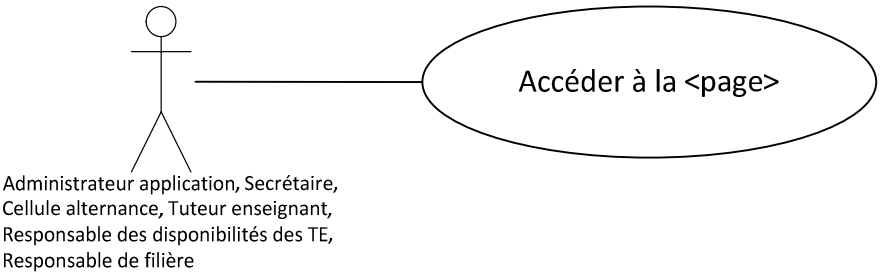


Figure 14 : Diagramme du cas d'utilisation Accéder à la <page>

✓ Acteurs

- Secrétaire,
- Cellule alternance,
- Administrateur application,
- Tuteur enseignant,
- Responsable des disponibilités des TE,
- Responsable de filière.

✓ Description

Cette action permet à l'utilisateur d'accéder à la <page> de gestion d'objets du système, après s'être authentifié sur la page d'authentification.

✓ Priorité

Priorité finale
2

✓ Pré-conditions

- L'application doit être lancée,
- L'utilisateur s'est authentifié sur le système,
- L'utilisateur se situe sur la page d'accueil.

✓ Tableau conversationnel

Actions menés par l'acteur	Actions réalisés par le système
1. Demande l'accès à la <page>.	
	2. Charge et affiche la <page>.

✓ Post-conditions

- L'utilisateur se situe sur la <page>.

✓ Exceptions

Erreur levée	Actions à réaliser par le système
L'accès à la <page> est indisponible.	Un message l'indique à l'utilisateur.

✓ Besoins IHM

- Un élément permettant l'accès à la <page>.

✓ FQM

Fonction	Fréquence	Qualité	Mesure
Accéder à la <page>	4 : élevée	Rapidité	Affichage des <pages> instantané.
		Sécurité	L'accès à la <page> doit être sécurisé et autorisé pour acteurs concernés.

4.3 USE CASE générique n°2 : Utiliser l'application du <client>

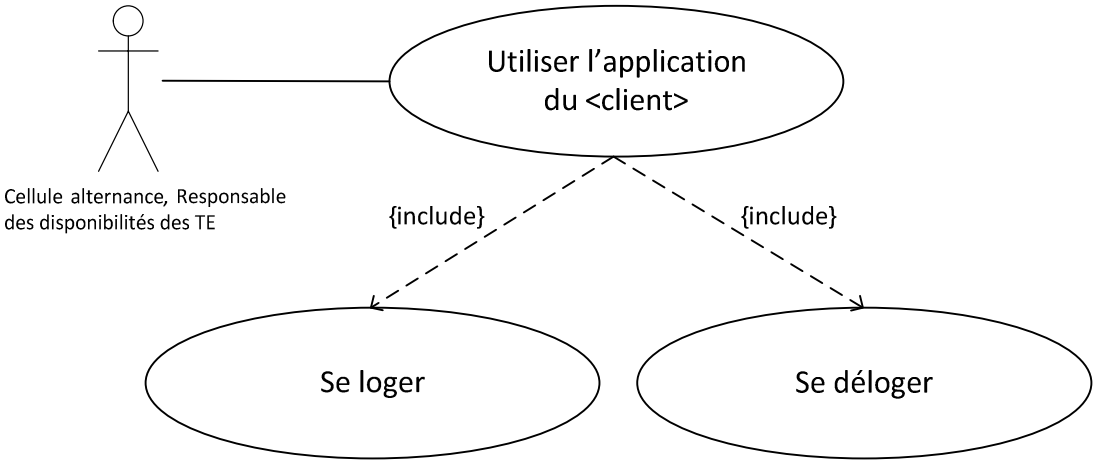


Figure 15 : Diagramme du cas d'utilisation Utiliser l'application d'un <client>

✓ Acteurs

- Administrateur application,
- Secrétaire,
- Tuteur enseignant,
- Responsable de filière.

✓ Description

Cette action permet de démarrer, utiliser puis quitter l'application du <client> du logiciel HIP.

✓ Priorité

Priorité finale
4

✓ Pré-conditions

- L'application du <client> doit être accessible pour l'utilisateur,
- L'utilisateur doit posséder les droits d'utilisation du <client>.

✓ Tableau conversationnel

Actions menés par l'acteur	Actions réalisés par le système
2. Démarre l'application du <client>.	
	3. Invite l'acteur à se loger via l'écran d'authentification.
4. Se loge (Cf. UC n°1.1 : *Se loger*).	
5. Utilise l'application.	
6. Demande à se déloger (Cf. UC n°1.1 : *Se déloger*).	
	7. Ferme la session de l'utilisateur et charge la page d'authentification.
8. Peut fermer l'application du <client>.	

✓ Post-conditions

- La session est fermée,
- L'application du <client> est fermée.

✓ Exceptions

Erreur levée	Actions à réaliser par le système
L'accès au serveur est indisponible.	Un message l'indique à l'acteur.

✓ FQM

Fonction	Fréquence	Qualité	Mesure
Utiliser l'application du <client>	4 : élevée	Rapidité	Affichage des pages instantané.
		Sécurité	L'accès au serveur doit être sécurisé et autorisé pour les acteurs concernés.

4.4 USE CASE générique n°3: Ajouter un XXX

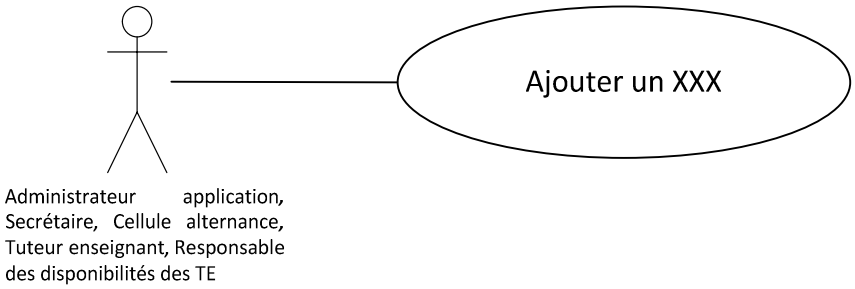


Figure 16 : Diagramme du cas d'utilisation Ajouter un XXX

✓ Acteurs

- Administrateur Application,
- Secrétaire,
- Cellule alternance,
- Tuteur enseignant,
- Responsable des disponibilités d'un TE.

✓ Description

Cette action permet aux acteurs concernés d'ajouter un XXX du système.

✓ Priorité

Priorité
3

✓ Pré-conditions

- L'application est lancée,
- L'utilisateur est authentifié,
- L'utilisateur se situe sur la page de gestion de XXX concerné.

✓ Tableau conversationnel

Actions menés par l'acteur	Actions réalisés par le système
1. Demande la création de XXX.	
	2. Affiche le formulaire de saisie des données.
3. Saisit les données relatives à XXX.	
4. Valide la saisie.	
	5. Accède à la base de données et y ajoute XXX avec ses données relatives.
	6. Informe l'utilisateur que XXX est ajouté.
	7. Affiche la page de gestion de XXX concerné.

✓ Post-conditions

- XXX est ajouté,
- La page de gestion de XXX concerné est affichée.

✓ Flots alternatifs

a. L'utilisateur annule la suppression lors de l'étape 4 de la conversation

Actions menées par l'acteur	Actions réalisées par le système
3. Saisit les données relatives à XXX.	
4.a. Annule la saisie.	
	5.a Affiche la page de gestion de XXX concerné.

✓ Exceptions

Erreur levée	Actions à réaliser par le système
L'accès à la base de données lors de la suppression de XXX a échoué.	La suppression n'est pas effectuée, l'utilisateur reste dans le menu de gestion de XXX concerné.

✓ Besoins IHM

- Un élément pour demander l'ajout d'un XXX,
- Un formulaire composé de composants de saisie (champs de texte, listes de choix, etc.) pour saisir les informations liées à l'objet ajouté,
- Une zone de confirmation contenant un élément de validation et un élément d'annulation,
- Une barre des tâches pour informer de la suppression de XXX.

✓ FQM

Fonction	Fréquence	Qualité	Mesure
Ajouter un XXX	3 : régulière	Rapidité	Instantanée.
		Sécurité	Gérer les accès concurrents aux données de la base.

