Eau Vive
Gestion d'un stade pour kayakistes

Par
QUAGHEBEUR Florian – BESANÇON Thomas
BAILLEUL Alexandre – SIMON Yann

Sommaire

Introduction:

Dans le cadre de notre formation au sein de l'IUT Informatique de Caen, nous sommes amenés à étudier une méthode de modélisation des logiciel nommée UML. Dans le cadre de cet enseignement, nous avons analysé et produit un logiciel de gestion d'un stade pour kayakistes fonctionnant en fonction du cycle des marées.

Le principe de fonctionnement du stade est ainsi prévu : la marée montante rempli une grande réserve d'eau qui assure, une fois la marée retirée du stade, l'alimentation en eau du stade. Le système informatique doit donc pouvoir gérer le remplissage de la réserve et l'écoulement de l'eau dans le stade selon plusieurs débits correspondants aux différents niveaux des kayakistes attendus dans le stade.

En dehors du domaine géré par le logiciel, le gérant du stade assure la sécurité des kayakistes vis a vis du stade ainsi que le placement des obstacles dans le stade lorsque ce dernier est vide.

Après cette introduction rapide au projet que nous avons mené, nous allons reprendre les différentes parties de notre étude, en commençant par résumer nos cours d'UML puis en détaillant notre étude générale du projet Eau Vive et pour finir par les détails de notre analyse du système a concevoir.

1. Les cours

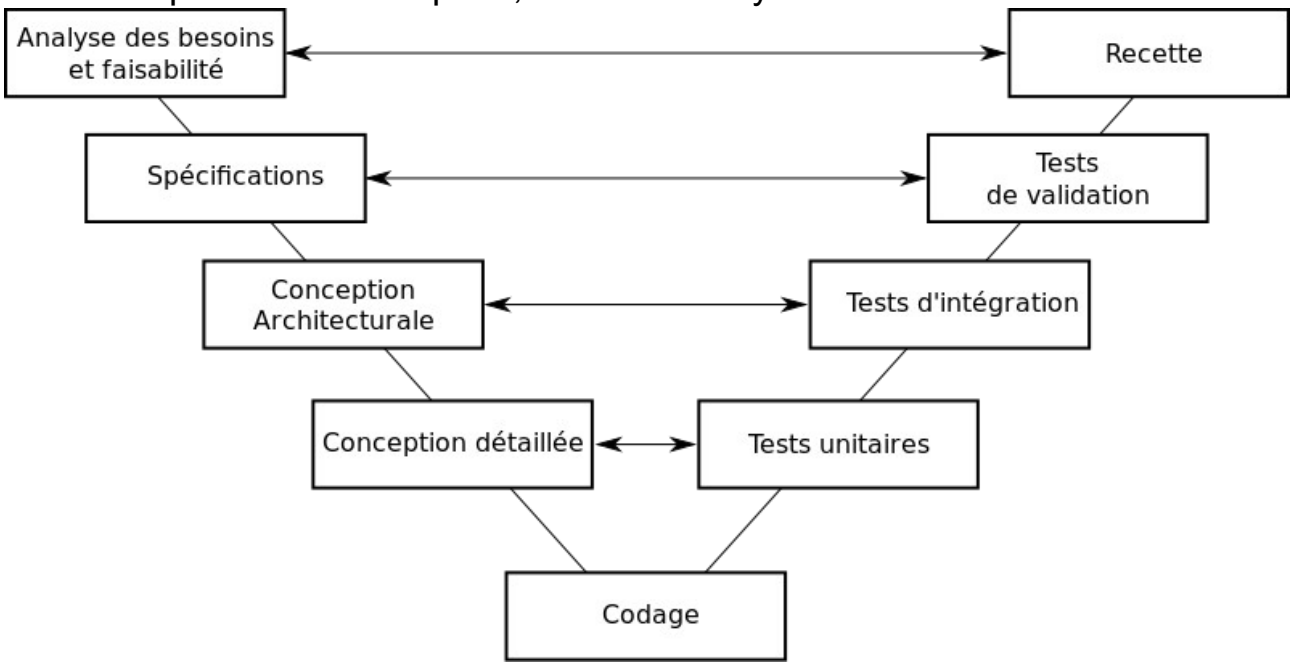
1. Le contexte d'un projet

Afin de mener à bien un développement logiciel, il est nécessaire de se doter de connaissances du domaine concerné par le logiciel : les connaissances métier. Le risque d'une absence de connaissances métier est l'incompréhension entre le développeur et le client. De fait, l'équipe de développement doit chercher à obtenir toutes les informations nécessaires au déroulement du projet. Le client peut penser à tort qu'une fonctionnalité est implicite dans son énoncé, le développeur doit alors le percevoir et en parler clairement avec le client. Le contrat ou cahier des charges doit poser tout le projet à l'écrit afin qu'il n'y ait pas d'omissions dans les tâches à réaliser, servant ainsi de bornes au développeur du projet.

2. Les méthodes de développement cyclique

Le cycle en V

La méthode du cycle en V consiste a réaliser tout le projet étape par étape. On réalise ainsi la définition de tous les besoins, puis la spécification des besoins, puis toute la phase de conception, etc. Ainsi le cycle en V suit le modèle suivant :



Le défaut du cycle en V (autrefois recommandé) est l'absence de perception des problèmes du logiciel avant le rendu de ce dernier. Par exemple, une erreur d'analyse des besoins n'est perçue qu'à la validation du logiciel, bien trop tard pour refaire toutes les étapes nécessaires pour la corriger.

De plus, en cas d'abandon du projet, celui ci est difficilement reprenable car toute la documentation est réalisée après la validation et donc en fin de projet.

