

Rapport BD50 / GL52

Sujet 8

Tour de France Cycliste



Groupe 11

Clément GHNASSIA

Yassine OUAAMOU

Thomas SIMON

Responsables d'UV

Christian FISCHER [BD50]

Abderrafiâa KOUKAM [GL52]

Table des matières

Introduction.....	7
I.Objectif du document.....	7
II.Champ d'application.....	7
III.Définitions et abréviations.....	8
IV.Organisation du document.....	9
Partie 1 Spécifications.....	10
I.Description générale.....	10
1.Domaine étudié.....	10
2.Acteurs.....	10
3.Fonctionnement du système.....	12
4.Flux.....	13
5.Contraintes de développement.....	15
6.Hypothèses et dépendances.....	16
7.Périmètre du système.....	17
8.Extensions.....	17
II.Besoins fonctionnels.....	19
1.Diagramme des cas d'utilisation	19
2.Diagrammes de séquence.....	28
3.Domain Model.....	34
Partie 2 Conception.....	36
I.Architecture.....	36
II.Conception détaillée.....	38
1.Diagramme de classe.....	38
2.Diagrammes de séquence.....	40
3.Diagrammes d'état-transition.....	45
III.Dictionnaire de données.....	49
IV.Modèle entité association.....	53
1.Gestion des inscriptions.....	53
2.Gestion de la course.....	54
3.Gestion des classements.....	55
4.Gestion du plus combatif.....	57
5.Gestion des contrôles anti-dopage.....	57
6.Gestion des utilisateurs.....	58
V.Modèle Logique Relationnel.....	59
1.MLR Normalisé.....	59
2.Tехники d'optimisation.....	64
VI.Optimisation spécifique Oracle.....	70
1.Stratégie d'indexation.....	70
2.Stratégie de partitionnement.....	72
3.MLR Optimisé.....	74
VII.Architecture applicative	78
1.Architecture technique.....	78
2.Cartographie fonctionnelle par module.....	79
Partie 3 Développement.....	81
I.Organisation des scripts SQL.....	81
II.Automatisation.....	81
III.Insertion d'un jeu de données.....	85

IV.Développement Oracle Web Toolkit.....	85
V.Développement APEX.....	92
Conclusion.....	97
Annexes.....	98
I.Modèle conceptuel de données.....	99
II.Modèle Logique Relationnel Normalisé.....	100
III.Modèle Logique Relationnel Optimisé.....	101

Index des illustrations

Illustration 1: Diagramme de contexte.....	13
Illustration 2: Diagramme de cas d'utilisation.....	19
Illustration 3: Diagramme de séquence (System Level) - Visualiser Informations Coureur.....	28
Illustration 4: Diagramme de séquence (System Level) - Visualiser Classement.....	29
Illustration 5: Diagramme de séquence (System Level) - Créer Équipe.....	30
Illustration 6: Diagramme de séquence (System Level) - Incrire Participant.....	31
Illustration 7: Diagramme de séquence (System Level) - Créer Tour.....	32
Illustration 8: Domain Model.....	34
Illustration 9: Packages de l'application.....	37
Illustration 10: Diagramme de classes.....	39
Illustration 11: Diagramme de séquence - Visualiser informations coureur – Object Level.....	40
Illustration 12: Diagramme de séquence - Visualiser classement – Object Level.....	41
Illustration 13: Diagramme Séquence - Créer équipe - Object Level.....	42
Illustration 14: Diagramme séquence - Incrire Participant - Object Level.....	43
Illustration 15: Diagramme séquence - Créer Tour - Object Level.....	44
Illustration 16: Diagramme d'état-transition - Tour.....	45
Illustration 17: Diagramme d'état-transition - Etape.....	46
Illustration 18: Diagramme d'état-transition - Participant.....	47
Illustration 19: Diagramme d'état transition - Équipe.....	48
Illustration 20: MEA Gestion des inscriptions.....	53
Illustration 21: MEA Gestion de la course.....	54
Illustration 22: MEA Gestion des classements.....	55
Illustration 23: MEA Gestion du plus combatif.....	57
Illustration 24: MEA Gestion contrôles anti-dopage.....	57
Illustration 25: MEA Gestion des utilisateurs.....	58
Illustration 26: MLR Normalisé - Gestion des inscriptions.....	59
Illustration 27: MLR Normalisé - Gestion de la course.....	60
Illustration 28: MLR Normalisé - Gestion du classement.....	61
Illustration 29: MLR Normalisé - Gestion combatif.....	62
Illustration 30: MLR Normalisé - Gestion des utilisateurs.....	63
Illustration 31: MLR Optimisé - Gestion des inscriptions.....	74
Illustration 32: MLR Optimisé - Gestion de la course	75
Illustration 33: MLR Optimisé - Gestion du classement.....	76
Illustration 34: MLR Optimisé - Gestion des utilisateurs.....	77
Illustration 35: MLR Optimisé - Gestion dopage.....	77
Illustration 36: MLR Optimisé - Gestion combatif.....	78
Illustration 37: Architecture technique.....	79
Illustration 38: Cartographie fonctionnelle par module.....	80
Illustration 39: Fonctionnement de TI_PASSER_BEFORE et TI_PASSER_AFTER.....	83
Illustration 40: Liste des packages.....	86
Illustration 41: Extrait interface package DB_INSCRIPTION.....	86
Illustration 42: Détail fonction getPartCrit.....	87
Illustration 43: Extrait interface package DB_COMMUN.....	88
Illustration 44: Extrait interface package DB_RESULTAT.....	88
Illustration 45: Détail fonction getPorteur.....	89
Illustration 46: Détail fonction getEtapeRanking.....	89
Illustration 47: Détail fonction getSelectedTour.....	90