Diagramme de séquence

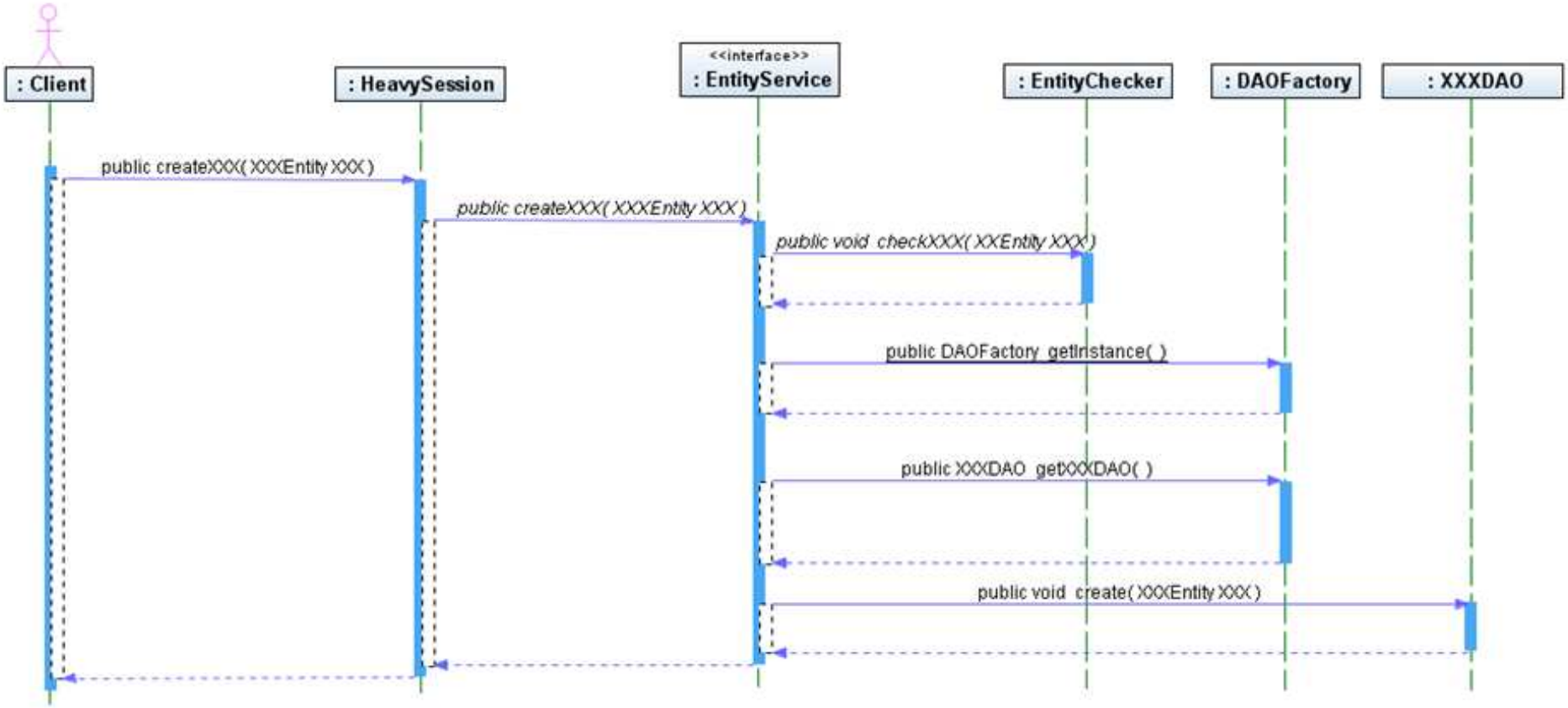


Figure 17 : Diagramme de séquence Ajouter un XXX

4.5 USE CASE générique n°4: Modifier un XXX

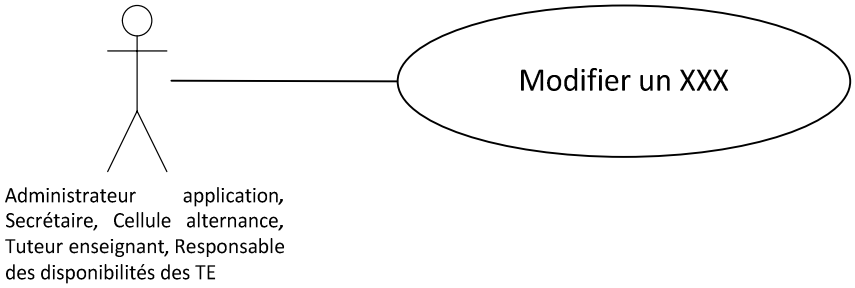


Figure 18 : Diagramme du cas d'utilisation Modifier un XXX

✓ Acteurs

- Administrateur Application,
- Secrétaire,
- Cellule alternance,
- Tuteur enseignant,
- Responsable des disponibilités d'un TE.

✓ Description

Cette action permet aux acteurs concernés de modifier un XXX du système.

✓ Priorité

Priorité
3

✓ Pré-conditions

- L'application est lancée,
- L'utilisateur est authentifié,
- L'utilisateur se situe sur la page de gestion de XXX concerné,
- XXX concerné existe.

✓ Tableau conversationnel

Actions menés par l'acteur	Actions réalisés par le système
1. Demande la modification de XXX.	
	2. Affiche le formulaire de saisie des données.
3. Modifie les données relatives à XXX.	
4. Valide les modifications.	
	5. Accède à la base de données et y modifie XXX avec les nouvelles données relatives.
	6. Informe l'utilisateur que XXX est modifié.
	7. Affiche la page de gestion de XXX concerné.

✓ Post-conditions

- XXX est ajouté,
- La page de gestion de XXX concerné est affichée.

✓ Flots alternatifs

b. L'utilisateur annule la modification lors de l'étape 4 de la conversation

Actions menées par l'acteur	Actions réalisées par le système
3. Modifie les données relatives à XXX.	
4.a. Annule les modifications.	
	5.a Affiche la page de gestion de XXX concerné.

✓ Exceptions

Erreur levée	Actions à réaliser par le système
L'accès à la base de données lors de la suppression de XXX a échoué.	La suppression n'est pas effectuée, l'utilisateur reste dans le menu de gestion de XXX concerné.

✓ Besoins IHM

- Un élément pour demander l'ajout d'un XXX,
- Un formulaire composé de composants de saisie (champs de texte, listes de choix, etc.) pour modifier les informations liées à l'objet ajouté,
- Une zone de confirmation contenant un élément de validation et un élément d'annulation,
- Une barre des tâches pour informer de la suppression de XXX.

✓ FQM

Fonction	Fréquence	Qualité	Mesure
Modifier un XXX	3 : régulière	Rapidité	Instantanée.
		Sécurité	Gérer les accès concurrents aux données de la base.