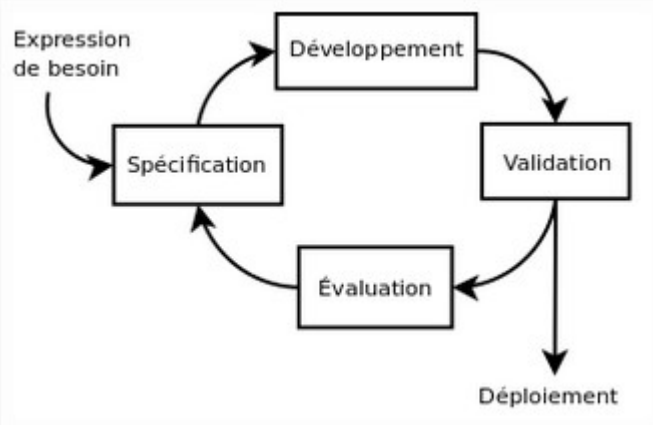
Le cycle itératif incrémental

Les méthodes agiles sont bien meilleures que le cycle en V car elles fonctionnent par production de lots. Un lot achevé contient une version fonctionnelle du projet et a été réalisé avec toutes les étapes d'un cycle en V

Un développement par cycles itératifs et incrémentaux correspond ainsi à plusieurs petits cycles en V successifs menant chacun une fonctionnalité du logiciel à son terme.

L'intérêt est qu'un projet important se décompose alors en projets courts(lots) qui une fois terminés sont documentés. Il est alors possible de reprendre facilement un projet suivant cette méthode et de repérer les erreurs dans le logiciel bien avant le rendu du projet.

Un cycle itératif incrémental est donc modélisé comme suit :



3. Les modèles de conception logicielle

Le problème de la modélisation est celui des modèles multiples. En effet, il existe de multiples méthodes de modélisation selon les habitudes de chaque équipe de développement.

Il est donc nécessaire de fusionner tous ces modèles afin de permettre aux équipes de mieux communiquer entre elles : la méthode UML. De même, les langages de développement se sont limités pour éviter trop de disparités dans les méthodes de programmation

par exemple :

les automates anciennement réalisés en Cobol, sont désormais réalisés en C ou dans un autre langage dérivé

Les réseaux en entreprise tendent aussi vers une norme pour assurer une meilleure compatibilité. La plupart emploient TCP/IP de nos jours.

Les composantes du langage UML

Le langage UML emploi une série de diagrammes dont la visée varie. Chacun de ces diagrammes est cependant relié aux autres puisqu'ils se complètent mutuellement.
Ces diagrammes sont les suivants :

Le Diagramme de Cas d'Utilisation(DCU)

Le DCU étudie les acteurs qui interagissent avec le logiciel et explicite dans quelles actions ces acteurs interviennent. L'ensemble des fonctionnalités du logiciel est ainsi établie dans ce diagramme qui sert de base à la conception analytique du logiciel.

Le Diagramme d'Objet

Ce diagramme sert de base au diagramme de classes en résumant les objets qui devront être utilisés dans le logiciel.

Le Diagramme de Classes

Conçu dans un premier temps a partir des deux diagrammes précédents, il est celui qui est au plus près du développeur puisqu'il modélise tout le code que ce dernier devra produire. Au final, ce diagramme recense tous les objets, les méthodes et les fonctions qui sont à implémenter dans le logiciel.
Ce diagramme est à faire évoluer au fil des étapes de la modélisation UML. Les diagrammes suivants sont donc susceptibles de modifier le diagramme de classes.

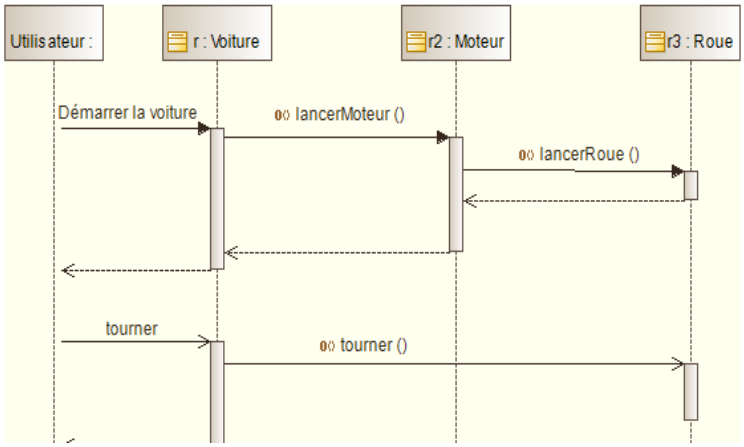
Le Diagramme de Séquence

Le diagramme de séquence représente les intéractions nécessaires entre le système et les acteurs pour mener une tâche à bien. Il représente ainsi chronologiquement les échanges entre le système et les acteurs.

Le diagramme de séquence emploi trois types de flèches pour modéliser les interactions entre les éléments du système : la flèche pleine (—►) représentant un message bloquant et la flèche vide (—>) modélisant un message non bloquant, enfin, la flèche en pointillé (- - - -►) représente pour sa part la réponse d'une fonction, c'est à dire sa valeur de retour.

Chaque élément intervenant du système est modélisé par une boite sous laquelle une ligne indique la ligne temporelle de l'objet. Un rectangle sur cette ligne signifie qu'une fonction est en cours d'exécution. De part la logique de ce diagramme, une fonction ne peut s'activer que si une flèche appelante (pleine ou vide) l'appelle.

Toutes ces notions sont représentées dans le diagramme ci dessous.



Lancer le moteur d'une voiture est synchrone, en effet, inutile de demander autre chose a la voiture pendant le démarrage du moteur(mettre le contact).

A l'inverse, tourner le volant n'empêche pas de faire une autre action. Il s'agit donc d'un message asynchrone.

Le Diagramme de Communication

Ce diagramme est l'équivalent du diagramme de séquence mais orienté vers les interactions entre objets du système. A la différence du diagramme de séquence, il se focalise sur la succession d'appel de méthodes dans le logiciel et ne prend donc pas en compte les acteurs extérieurs.

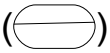
Sa conception emploie à la fois le diagramme de classes, le DCU et le diagramme de séquence puisqu'il s'agit de représenter les interactions des objets(classes) du système avec les utilisateurs et de tenir compte de l'ordre dans lequel les événements se produisent.

La modélisation du diagramme de communication est très proche de celle du diagramme de séquence. Les classes sont reliées par des liens (lignes simples) et les appels de fonction entre elles sont modélisés par les mêmes flèches pleines et vides qui représentent le même type d'appel.

A partir des diagrammes de séquence et de communication, une partie des attributs et méthodes du diagramme de classes apparaissent et peuvent donc lui être ajoutés.

Le Diagramme d'Etats transitions

Ce diagramme représente les différents états que peut prendre un objet, une Classe, pendant l'exécution du logiciel. Il en décrit ainsi la dynamique en présentant les différents états dans lesquels la classe peut se trouver et comment elle s'y retrouve.

Les différents éléments constitutifs d'un diagramme d'états transitions sont le rectangle () incarnant un état avec les événements qui s'y produisent, les points noir () et cerclés () indiquent respectivement les états d'origine et finaux d'une classe, enfin, les flèches représentent quand à elles les transitions entre états souvent associées à un texte () présentant la condition de transition.

Les états peuvent contenir des fonctionnalités qui sont celles employées par cet état. Chaque fonctionnalité est associé à un type d'utilisation propre à l'état. Ce type peut être "*entry :* " pour indiquer une méthode appelée au début de cet état ou "*exit :* " pour indiquer les méthodes s'activant en fin d'état. Les transitions d'état étant instantanées, il ne s'agit alors que de méthodes dites instantanées, c'est à dire courtes et sans boucles dans leur traitement. Les autres types sont "*do :* " qui exprime une méthode employée toute la durée de l'état, enfin "*on <condition> :* " représente le cas d'une méthode employée dans un certain cas de cet état, par exemple en fonction de l'état d'où provient la classe.

Les autres diagrammes

Il existe d'autres diagrammes que nous n'avons pas étudié en détail dans ce cours. Ces diagrammes sont rassemblés ci-après.

Nom du diagramme	utilité
Le Diagramme d'Activité	Le diagramme d'activité permet de modéliser les actions effectuées par une fonctionnalité du système. En d'autres termes, il représente le parcours algorithmique d'une fonction dans le système.
Le Diagramme de Déploiement	Le diagramme de déploiement incarne la structure physique du logiciel. Y sont ainsi représentés les terminaux, serveurs et autres capteurs qui font partie du logiciel. On y modélise aussi les moyens d’interaction entre tout ces appareils.
Le Diagramme de Composants	

2. Analyse générale du projet

1. Présentation du stade

Le stade eau-vive se base sur la mécanique de la marée pour assurer le fonctionnement d'un parcours de kayak. Le stade schématisé ci dessous présente les éléments fonctionnels importants du stade

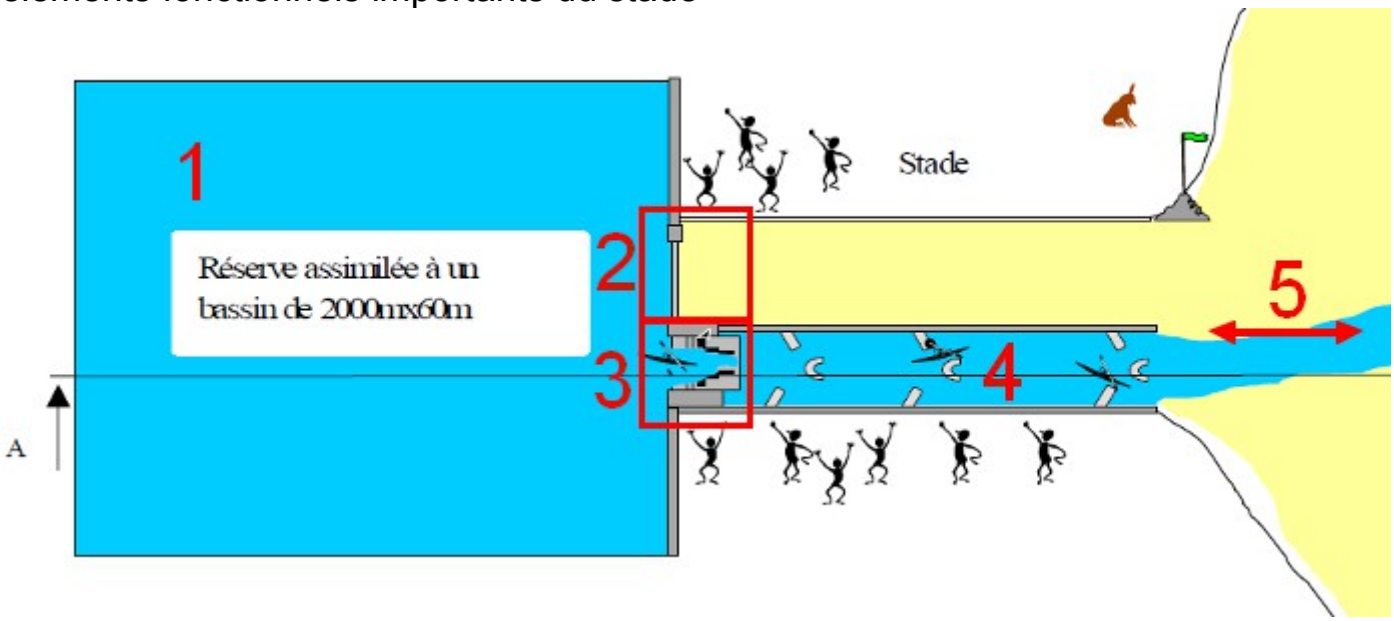


Schéma des éléments du stade

Cinq éléments principaux sont a prendre en compte dans la mécanique du stade :

1) La réserve d'eau qui assure l'alimentation du stade pendant son fonctionnement.
2) La vanne StockVide permet de purger la réserve d'eau lorsque la marrée descend pour lutter contre l'envasement dans celle-ci. Elle permet aussi le remplissage de la réserve pendant la marée montante.
3) La vanne Omniflot gère le débit d'eau envoyé dans le stade et en conséquence la difficulté du parcours dans le stade.
4) Le stade est le lieux où, lorsque le niveau de la mer est en dessous de ce dernier, les kayakistes pratiquent leur sport.
5) La marée donne le rythme des séances et assure le rechargement de la réserve.

Le stade contient deux points de repère importants : le bas du stade, point le plus bas de l'édifice il délimite la plus haute mer à laquelle une séance peut avoir lieue. Le radier, niveau minimal de la réserve, a ce niveau la réserve ne peut plus fournir d'eau au stade et la séance doit donc être interrompue.

2. Etude du cycle de vie du stade

Pour mener a bien ce projet, il nous a fallu comprendre le fonctionnement de la marée et l'utilisation de l'eau dans le stade. En conséquence, nous avons analysé les variations de niveau dues à la marée et l'écoulement de l'eau dans le stade.