Diagramme de séquence

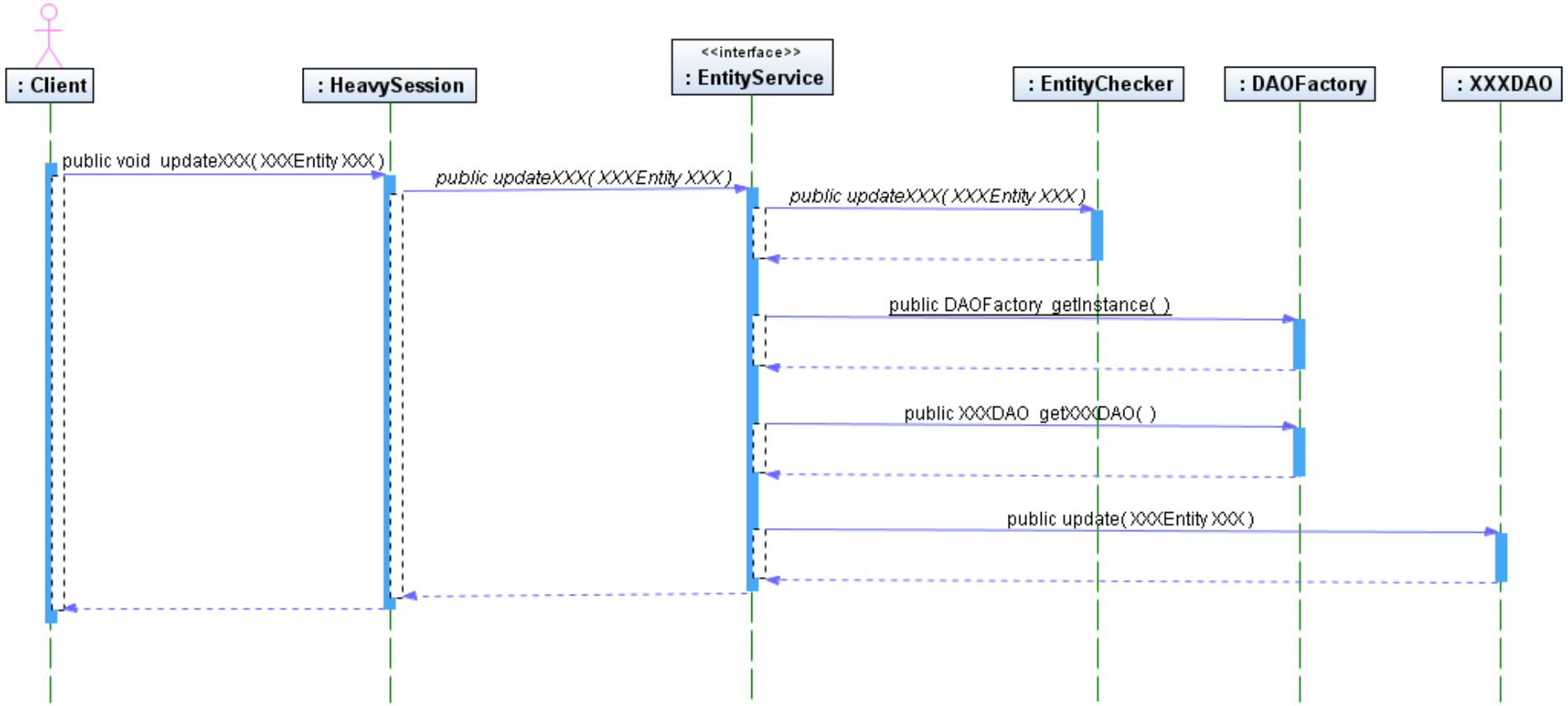


Figure 19 : Diagramme de séquence Modifier un XXX

4.6 USE CASE générique n°5: Supprimer un XXX

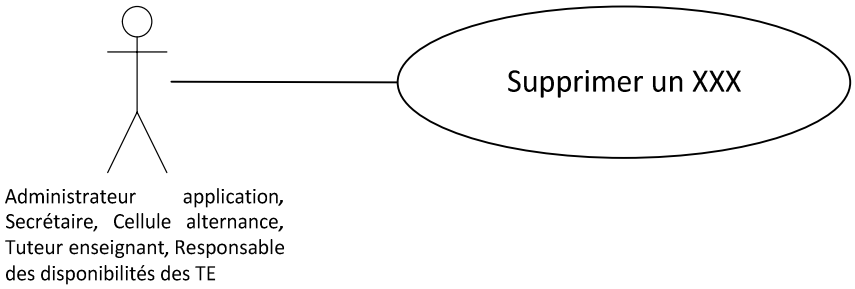


Figure 20 : Diagramme du cas d'utilisation Supprimer un XXX

✓ Acteurs

- Administrateur Application,
- Secrétaire,
- Cellule alternance,
- Tuteur enseignant,
- Responsable des disponibilités d'un TE.

✓ Description

Cette action permet aux acteurs concernés de supprimer un XXX du système.

✓ Priorité

Priorité
3

✓ Pré-conditions

- L'application est lancée,
- L'utilisateur est authentifié,
- L'utilisateur se situe sur la page de gestion de XXX concerné,
- XXX concerné existe.

✓ Tableau conversationnel

Actions menés par l'acteur	Actions réalisés par le système
1. Demande la suppression de XXX.	
	2. S'assure du choix de suppression.
3. Confirme la suppression.	
	4. Accède à la base de données et y supprime XXX.
	5. Informe l'utilisateur que XXX est

supprimé.

✓ Post-conditions

- XXX est supprimé.

✓ Flots alternatifs

c. L'utilisateur annule la suppression lors de l'étape 3 de la conversation

Actions menées par l'acteur	Actions réalisées par le système
	2. S'assure du choix de suppression.
3.a. Annule la suppression.	

✓ Exceptions

Erreur levée	Actions à réaliser par le système
L'accès à la base de données lors de la suppression de XXX a échoué.	La suppression n'est pas effectuée, l'utilisateur reste dans le menu de gestion de XXX concerné.

✓ Besoins IHM

- Un élément pour demander la suppression d'un XXX,
- Un message de demande de confirmation ou annulation,
- Une barre des tâches pour informer de la suppression de XXX.

✓ FQM

Fonction	Fréquence	Qualité	Mesure
Supprimer un XXX	2 : occasionnelle	Rapidité	Instantanée.
		Sécurité	Gérer les accès concurrents aux données de la base.

Diagramme de séquence

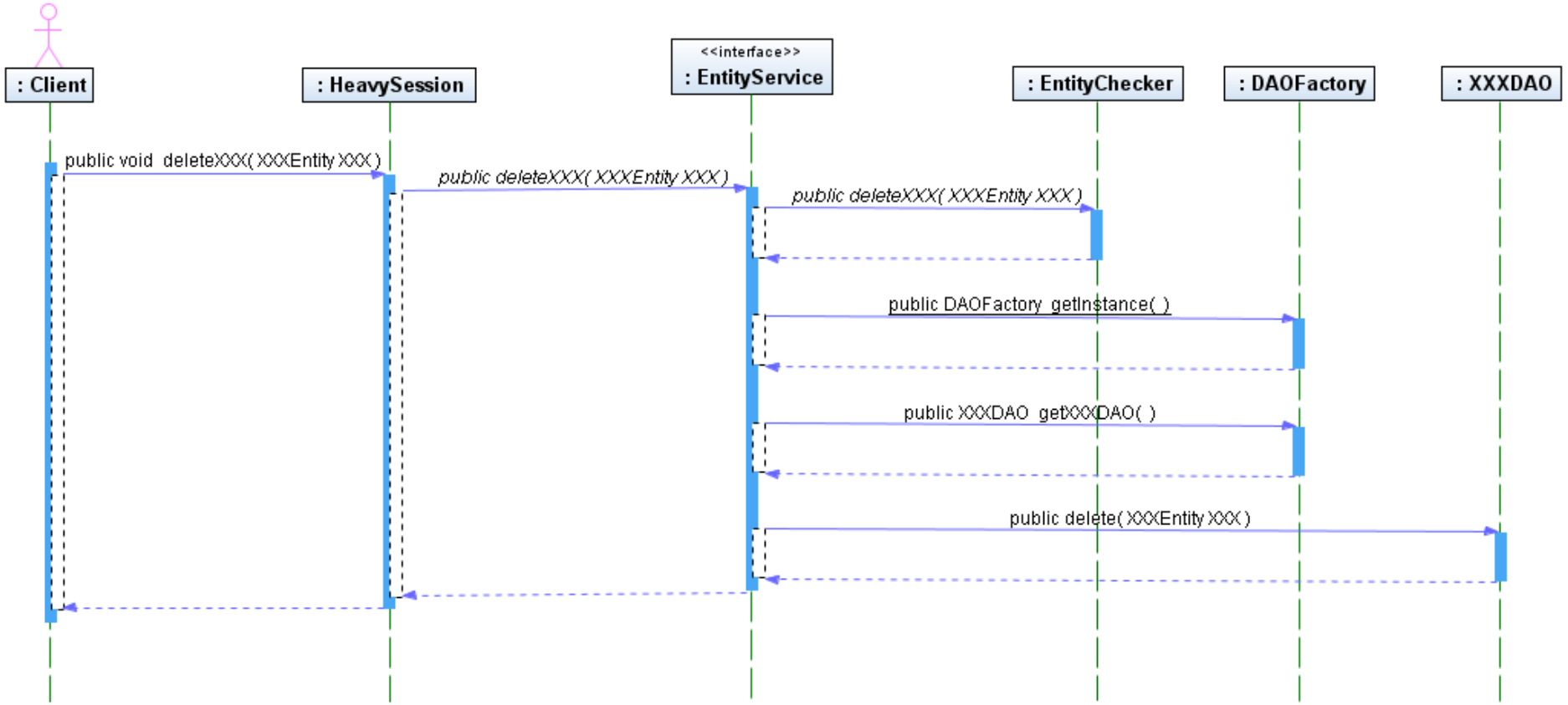


Figure 21 : Diagramme de séquence Supprimer un XXX

4.7 USE CASE générique n°6 : Visualiser un XXX

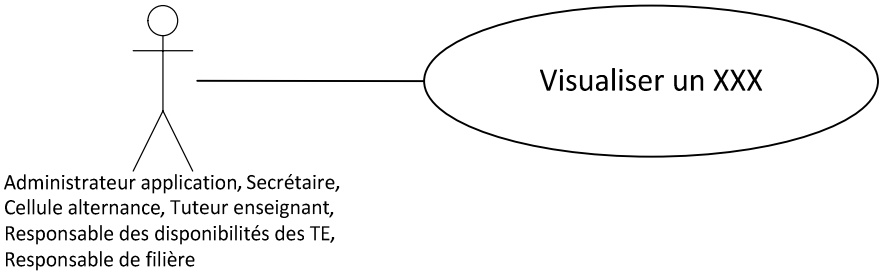


Figure 22 : Diagramme du cas d'utilisation Visualiser un XXX

✓ Acteurs

- Cellule Alternance,
- Secrétaire,
- Administrateur application,
- Tuteur enseignant,
- Responsable des disponibilités des TE,
- Responsable de filière.