Étude du remplissage

Coefficient de la marée	**45**	**60**	**95**
Niveau Hydrographique de pleine mer (en m)	**7 m**	**8 m**	**9 m**
Niveau NGF de Pleine Mer (PM) (en m)	7 - 5 = **2 m**	**3 m**	9 – 5 = **4 m**
Niveau NGF de l'eau dans la réserve à PM +3 (en m) (vannes totalement relevées)	**2 m**	**3 m**	**3,5 m**
Niveau NGF de l'eau dans la rivière artificielle à PM +3 (vannes totalement relevées)	**0 m**	**0 m**	**0 m**

Etude de l'utilisation du stade

Débit d'utilisation	**4 $m^3.s^{-1}$**	**14 $m^3.s^{-1}$**
Hauteur minimum d'eau (en m) au dessus du radier permettant d'assurer le débit demandé	**0,4 m**	**0,9 m**
Volume d'eau utilisable dans la réserve en m^3 (on tiendra compte de la contrainte précédente)	**2 000*60*0,4 = 48 000 m^3**	**2 000*60*0,9 = 108 000 m^3**
Pendant combien de temps (en h:mm) la réserve pourra t-elle fournir le débit souhaité.	**48 000 / 4 = 03:20:00**	**108 000 / 14 = 02:09:00**
Étant données les contraintes sur les périodes d'utilisation du stade (niveau de la mer <0 m NGF), le volume utile d'eau sera-t-il totalement utilisé ?	**OUI**	**OUI**
Combien de temps faut-il pour que le niveau d'eau dans la réserve baisse de 1 cm ?	**2 000*60*0,01/4 = 00:05:00**	**2 000*60*0,01/14 = 00:01:25**

Horaires d'utilisation du stade

Coefficient de la marée	**45**	**95**
A marée descendante, à quelle heure (par rapport à la pleine mer) peut on commencer à utiliser le stade ?	**PM + 3**	**PM + 3**
A marée montante, à quelle heure (par rapport à la pleine mer précédente) doit on cesser d'utiliser le stade ?	**PM + 9**	**PM + 9**
Entre deux Pleines Mer, quelle est la durée maximale d'utilisation du stade ?	**6 heures**	**6 heures**

Cycle de vie du stade

De nos études précédentes sur le cycle de la marée et le fonctionnement du stade, nous avons pus en déduire les moments importants de la vie du stade. Quatre moments sont ainsi de première importance dans le cycle de vie du stade : la pleine mer (notée PM), le moment où le niveau de la mer croise le niveau du bas du stade : PM+3 et PM+9, respectivement a marée descendante et marée montante, et enfin, le manque d'eau dans la réserve entre PM+3 et PM+9.

Les intérêts de ces quatre moments sont résumées dans le tableau suivant :

Moment	Événement important	Conséquence
PM	Pleine mer	La réserve est pleine
PM+3	La mer passe sous le niveau du stade	La séance peut commencer
Manque d'eau	La réserve arrive a un niveau critique	La séance doit s'arrêter
PM+9	La mer passe au dessus du niveau du stade	La séance doit s'arrêter Le gérant du stade amorce le remplissage de la réserve

Afin de présenter l'importance de chaque moment, nous allons simuler deux cas de figure concernant le cycle de vie du stade : le cas d'un faible débit pour un fort coefficient de marée, cas où la fin de la séance dépendra de la marée et non d'un manque d'eau dans le stade, puis le cas contraire où le manque d'eau sera à l'origine de la fin de la séance.

Pour chaque cas, la période représentée sera d'une pleine mer (PM) à la suivante(PM+12).

Cas 1 : fort coefficient de marée et faible débit (séance niveau initiation)

Période	Evènements
PM	La réserve est pleine On ferme les vannes pour retenir l'eau dans la réserve. Le stade est fermé
PM+3	Le stade est entièrement au dessus du niveau de la mer. La séance peut commencer sur indication du gérant du stade. **Le stade ouvre**
PM+9	La mer est revenue au niveau du stade Le gérant est informé par le système qu'il faut mettre un terme à la séance. Sur confirmation du gérant(stade vide de tout kayakiste), les vannes sont affalées pour vidanger le stade, la réserve commence ensuite a se remplir. **Le stade ferme**
PM+12	La mer est au plus haut On ferme les vannes pour retenir l'eau dans la réserve.

Cas 2 : faible coefficient de marée et fort débit (séance niveau compétition)

Période	Evènements
PM	La réserve est remplie au maximum (elle n'est pas pleine car l'amplitude de la marée n'est pas suffisante) On ferme les vannes pour retenir l'eau dans la réserve. Le stade est fermé
PM+3	Le stade est entièrement au dessus du niveau de la mer. La séance peut commencer sur indication du gérant du stade. **Le stade ouvre**
Manque d'eau	Le gérant est informé par le système qu'il faut mettre un terme à la séance car l'eau va manquer dans la réserve. Sur confirmation du gérant(stade vide de tout kayakiste), les vannes sont affalées pour vidanger le stade **Le stade ferme**
PM+9	La mer est revenue au niveau du stade La réserve commence a se remplir
PM+12	La mer est au plus haut On ferme les vannes pour retenir l'eau dans la réserve.

3. Solutions technologiques

Afin de concevoir le système de ce stade, il est nécessaire de trouver les systèmes électroniques qui permettront de récupérer et de transmettre les relevés concernant le niveau de la marée et de la réserve. Pour ce faire, nous allons utiliser deux produits de la marque VEGA qui nous permettront de faire des mesures et de les transmettre au serveur où le logiciel traitera les informations. Voici les deux produits qui seront utilisés pour les mesures des niveaux de la marée et de la réserve :

Radar Vegaflex 81

Transmetteur Vegamet 624

Les radars et le transmetteur seront reliés par un câble permettant l'échange de données, l'ordinateur et le transmetteur seront quant à eux reliés par un câble Ethernet.

Pour pouvoir utiliser le transmetteur, le serveur devra être configuré à l'aide du logiciel PACTware et devra avoir un pilote d'appareil adéquat installé.

Les radars Vegaflex 81 seront placés dans la mer et dans la réserve. Le radar communiquera avec le transmetteur Vegamet 624 qui fera la liaison entre ces derniers et le serveur. Il recevra des données provenant des radars sous forme de nombres de 32 bits à virgule flottante au format IEEE-754. Le transmetteur va convertir les données reçues en valeur sous le format souhaité sur le serveur.

Suite a ce traitement et à l'envoi des informations au serveur, le logiciel pourra alors les traiter et en déduire les actions a effectuer sur le stade ou les informations a transmettre à l'utilisateur.

3. Analyse détaillée et conception

1. Etude des fonctionnalités du stade

Dans l'objectif de réaliser le système informatique requis pour ce stade, la première étape à été de définir l'ensemble des fonctionnalités qui sont nécessaires à la gestion du stade. En ce sens nous avons établi le diagramme de cas d'utilisation du logiciel en deux temps. En premier lieu nous avons défini les cas d'utilisation simples, directement en lien avec les acteurs du système que sont la marée et le gérant du stade. Nous avons ensuite approfondi chacun de ces cas pour en détailler le fonctionnement. Ces deux diagrammes sont présentés ci-dessous :

Diagramme de Cas d'Utilisation (niveau 1 : réduit)

Diagramme de Cas d'Utilisation (niveau 2 : étendu)

Nous avons pu déduire de ces premiers éléments d'étude du fonctionnement du logiciel une première version du diagramme de classes.

Pour réaliser la première version du diagramme de classes, nous nous sommes appuyés sur le diagramme de cas d'utilisations de niveau 2. Nous avons transformé chaque cas en une fonction et en paramètres nécessaires pour la réaliser. Il s'agit ensuite de réunir ces fonctions et paramètres par thèmes qui deviendront les objets du diagramme de classe.

Certaines fonctions et distinctions de Classes sont issues de spécificités du sujet non explicites dans le diagramme de cas d'utilisations ou d'un premier effort de songer aux fonctions qui pourront être utiles pour la réalisation des autres fonctionnalités. Le tableau suivant présente la déduction des éléments du diagramme de classe déduits du diagramme de cas d'utilisation.

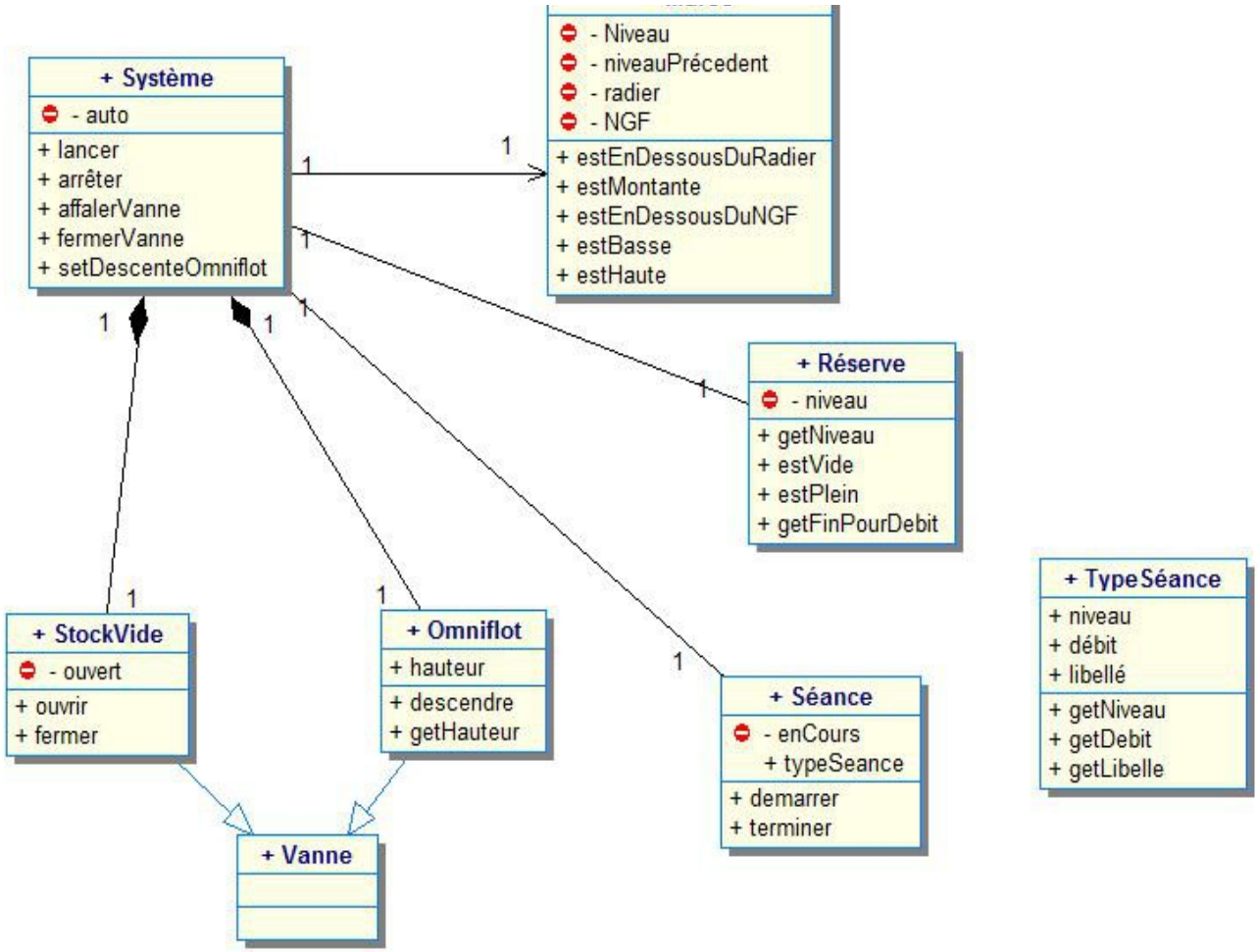
Cas d'utilisation	fonction	paramètres	Objet	
Lancer le système	lancer()		Système	
en mode automatique	lancer(auto)	Auto = vrai	Système	
en mode manuel	lancer(!auto)	Auto = faux	Système	
Placer les obstacles				hors système
Système arrêté				hors système
Vérifier que la marrée est sous le niveau du radier	estEnDessousDuRadier()	Radier	Marée	
Affaler les vannes pour la vidange	affalerVannes()		Vannes	
Débuter une séance	démarrer()	EnCours	Séance	
Arrêter le système	arrêter()		Système	
Niveau de la marée sous le niveau NGF	estEnDessousDuNGF()	NGF	Marée	
Ouverture de la vanne stockVide	ouvrir()	ouvert	StockVide	
Remplissage du stock	fermer()		Vannes	stipulé pour finir le remplissage
Marée montante	estMontante()	Niveau NiveauPrécédent	Marée	on sait si la marée monte en comparant deux relevés de niveau
Donner le niveau de la séance		Niveau	Séance	
Terminer une séance	terminer()		Séance	
Gérer la vanne omniflot	desendre(valeur)	Hauteur	Omniflot	
Vannes fermées et stock non vide	estVide()		Stock	

A cette étude, nous avons ajouté l'analyse des objets physique et informatique qui à pour but de lister les objets physique intervenant dans le système et à leur donner leur équivalent en classe au sein du logiciel. Cet élément d'étude nous a permi de vérifier qu'aucun élément n'a été oublié dans le diagramme de classe à réaliser.