✓ Description

L'utilisateur sélectionne et visualise un XXX du système du logiciel HIP.

✓ Priorité

Priorité
3

✓ Pré-conditions

- L'utilisateur doit être logé,
- L'utilisateur se trouve sur la page de gestion de XXX concerné,
- XXX concerné existe.

✓ Tableau conversationnel

Actions menés par l'acteur	Actions réalisés par le système
1. Sélectionne un XXX.	
2. Demande à visualiser cet XXX.	
	3. Affiche la vue permettant de visualiser le XXX.

✓ Post-conditions

- L'objet est visualisable par l'utilisateur.

✓ Exceptions

Erreur levée	Actions à réaliser par le système
Lors de la sélection de la vue des données de XXX (action 2), l'accès à la base de données est indisponible.	Les données de XXX ne sont pas affichées et un message d'erreur apparaît.

✓ Besoins IHM

- Un élément pour demander la visualisation de XXX,
- Une zone d'affichage structurée permettant d'afficher les informations de XXX.

✓ FQM

Fonction	Fréquence	Qualité	Mesure
Visualiser un XXX	3 : régulière	Rapidité	Affichage inférieur à une seconde.
		Ergonomie	Lecture facile et rapide.

Diagramme de séquence

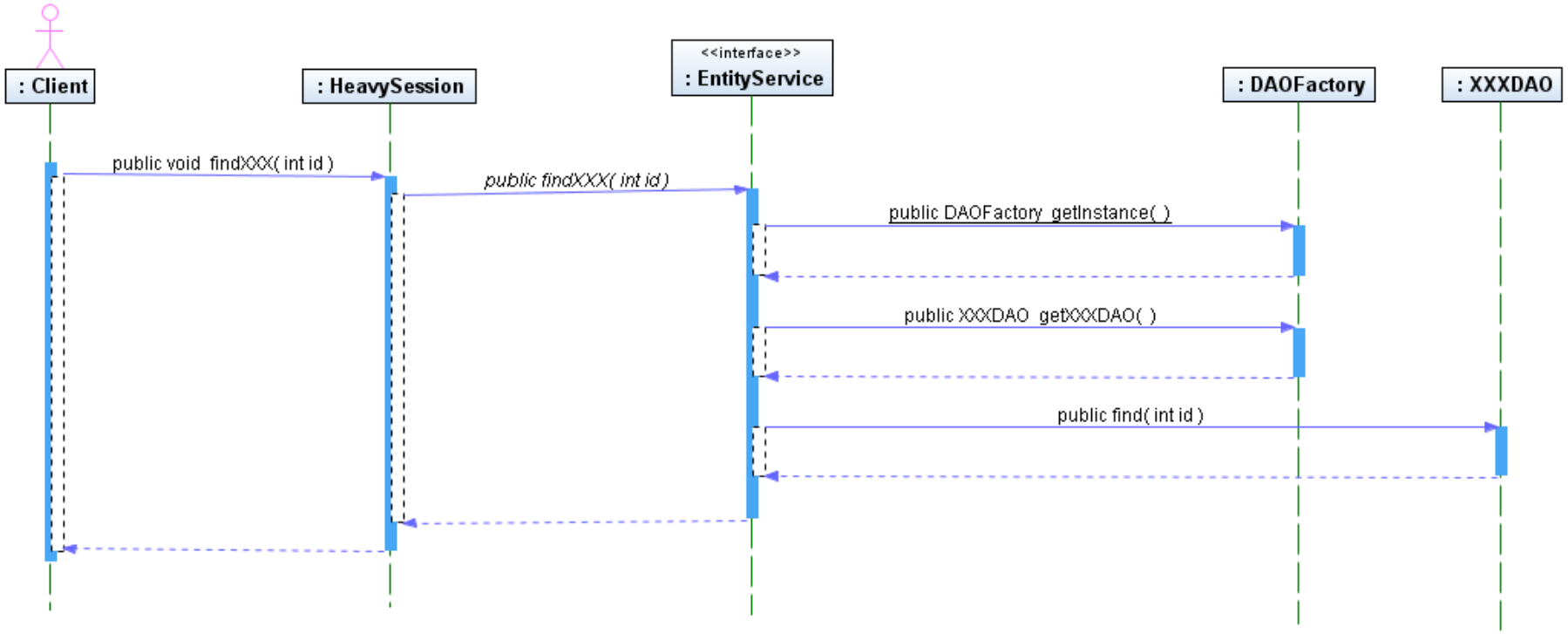


Figure 23 : Diagramme de séquence visualiser un XXX

5 Diagrammes de séquence

Nous allons présenter pour chaque lot du logiciel tous les diagrammes de séquences qui dépendent de Use Cases non génériques. Cette partie sera itérative et sera complétée lors de chaque rendu de version du logiciel.

5.1 Diagrammes de séquence de la version 1 du logiciel

5.1.1 Gestion des apprentis

Ce lot contient les UCs suivants :

- Accéder à la page de gestion des apprentis,
- Visualiser la liste apprentis,
- Importer des apprentis,
- Ajouter un apprenti,
- Visualiser les informations d'un apprenti,
- Supprimer un apprenti.

Les diagrammes de séquence des UCs *ajout d'un apprenti*, *accéder à la page de gestion des apprentis*, *visualiser la liste des apprentis*, *visualiser les informations d'un apprenti* et *supprimer un apprenti* sont traités dans la partie UCs génériques.