Objet physique	Rôle dans le système	Objet Informatique
Ordinateur	Porteur du système	Système
Vanne Omniflot	Gère le débit dans le stade	Omniflot
Vanne StockVide	Gère le remplissage vidange de la réserve	StockVide
Marée	Rythme le fonctionnement du stade	Marée
Réserve	Stock d'eau pour le stade, son niveau doit être surveillé	Réserve
Obstacles	Placé pour les kayakistes, pas de gestion logiciel	inexistant dans le système
Capteurs	Relèvent le niveau de l'eau dans la réserve et/ou de la marée	(intégré aux objets Marée et Réserve)

Après vérification que nous n'avions rien oublié dans notre transition du diagramme de cas d'utilisation vers notre diagramme de classe, nous avons établi une première version du diagramme de classes comme suit :

Diagramme de classe (version 1)



2. Analyse de scénarios d'utilisation

Afin de parfaire le diagramme de classe précédent, nous avons modélisé sous la forme de diagrammes de séquence et de collaboration de deux cas extrêmes du fonctionnement du stade. Ces scénarios étudiés sont sensiblement proches des cas extrêmes du cycle de vie du stade présentés dans la partie 2.2. (page 9) du présent rapport. Ainsi nous avons analysé le cas d'un fort débit à faible coefficient de marée puis le cas d'un débit faible pour un coefficient de marée élevé.

Scénario 1 : débit élevé et coefficient de marée faible

A titre de rappel : nous avions observé précédemment que, dans ce cas de figure, la réserve d'eau n'est pas assez remplie pour assurer le fonctionnement du stade jusqu'au retour de la marée. Le cas étudié ici est donc celui d'une fermeture du stade pour cause de manque d'eau.

Diagramme de Séquence (scénario 1)

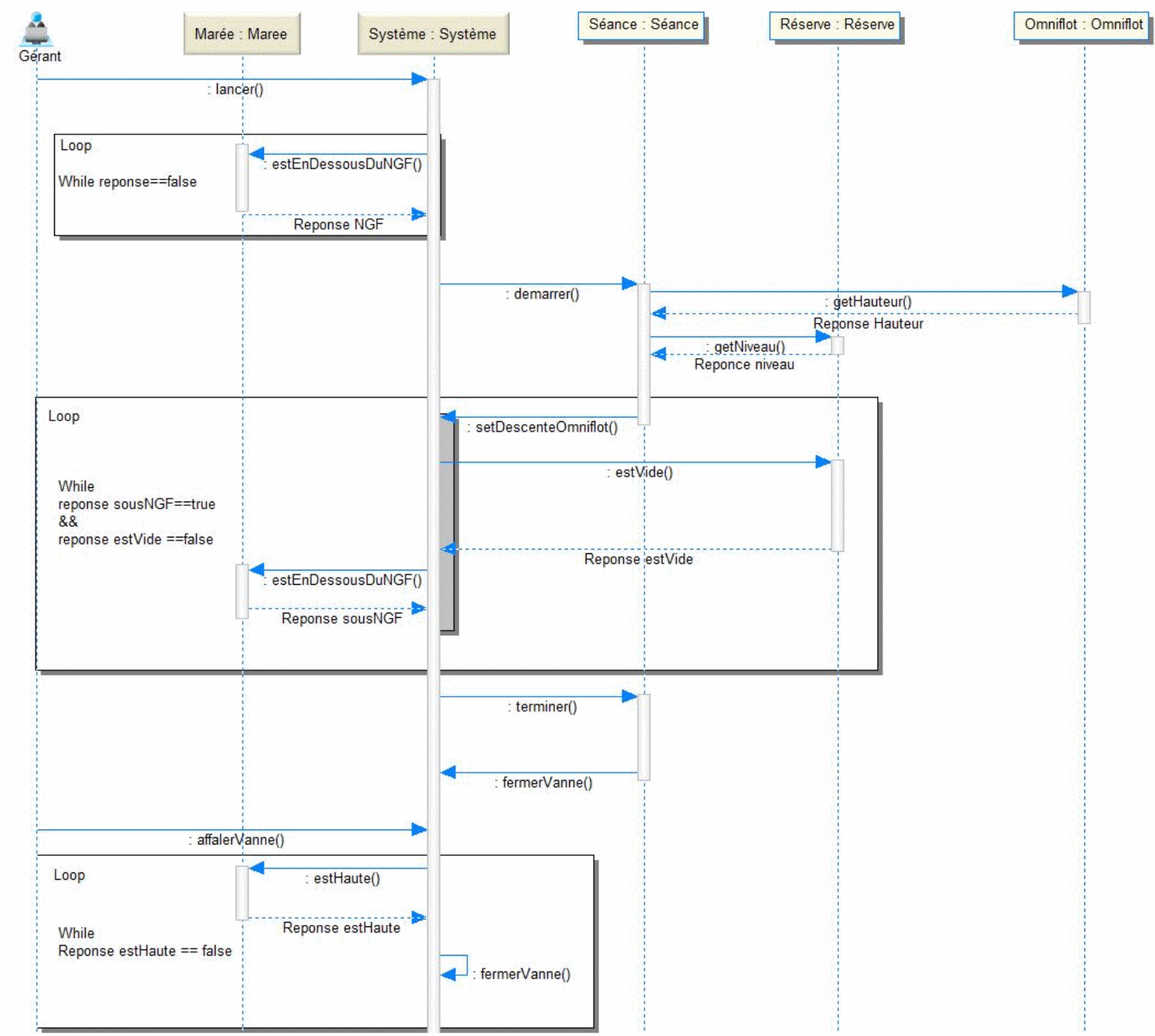
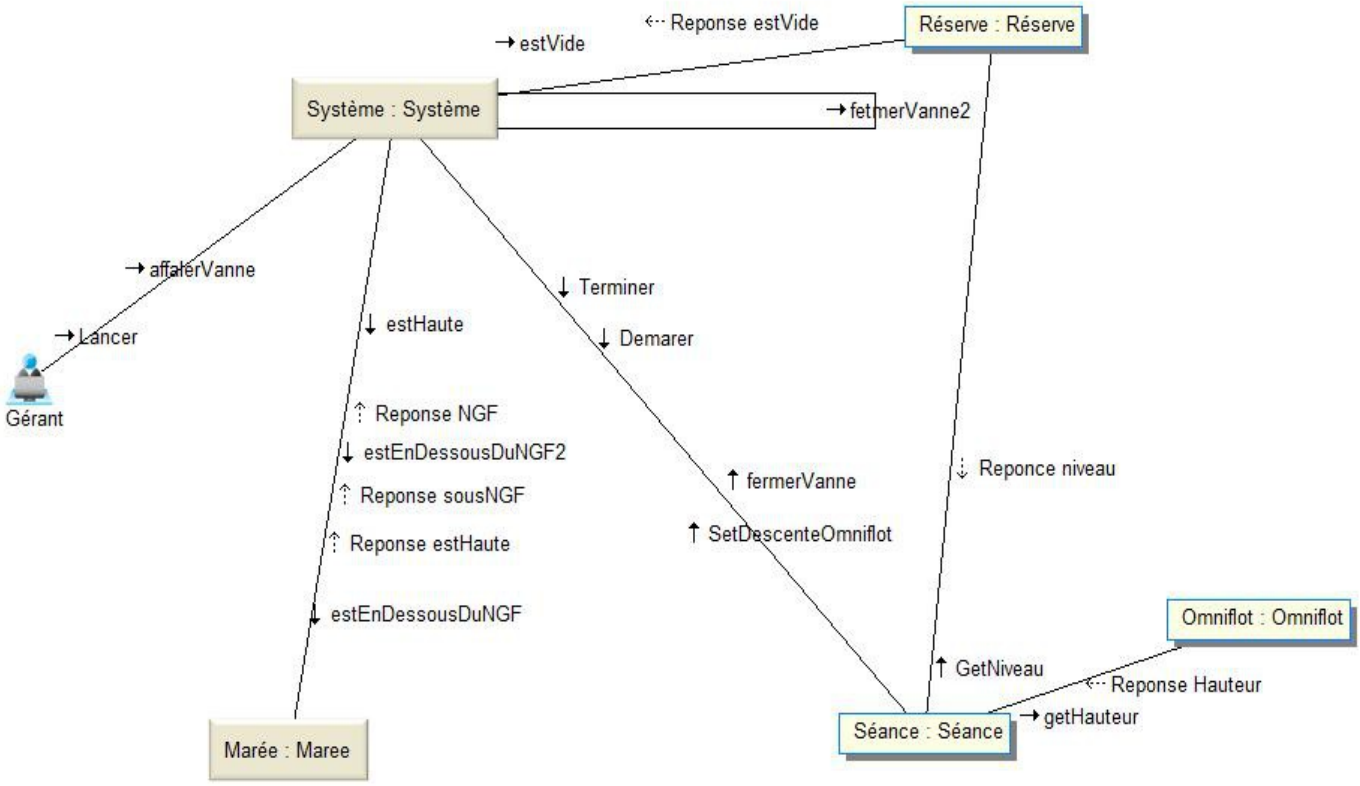


Diagramme de Collaboration (scénario 1)



Scénario 2 : débit faible et coefficient de marée élevé

Dans le cadre de ce scénario, nous avons eu à employer le diagramme de classe proposé par notre professeur. A ce titre nous avons conçu le diagramme de séquence correspondant au modèle de classes qui nous a été suggéré et présenté ci-dessous.

Diagramme de Classe (suggéré pour l'exercice)