5.1.1.1 Importer un apprenti

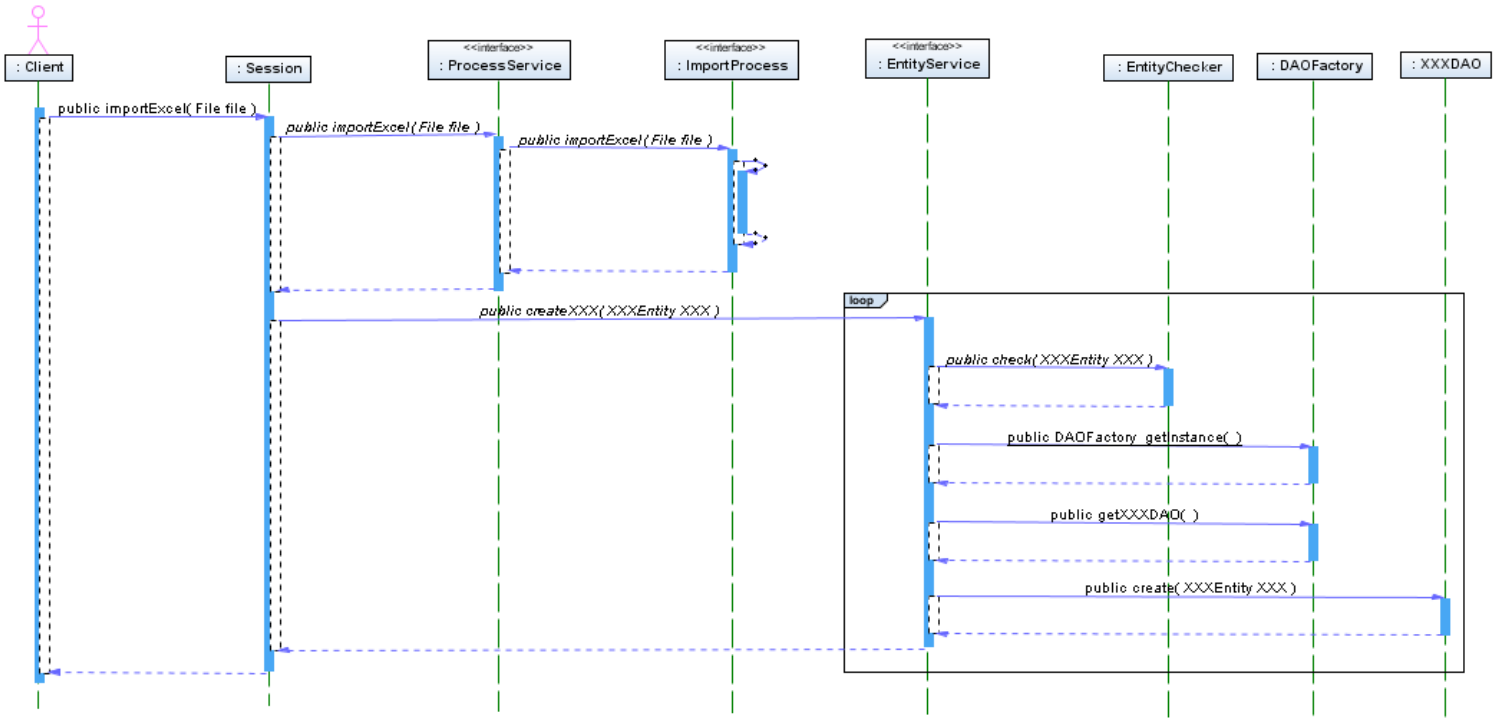


Figure 24 : Diagramme de séquence : importer un apprenti

L'opération pour importer le fichier Microsoft Excel se fait de la même manière sur le client léger que sur le client lourd. La seule différence est l'objet session qui est un HeavySession dans le cas du client lourd et un LightSession dans le cas du client léger.

L'opération peut être coupée en deux étapes :

- Dans un premier temps, le client informe la couche session en appelant la méthode improtExcel() avec comme paramètre le contenu du fichier. La session informe alors à son tour la couche ProcessService en appelant la méthode importExcel(). La couche ProcessService lance alors le traitement qui consiste principalement à parser le fichier. Une fois le traitement fini, le résultat remonte à la couche session.

- Dans un deuxième temps, la couche session crée en boucle toutes les nouvelles entités comme pour le Use Case générique de création. Une fois toutes les entités créées et ajoutées à la base de données, la couche session informe le client que la tâche est terminée.

5.1.2 Authentification et gestion des droits des utilisateurs

Ce lot contient les UCs suivants :

- Se loger,
- Se déloger,
- Créer un utilisateur,
- Supprimer un utilisateur,
- Modifier un utilisateur,
- Afficher un utilisateur,
- Modifier le profil des utilisateurs.

Les diagrammes de séquence des UCs *créer un utilisateur*, *supprimer un utilisateur*, *modifier un utilisateur*, *afficher un utilisateur* et *modifier le profil des utilisateurs* sont traités dans la partie UCs génériques.

5.1.2.1 Se loger

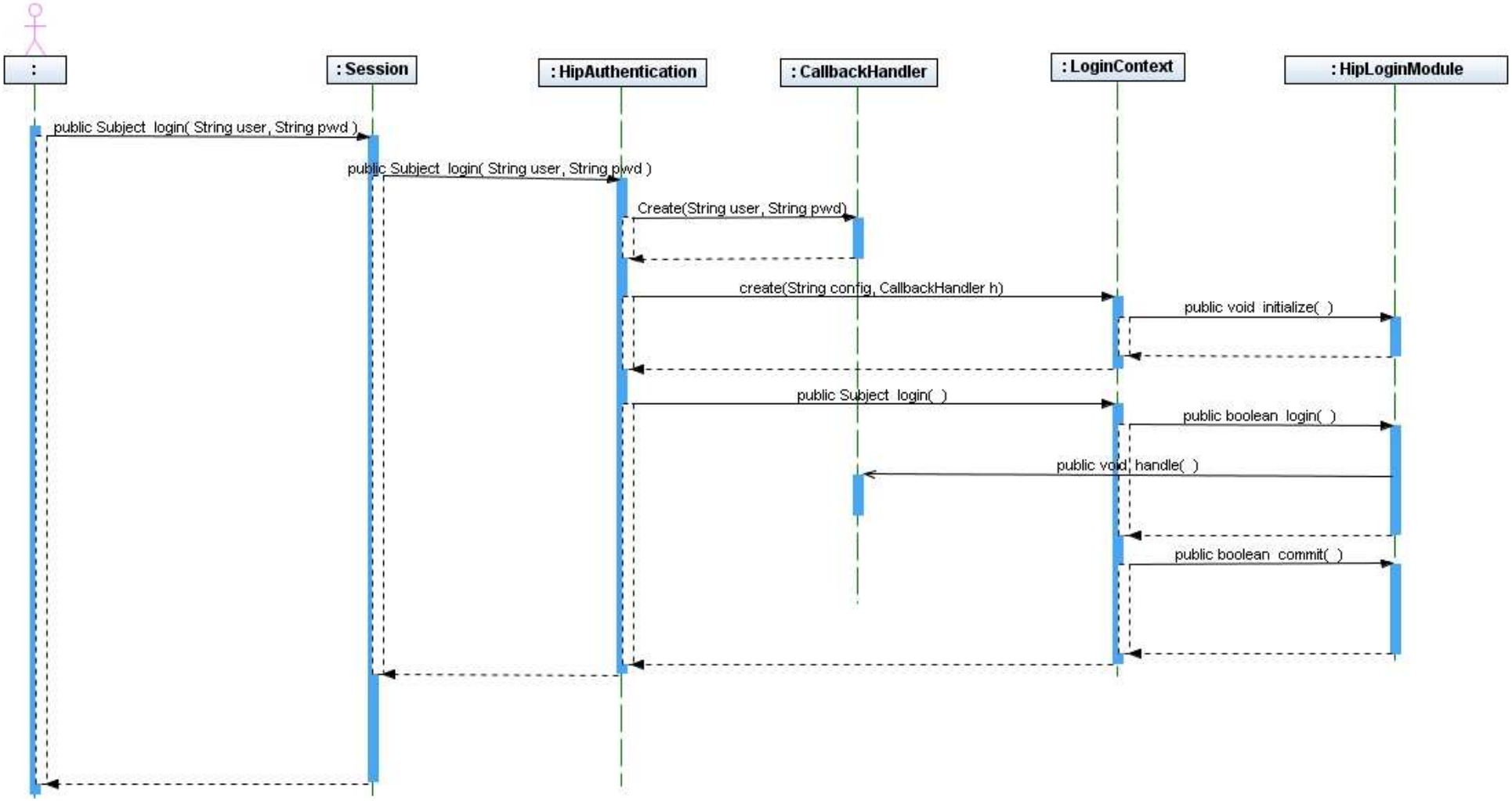


Figure 25 : Diagramme de séquence : se loger

L'action *se loger* est la même pour le client lourd et le client léger. La seule chose qui diffère est l'objet Session : pour le client lourd son nom est HeavySession alors que pour le client léger, c'est LightSession.

La classe HipLoginModule représente les deux LoginModule de Hip : LDAPHipLoginModule et LocalHipLoginModule.

Dans un premier temps, l'utilisateur appelle la méthode login() sur l'objet session en lui fournissant un login et un mot de passe. La session appelle ensuite la méthode login() de la classe HipAuthentication.

La classe HipAuthentication crée un LoginContext en fournissant le nom de la configuration la représentant. Ce nom est inscrit dans le fichier de configuration de JAAS. La classe fournit, lors de la création d'un LoginContext, un CallbackHandler qui permet de stocker le login et le mot de passe de l'utilisateur.

Lorsque l'on crée le LoginContext, celui-ci déclenche la méthode initialize() de tous les LoginModule répertoriés au niveau de la configuration sélectionnée du fichier de configuration.

Ensuite, HipAuthenticate appelle la méthode login() du LoginContext :

- Cette méthode appelle tous les LoginModule indiqués pour cette configuration (LDAPHipLoginModule et LocalHipLoginModule). Si, pour un des LoginModule, elle trouve une correspondance entre le login et le mot de passe, elle crée un sujet authentifié.

- Ensuite, elle appelle la méthode commit() du LoginModule, qui permet d'assigner des rôles à l'utilisateur.

- Pour finir, elle renvoie un objet Subject qui représente l'utilisateur authentifié et qui contient une liste de ses rôles.

5.1.2.2 Se déloger

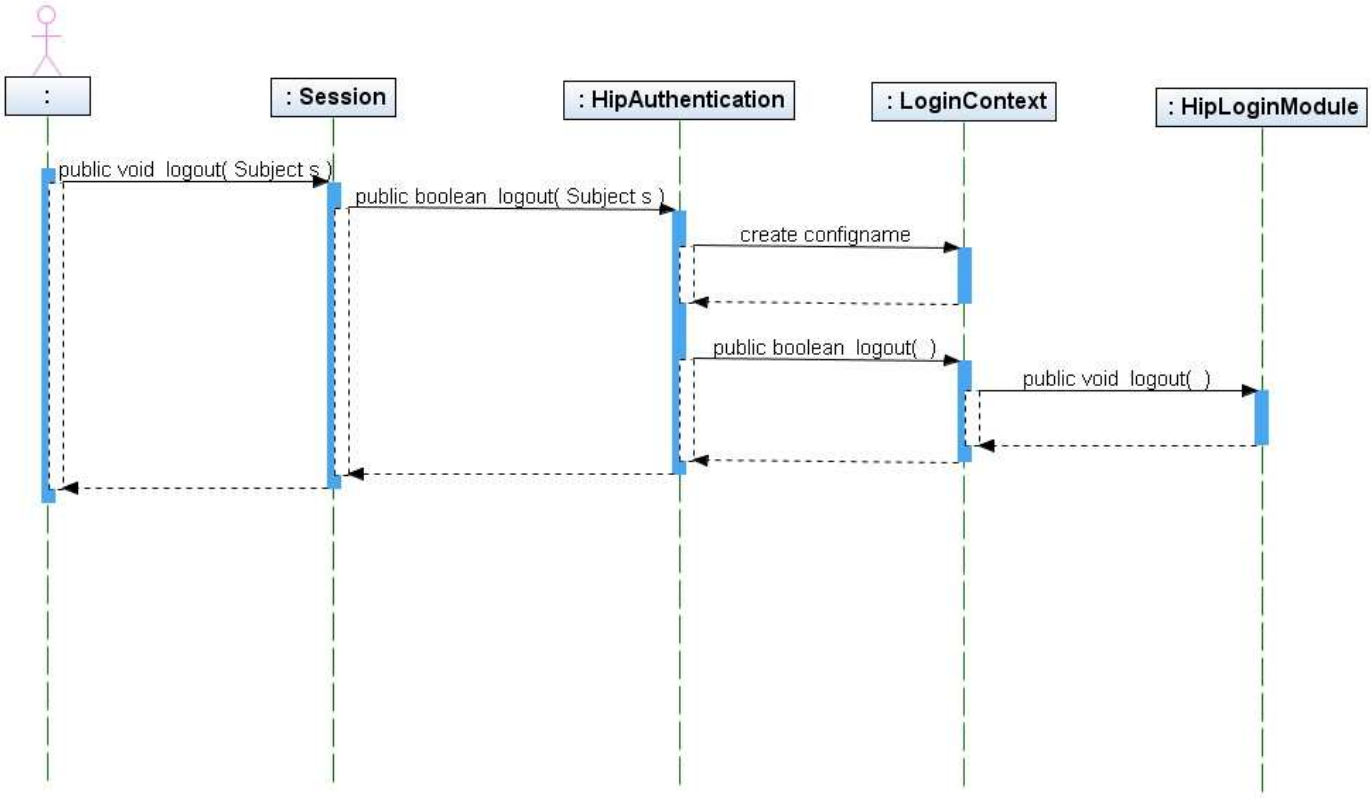


Figure 26 : diagramme de séquence : se déloger

De même que pour les parties précédentes, l'opération *se déloger* est identique sur le client lourd et sur le client léger, seul l'objet Session diffère.

Pour commencer, l'utilisateur appelle la méthode logout() sur l'objet session en lui passant l'objet Subject de l'utilisateur. Puis, la session appelle la méthode logout() de la classe HipAuthentication.

Ensuite, La classe HipAuthentication crée un LoginContext en fournissant le nom de la configuration la représentant. Ce nom est inscrit dans le fichier de configuration de JAAS.

Pour finir, HipAuthenticate appelle la méthode logout() du LoginContext. Après l'appel de cette méthode, l'utilisateur est délogé.

5.2 Diagrammes de séquence de la version 2 du logiciel

5.2.1 Gestion des Disponibilités

Ce lot contient les UCs suivants :

- Accéder à la page de gestion des disponibilités des tuteurs enseignants,
- Ajouter un tuteur enseignant,
- Modifier les informations d'un tuteur enseignant,
- Supprimer un tuteur enseignant,
- Saisir les disponibilités d'un tuteur enseignant,
- Modifier les disponibilités d'un tuteur enseignant,
- Visualiser les disponibilités d'un tuteur enseignant.

5.3 Diagrammes de séquence de la version 3 du logiciel

5.3.1 Gestion des plannings

Ce lot contient les UCs suivants :

- Accéder à la page de gestion des plannings,
- Sélectionner un planning,
- Créer un planning,
- Visualiser un planning (choix entre les différentes vues),
- Supprimer un planning,
- Enregistrer un planning,
- Générer un planning,
- Valider un planning,
- Modifier les paramètres d'un planning,
- Ajouter une soutenance à un planning,
- Supprimer une soutenance d'un planning,
- Visualiser les informations d'une soutenance d'un planning,
- Modifier les informations d'une soutenance d'un planning,
- Ajouter un enseignant de communication à la liste des enseignants devant assister aux soutenances,
- Supprimer un enseignant de communication à la liste des enseignants devant assister aux soutenances,
- Saisir les disponibilités des enseignants de communication,
- Affecter automatiquement les enseignants de communication aux différentes soutenances,
- Consulter les soutenances non casées dans le planning,
- Placer une soutenance non casée dans le planning,
- Rechercher dans le planning.

5.3.2 Gestion des convocations

Ce lot contient les UCs suivants :

- Accéder à la page de gestion des convocations.
- Visualiser la liste des convocations d'un un planning donné,
- Visualiser une convocation,
- Supprimer une convocation,
- Modifier une convocation,
- Envoyer une convocation,
- Envoyer toutes les convocations,
- Rechercher une convocation dans la liste des convocations,
- Sélectionner un modèle pour les convocations,
- Modifier le modèle des convocations.

5.4 Diagrammes de séquence de la version 4 du logiciel

5.4.1 Impression

Ce lot contient les UCs suivants :

- Imprimer un planning,
- Imprimer les convocations,
- Imprimer les informations d'un apprenti,
- Imprimer les disponibilités d'un tuteur enseignant.

5.4.2 Export

Ce lot contient les UCs suivants :

- Export d'un planning,
- Export des convocations,
- Export des informations d'un apprenti,
- Export des disponibilités d'un tuteur enseignant.

5.4.3 Internationalisation

Ce lot contient les UCs suivants :

- Modifier la langue du logiciel.

6 Répartition des rôles pour la V1

Le tableau suivant présente les prévisions d'affectation des personnes de l'équipe aux tâches pour la réalisation de la V1.

Les tâches principales sont découpées en trois entités :

- Le serveur d'application,
- Le client lourd,
- Le client léger.

Tous les membres de l'équipe travailleront sur chacune de ses entités afin de parfaire leurs connaissances et de ne pas se focaliser sur l'une d'entre elles.

D'autre part, chaque sous-tâche ou paquetage sera à réaliser par au minimum deux membres, afin de toujours travailler en équipe de développement.

Quand au chef de projet, il supervisera l'ensemble, sans prendre part au départ à la réalisation du code.

Entité	Tâche	Equipe				
		Louis ANDRIET	Julien DASTUGUE	Sébastien LAROCHE	Michaël MADEGARD	Cédric MYLLE
Serveur d'application	JPA	✓	✓			✓
	EJB (Traitement métier)	✓	✓	✓	✓	
Client lourd	Interface graphique			✓	✓	✓
	Authentifi-cation	✓	✓			
	Gestion des apprentis	✓	✓			

Entité	Tâche	Equipe				
		Louis ANDRIET	**Julien DASTUGUE**	**Sébastien LAROCHE**	**Michaël MADEGARD**	**Cédric MYLLE**
Client léger	Interface graphique (JSF)			✓	✓	✓
	Authentifi-cation	✓	✓			
	Gestion des apprentis	✓	✓			
	Gestion des droits des utilisateurs			✓	✓	✓

Conclusion

Le présent document a permis de définir, de façon technique et approfondie, tous les éléments utiles à la préparation du projet HIP, que ce soit en termes de technologies ou de concepts.

Dans un premier temps, le cahier des charges technique a défini, de façon détaillée, l'architecture générale de notre application et dégagé les besoins technologiques qu'implique cette architecture. Les couches fonctionnelles de notre application ont ainsi pu être identifiées.