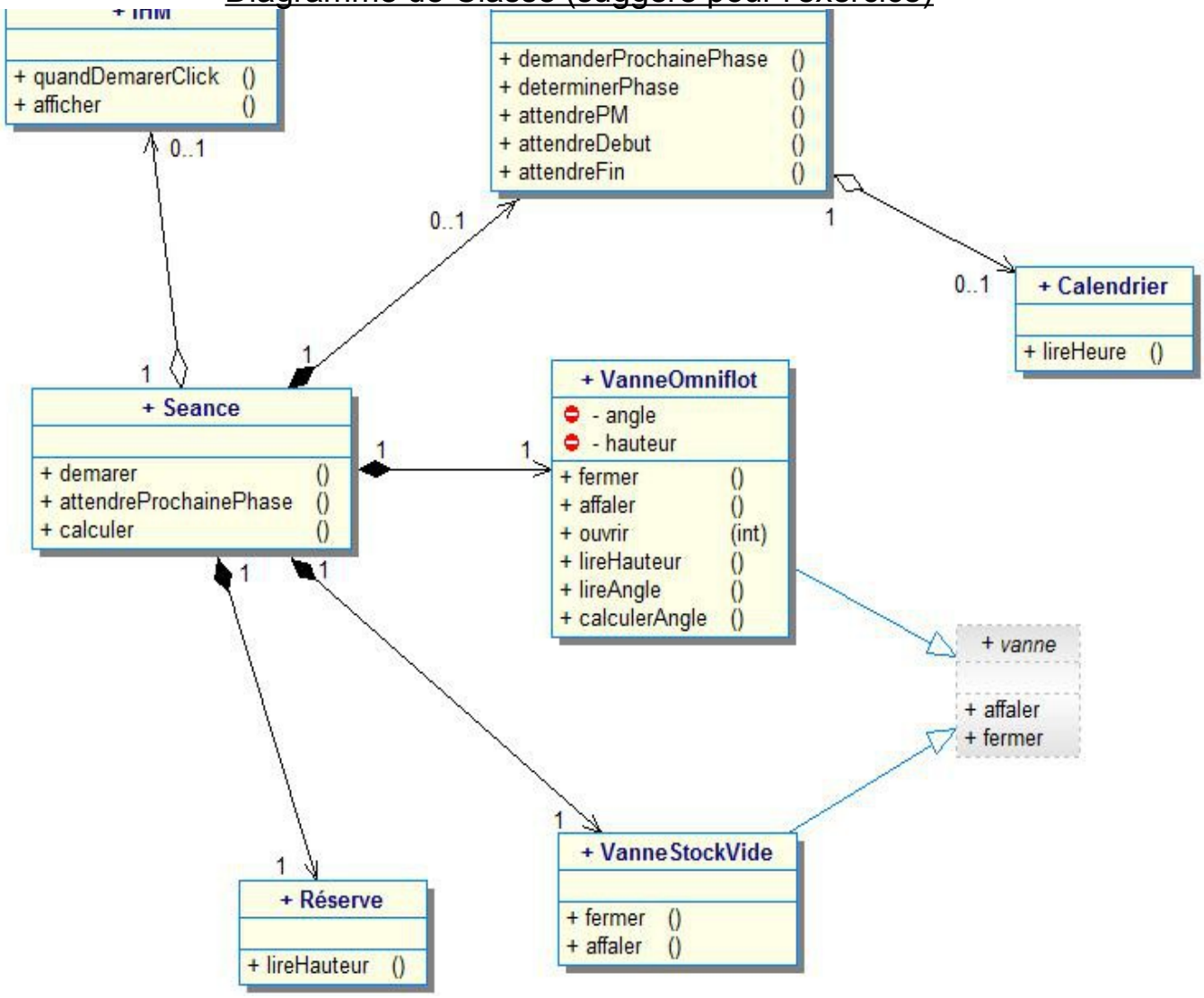


Diagramme de Séquence (scénario 2)

3. Correction du devoir

Questions générales sur la technologie.

1) Donnez la liste des types de mémoires mortes(ROM). Quelle est la particularité de la mémoire flash ?

Les types de mémoires mortes sont la ROM qui est une mémoire morte programmée ; la PROM qui est une mémoire morte vierge ; la EPROM qui est une mémoire vierge ou programmée qui a la particularité de pouvoir s'effacer avec des rayons ultraviolets ; la EEPROM qui est une mémoire morte effaçable en utilisant un courant électrique

2) Qu'est ce qu'un codeur optique. Quel est son rôle dans le système. Quel est la différence entre un codeur incrémental et un codeur absolu ?

Le codeur optique est un dispositif qui permet de mesurer un déplacement linéaire(déplacement angulaire) ou rotatif.

Il ne possède pas de mémoire. Il transmet des informations sur ses déplacements grâce à des fils de communication.

Les mesures sont faites grâce à un faisceau lumineux qui traverse des trous.- Plus il y a de trous, plus la résolution est importante et donc plus il est précis.

Le codeur relatif quant à lui envoi les informations sur ses déplacements par 2 fils de communication. L'information révèle dans quel sens le déplacement était-il orienté(horizontal) et de combien de crans il a bougé.

Enfin, le codeur absolu lui, nous offre en permanence les informations sur les déplacements linéaires lors d'un déplacement. En effet il n’envoie l'information précisément lorsque

3) L'ordinateur industriel possède un serveur WEB, un serveur TELNET et un client BOOTP. Quels sont leurs rôles dans le système et comment fonctionnent-ils techniquement ?

Le serveur TELNET permet de prendre la main à distance en mode console. Permet d'ouvrir une session. A ne pas confondre avec une application TELNET qui est un client permettant de se connecter à n'importe service en mode console.

Le serveur WEB permet de se connecter a un serveur http pour(en général) accéder à des applications graphiques. Pour le système d'eau vive, permettre les maintenances du système à distance.

Le client BOOTP permet de booter à partir d'un système d'exploitation distant

pour les machines qui n'ont pas de disque dur.

Modélisation UML

1) Dans un diagramme Use Case, quelles sont les différentes relations possibles entre 2 cas? Détaillez. Comment peut-on être sur qu'un objet ou qu'une personne est acteur/actrice du système.

Les différentes associations entre 2 cas sont include (le cas A nécessite le cas B pour se produire), extend (le cas A est une possibilité supplémentaire au cas B. Il peut se produire ou non) et generalize(héritage entre 2 cas. Un cas en spécialise un autre.)

2) Donnez la modélisation d'un système téléviseur basique avec au minimum 2 acteurs principaux avec des rôles biens distincts, 2 cas et 2 sous-cas. Choisissez les cas les plus révélateurs du système. Au moins 2 types différents de relation inter-cas doivent être mis en œuvre.

L'acteur peut interagir avec le système et si on l'enlève on peux toujours utiliser une partie du système.

Les acteurs principaux sont la télécommande et l'utilisateur, le lecteur DVD et enfin l'antenne.

Les utilisations possibles du téléviseur sont de regarder la télévision, de changer de chaîne ou de régler le téléviseur via des options de réglage.

Conclusion

Au cours de ce projet nous avons ainsi été amené à étudier un problème de conception logicielle. Suite à la réalisation de plusieurs diagrammes UML d'étude de la demande de système informatique, nous avons pu envisager la mise en œuvre du système en question.

Dans le cadre de cette réalisation, nous avons choisi d'utiliser le langage de programmation JAVA qui est le langage en moyenne le plus maîtrisé par l'ensemble des membres du groupe.

Le logiciel sera composé d'une maquette (interface graphique) et d'une partie simulation correspondant au système a concevoir. La maquette représentera le cycle de vie du stade en tenant compte de deux paramètres sélectionnables par l'utilisateur : le coefficient de marée représenté par faible, moyen et fort ainsi que le niveau de la séance présentant le débit du stade. Les étapes successives représentées par la maquette seront celles des 12 heures d'un cycle de marée, de PM à PM+11.

Ce projet d'étude d'un cas de réalisation d'un système informatique nous a beaucoup apporté en terme de connaissances et de maîtrise des outils de modélisation UML. En effet, aucun des membres du groupe ne savait, avant ce projet, comment relier les différents diagrammes UML entre eux. De même, c'est toute une partie de la réalisation d'un système à laquelle nous n'avions jamais été confronté : la partie physique et électronique, que nous avons appris à rechercher et a mettre en œuvre sur le plan théorique.