En second lieu, les besoins technologiques de l'application ont conduit l'équipe Hybrid à effectuer une recherche des solutions logicielles existantes qui répondent à ces besoins. Une étude comparative et précise des logiciels convenables a permis à l'équipe de retenir, pour chaque besoin, la solution qui paraissait la plus adaptée à notre architecture.

Enfin, ce document a décrit, pour le serveur et les clients de notre application, l'architecture conceptuelle du logiciel. Cela se traduit par la définition des paquetages fonctionnels de notre solution et des classes métiers qui les composent. Ces éléments représentent la structure de notre logiciel. Les diagrammes de séquences et cas d'utilisation génériques de notre solution ont, pour terminer, été décrits dans cette partie.

Glossaire

Administrateur application :

Personne ayant les droits pour gérer les comptes et mots de passe du système d'information.

Algorithme :

Assemblage d'instructions à suivre afin d'obtenir l'exécution d'un programme ou d'une tâche.

Annuaire LDAP :

Annuaire extérieur à l'application contenant des informations utiles pour identifier toutes les personnes ayant un compte informatique à l'école Ingénieurs 2000. Ces informations sont le login, le mot de passe, le nom, le prénom et l'adresse mail.

Apache :

Logiciel de serveur Web diffusé en logiciel libre. C'est actuellement le standard du marché, loin devant les solutions exclusives.

Apache License (Apache Software License) :

La licence Apache est une licence de logiciel libre et open source. Elle est écrite par l'Apache Software Foundation, qui l'applique à tous les logiciels qu'elle publie.

Application Programming Interface ou API :

Interface de programmation qui permet de définir la manière dont un composant informatique peut communiquer avec un autre.

Apprenti :

Personne passant l'épreuve de la soutenance ou du mémoire.

Base de données ou BDD :

Ensemble structuré permettant de stocker des informations.

Bug ou bogue :

Anomalie dans un programme informatique l'empêchant de fonctionner correctement.

Candide :

Personne participant au mémoire en tant que non expert.

CDC : Cahier Des Charges

Document regroupant les besoins et contraintes du client servant de base à la signature du contrat avec ce dernier.

CDCF : Cahier Des Charges Fonctionnel

Document regroupant l'ensemble des cas d'utilisation du logiciel débouchant sur la première version du logiciel.

CDCF : Cahier Des Charges Technique

Document représentant les réponses techniques du cahier des charges fonctionnel. Il regroupe l'architecture logicielle du projet ainsi que les choix techniques ayant été effectués.

Cellule alternance :

Personnes ayant le droit de saisir des informations pour la génération des soutenances (Mesdames C. LEBON et G. TOSTAIN).

Client léger :

Dans le cadre d'une application "web", on parlera de client léger en parlant du navigateur internet.

Client lourd :

Un client lourd est un logiciel qui propose des fonctionnalités complexes avec un traitement autonome. Contrairement au client léger, le client lourd ne dépend du serveur que pour l'échange des données dont il prend généralement en charge l'intégralité du traitement.

Common Development and Distribution License (CDDL) :

La CDDL (Common Development and Distribution License) est une licence open source créée par Sun Microsystems, basée sur la Mozilla Public License, version 1.1.

Compte :

Couple login/mot de passe permettant à une personne de s'authentifier sur le système d'information.

Convocation :

Avis écrit permettant à un apprenti de connaître la date, le lieu et les membres présents lors de sa soutenance ou de son mémoire.

Conteneur d'EJB :

Chaque instance d'un EJB se construit et vie dans un conteneur. Le conteneur est en fait un environnement d'exécution fournissant des services aux Entreprise JavaBeans. Parmi ces services, on a par exemple les connexions à la base de données, les transactions, la sécurité, la persistance…

CPL :

La Common Public License est une licence utilisée pour les logiciels libres.

Diagramme de collaboration :

Diagramme modélisant les interactions entre le système d'information et ses différents. Le diagramme précise, pour chaque interaction, les fonctionnalités qui interviennent.

Design pattern:

Les *Design patterns,* ou motifs de conception, sont des façons de programmer éprouvées et réputées pour apporter des propriétés comme la cohérence, la robustesse, la réutilisabilité, etc.

Disponibilité :

Plage horaire ou journée(s) pendant laquelle une personne sera disponible pour participer à une soutenance ou un mémoire d'ingénieur.

Données en entrées :

Les données saisies en entrées regroupant les informations nécessaires à la génération du planning des soutenances. Les données en entrées peuvent être soit sous la forme d'un fichier Microsoft Excel, soit saisies manuellement via une interface du système d'information.

Données du planning :

Représentent toutes les données relatives aux soutenances et mémoires d'ingénieur pouvant être modifiées manuellement par la cellule d'alternance une fois le planning généré.

Droits (ou privilèges) :

Qualités accordées à un compte dans le but de contrôler l'accès à une partie du logiciel.

EJB :

La technologie Enterprise JavaBeans (EJB) est une architecture de composants logiciels côté serveur pour la plateforme de développement J2EE.

EPL :

L'Eclipse Public License est une licence libre utilisée par le logiciel Eclipse.

Excel :

Tableur logiciel développé par Microsoft.

Expert :

Personne participant au mémoire en tant qu'expert dans une seule et unique filière. Généralement un Tuteur Ingénieur d'un autre apprenti.

Export :

Sauvegarder de données dans un format externe à l'application.

FAQ :

Frequently Asked Questions ou Foire Aux Questions : document régulièrement mis à jour, regroupant les réponses aux questions les plus fréquemment posées.

Filière :

Représente un pôle d'enseignement disposé au sein d'Ingénieurs 2000 et donc une spécialité du diplôme d'un apprenti.

FQM :

Les FQM sont les fonctions que le système exécute pour permettre à un utilisateur d'atteindre un objectif par l'application. Les fonctions sont définies par "Le système fait une action" et répondent à un certain nombre de qualités mesurables.

Framework:

En informatique, un framework est un espace de travail modulaire. C'est un ensemble de bibliothèques, d'outils et de conventions permettant le développement rapide d'applications. Il fournit suffisamment de briques logicielles et impose suffisamment de rigueur pour pouvoir produire une application aboutie et facile à maintenir.

Fusion :

Convocation générée pour les soutenances. Regroupe les différentes informations nécessaires à la soutenance (nom, prénom de l'apprenti, date, salle et noms des participants). Toutes ses informations sont éditables manuellement par la Cellule Alternance.

GNU General Public License (GPL) :

La Licence publique générale GNU, ou GNU GPL pour GNU General Public License en anglais, est une licence qui fixe les conditions légales de distribution des logiciels libres du projet GNU.

Informations :

Données relatives à un apprenti, telles que ses nom, prénom, adresse, sexe et âge, mais aussi les coordonnées de ses tuteurs ou encore ses disponibilités (ainsi celle de ses tuteurs) en cas de préparation au mémoire d'ingénieur.

J2EE :

Java 2 Platform Enterprise Edition est une plateforme pour le langage de programmation Java destiné aux applications d'entreprise comme les Servlets (conteneur Web) ou JSP (Framework Web).

Java :

Java est une technologie développée par Sun Microsystems : (la technologie Java™). Elle correspond à plusieurs produits et spécifications de logiciels qui, ensemble, constituent un système pour développer et déployer des applications.

JDBC :

Java Database Connectivity: une interface standardisée permettant à une application cliente sous Java d'accéder à une base de données.

Java Persistence API :

L'API de persistance des données JPA fait parti de la spécification EJB3. Spécification EJB3 qui fait elle-même parti de la plate-forme J2EE 5.0. La persistance des données en EJB3 est possible à l'intérieur d'un conteneur EJB3 aussi bien que dans une application autonome J2SE en dehors d'un conteneur particulier. Cette API réalise la fusion des travaux sur Hibernate avec la continuité des spécifications EJB précédentes 2.0 et 2.1.

Java Server Faces :

JSF est un framework destiné à la plate-forme Java, permettant de développer des interfaces graphiques améliorées pour des applications web exécutées côté serveur. Il utilise un modèle orienté événement, JSF se compose d'un ensemble d'API servant notamment à représenter les composants graphiques, à gérer les états et à supporter les événements. Il dispose également d'un ensemble de balises conçues pour exprimer les interfaces JSF à l'intérieur d'une page JSP.

Java Standard Tag Library :

JSTL est une bibliothèque de tags utilisables dans les pages JSP qui encapsule sous forme de simples balises les fonctionnalités de base d'un grand nombre d'applications Web. Elle fournie un support pour les tâches communes et structurelles comme l'itération et le conditionnel, des balises pour la manipulation des documents XML, des balises d'internalisation ainsi que des balises pour le SQL.

JNDI :

NDI est une extension JAVA standard qui fournit une API uniforme permettant d'accéder à divers services de noms et de répertoires. Derrière un nom JNDI, il peut y avoir un appel à des services CORBA, DNS, NIS, LDAP... En fait, JNDI permet de localiser et d'utiliser des ressources.

JSP :

La technologie JSP (Java Server Pages) est une extension de la notion de Servlet permettant de simplifier la génération de pages web dynamiques. Elle se sert de balises semblables au XML ainsi que de scriptlets Java afin d'incorporer la logique de fabrication directement dans le code HTML. JSP est un conçurent direct de l'ASP et du PHP.

LGPL :

La Licence publique générale limitée GNU, ou GNU LGPL (pour GNU Lesser General Public License en anglais), est une licence utilisée initialement pour les librairies, mais qui l'est aussi pour certains logiciels libres (comme OpenOffice.org).

Elle présente de grandes ressemblances avec la Licence publique générale GNU (ou GNU GPL), rédigée par le même organisme, la Free software foundation, visant à promouvoir le développement de logiciels libres. La différence entre ces deux licences réside principalement dans le fait que la LGPL permet de lier un programme tiers non-GPL à une bibliothèque LGPL, sans pour autant étendre la licence. Ainsi, il devient possible à un programmeur désireux de faire un logiciel propriétaire, d'utiliser certains outils du libre (ex. : la bibliothèque graphique GTK).

Cette licence limitée, ou amoindrie, a été créée pour permettre à certains logiciels libres de pénétrer tout de même certains domaines où le choix d'une publication entièrement libre de toute l'offre était impossible.

Login :

Nom d'un utilisateur lui permettant de s'authentifier au système d'information.

Lot :

Regroupement de fonctionnalités communes liées aux interactions entre les acteurs et le système d'information.

Mail ou courriel :

Message électronique envoyé par le biais d'Internet.

Mémoire :

Présentation à différents participants de l'activité professionnelle d'un apprenti de troisième année parmi lesquels figurent : son tuteur enseignant, son tuteur ingénieur, un président de jury, un expert et un candide.

Mode itératif :

Lors des mémoires, les apprentis se succédant dans une salle doivent être de la même filière.

Module :

Fonctionnalité du système d'information. Ses deux fonctionnalités sont la génération du planning de soutenances et la génération du planning de mémoires.

Navigateur :

Logiciel permettant d'avoir accès à l'Intranet de l'école et/ou Internet.

Paquetage :

Regroupement effectué sur un ensemble de cas d'utilisation s'appuyant sur les mêmes concepts. Même s'ils ne correspondent pas exactement aux paquetages techniques (paquetages Java), ils les préfigurent néanmoins.

PC personnel :

Ordinateur sur lequel sera installé le système d'informations.

Plage horaire :

Représente la période de la journée durant laquelle la soutenance est dispensée. Elle définit à ce titre une heure de début et une heure de fin de soutenance.

Planning :

Regroupement des soutenances ou mémoires d'ingénieur se tenant sur une semaine.

POJO :

POJO est un acronyme qui signifie Plain Old Java Object. Il fait référence à un objet qui dispose d'un constructeur sans paramètre ainsi qu'un couple d'accesseurs (get/set) pour chacun de ses champs.

Président du jury :

Participant au mémoire en tant que jury. Il est nommé par le responsable de filière une fois le planning établi.

Professeur de communication :

Participant à une soutenance en tant que professeur de communication.

Programmation par contraintes :

La programmation par contraintes (PPC, ou *CP* pour *Constraint Programming* en anglais) consiste à programmer avec des relations appelées contraintes. Un problème comporte un certain nombre de variables, chacune ayant un domaine (l'ensemble des valeurs que peut prendre cette variable), et un certain nombre de contraintes. Une contrainte implique une ou plusieurs variables, et restreint les valeurs que peuvent prendre simultanément ces variables. Trouver une solution à un problème de PPC consiste à décrire l'ensemble des affectations autorisées de chaque variable, de telle sorte que la totalité des contraintes soient satisfaites.

Responsable des disponibilités :

Acteur responsable de renseigner les disponibilités des tuteurs enseignants, dans le cadre des soutenances des 1ères et 2èmes années.

Secrétaire :

Personne saisissant les informations sur les apprentis de la filière dont elle est responsable.

Serveur d'applications :

Un serveur d'applications est un serveur sur lequel sont installées des applications accessibles par le réseau à des utilisateurs.

Servlet :

Une servlet est une application Java qui permet de générer dynamiquement des données au sein d'un serveur HTTP. Ces données sont le plus généralement présentées au format HTML, mais elles peuvent également l'être au format XML ou tout autre format destiné aux navigateurs Web.

Ce programme Java s'exécute dynamiquement sur le serveur Web et permet l'extension des fonctions de ce dernier, typiquement : accès à des bases de données, transactions d'e-commerce, etc. Une servlet peut être chargée automatiquement lors du démarrage du serveur Web ou lors de la première requête du client. Une fois chargées, les servlets restent actives dans l'attente d'autres requêtes du client.

SGBD ou système de gestion de bases de données :

La gestion et l'accès à une base de données sont assurés par un ensemble de programmes qui constituent le Système de gestion de base de données.

Solaris :

Système d'exploitation de type Unix développé par Sun Microsystems.

Soutenance :

Épreuve durant laquelle un apprenti de première ou deuxième année présente son activité professionnelle aux participants présents : Tuteur Ingénieur, Tuteur Enseignant et Professeur de communication.

SQL :

Structured query language (SQL), ou langage structuré de requêtes, est un pseudo-langage informatique (de type requête) standard et normalisé, destiné à interroger ou manipule une base de données relationnelle.

Système d'information :

Environnement informatique qui sera mis en place pour répondre aux besoins et contraintes du client.

Tuteur enseignant :

Enseignant ou personne liée à la formation responsable de l'évolution d'un ou plusieurs apprentis, notamment en période académique. Il participe aux soutenances et mémoires de son (ses) apprenti(s).

Tuteur ingénieur :

Ingénieur responsable de l'évolution d'un ou plusieurs apprentis durant la séquence professionnelle. Il participe aux soutenances et mémoires de son (ses) apprenti(s) et joue généralement le rôle d'expert pour le mémoire d'un autre apprenti.

UML : Unified Modeling Langage

Langage de modélisation pour la programmation orientée objet.

Upload :

Transfert d'un fichier de l'ordinateur vers un logiciel ou un serveur.

Utilisateur :

Personne se connectant au système d'information sans avoir de droit sur la gestion des comptes. Il s'agit principalement des secrétaires, de la cellule alternance, des responsables de disponibilités, de l'administrateur application et des tuteurs enseignants.

Table des illustrations

Figure 1 : Architecture J2EE classique 7
Figure 2 : Architecture Hip 8
Figure 3 : exemple SwingX 19
Figure 4 : Exemple d'utilisation de SwingX 20
Figure 5 : Diagramme des paquetages du serveur HIP 22
Figure 6 : Diagramme de classes du paquetage "core" 23
Figure 7 : Diagramme de classes du paquetage "data" 24
Figure 8 : Diagramme de classes du paquetage « entity » 25
Figure 9 : Diagramme de classes du paquetage « session » 27
Figure 10 : Diagramme de classes du paquetage « dao » 29
Figure 11 : Diagramme de classe du paquetage auth 30
Figure 12 : Diagramme des paquetages du client lourd HIP 32
Figure 13 : Diagramme des paquetages du paquetage « view » 33
Figure 14 : Diagramme du cas d'utilisation Accéder à la <page> 35
Figure 15 : Diagramme du cas d'utilisation Utiliser l'application d'un <client> 37
Figure 16 : Diagramme du cas d'utilisation Ajouter un XXX 39
Figure 17 : Diagramme de séquence Ajouter un XXX 41
Figure 18 : Diagramme du cas d'utilisation Modifier un XXX 42
Figure 19 : Diagramme de séquence Modifier un XXX 44
Figure 20 : Diagramme du cas d'utilisation Supprimer un XXX 45
Figure 21 : Diagramme de séquence Supprimer un XXX 47
Figure 22 : Diagramme du cas d'utilisation Visualiser un XXX 48
Figure 23 : Diagramme de séquence visualiser un XXX 50
Figure 24 : Diagramme de séquence : importer un apprenti 52
Figure 25 : Diagramme de séquence : se loger 54
Figure 26 : diagramme de séquence : se déloger 56