Illustration 48: Squelette page IHM.....	90
Illustration 49: Extrait procédure UI_LEQUIPE.....	91
Illustration 50: Page d'authentification.....	92
Illustration 51: Page d'accueil.....	92
Illustration 52: Page d'édition du tour.....	93
Illustration 53: Requête des étapes d'un tour.....	93
Illustration 54: Page gestion des participants.....	94
Illustration 55: Page gestion des équipes.....	94
Illustration 56: Page gestion des étapes.....	95
Illustration 57: Pages gestion des points de passage d'une étape.....	96
Illustration 58: Modèle complet.....	99
Illustration 59: MLR Normalisé Complet.....	100
Illustration 60: MLR Optimisé Complet.....	101

Historique des modifications

Version	Date	Auteur	Modifications du document
Étape 1	03/03/14	*	Réflexion sur le MCD
	10/03/14	*	Réalisation du MCD
	12/04/14	Clément GHNASSIA	Création du document
	13/04/14	Clément GHNASSIA	Ajout des partie Acteurs et Contexte du projet
	14/04/14	Clément GHNASSIA	Ajout des parties Périmètre et Extensions possibles
	14/04/14	Thomas SIMON	Ajout des parties Contraintes de développement
	14/04/14	Thomas SIMON	Ajout de la partie Flux
	15/04/14	Clément GHNASSIA	Fusion des rapports BD50 et GL52
	16/04/14	Thomas SIMON	Ajout partie définitions et abréviations
	16/04/14	Thomas SIMON	Mise en page du rapport
	17/04/14	Yassine OUAAMOU	Insertion du dictionnaire de données
	17/04/14	*	Insertion des commentaires (MCD et dictionnaire)
	18/04/14	Yassine OUAAMOU	Insertion du diagramme de contexte
	18/04/14	*	Finalisation de la 1ère partie du rapport
	27/04/15	Clément GHNASSIA	Ajout de templates <i>Use Case</i>
	30/04/14	Thomas SIMON	Ajout de templates <i>Use Case</i>
	07/05/14	*	Correction du dictionnaire
	08/05/14	*	Modification des acteurs
Étape 2	12/05/14	*	Réflexion sur optimisation MLR
	16/05/14	*	Réflexion sur optimisation MLR
	21/05/14	Thomas SIMON	Modification Use Case Description
	21/05/14	Thomas SIMON	Génération MLR optimisé
	22/05/14	Thomas SIMON	Ajout de templates <i>Actor</i>
	22/05/14	Clément GHNASSIA	Ajout partie stratégie d'optimisation
	22/05/14	Clément GHNASSIA	Ajout partie optimisations spécifiques à Oracle
	22/05/14	Yassine OUAAMOU	Ajout partie architecture applicative
	22/05/14	*	Finalisation de la 2e partie du rapport
	04/06/14	*	Réalisation des diagrammes de séquences
	07/06/14	Thomas SIMON	Réalisation du Domain Model
	08/06/14	Thomas SIMON et Clément GHNASSIA	Réalisation du diagramme de classe
	09/06/14	Thomas SIMON et Clément GHNASSIA	Réalisation des diagrammes d'état-transition
Etape 3	13/06/14	Clément GHNASSIA	Rédaction de la conclusion

Introduction

Dans le cadre d'une volonté d'amélioration constante de la gestion et la mise à disposition des données relatives au tour de France, les organisateurs de la course ont décidé de mettre en marche un projet destiné à couvrir les besoins actuels des différents acteurs interagissant avec la course, aussi bien pour les membres du tour que pour les journalistes ou encore les spectateurs.

La légendaire course, qui a fêté sa 100ème édition l'an passé, fait l'objet d'un intérêt en perpétuelle croissance. Le développement et la mise en place de ce nouveau système répond donc à des demandes et des besoins toujours plus importants, tant en qualité, qu'en quantité et qu'en scalabilité. Aujourd'hui, le tour de France est de loin la course la plus regardée à travers le monde ; en 2013 elle a attiré plus de 3 milliard de téléspectateurs et plus de 12 millions de spectateurs ont afflué sur les bords des routes.

L'objectif affiché dans la mise en place de cet outil se veut bénéfique tant aux organisateurs pour leur faciliter l'enregistrement des résultats que pour les différents types d'utilisateurs externes à la course cherchant à prendre connaissance d'un certain nombre d'informations comme les résultats sportifs ou les équipes et coureurs engagés.

I. Objectif du document

Le but de ce document est de fournir les spécifications d'un outil de gestion du tour de France. Le système sera développé dans son intégralité et pourra être utilisé aussi bien pour gérer la partie administrative de la course en temps réel, qu'être utilisé afin de fournir des renseignements pendant et après la course à un large public. Il est envisagé que ces renseignements puissent être proposés à des tiers, de manière contractualisée, afin d'amortir le coût de développement et de maintenance du système.

Ce document est tout d'abord destiné aux commanditaires de l'outil, c'est à dire la direction technique du tour de France, et dans une moindre mesure à l'équipe de développement en charge du projet. Il sera utilisé tout au long de la phase de développement afin d'assurer la conformité du logiciel avec les besoins, les contraintes et les choix faits, exprimés à la fois par les futurs utilisateurs et d'une manière plus générale, la plupart des acteurs directement impliqués dans ce projet.

Le développement de ce logiciel constitue un point stratégique et crucial pour l'organisation du tour de France. Il permettra, à terme, d'automatiser un certain nombre de processus, et de gagner en efficacité, en rapidité et en précision. Les différents intéressés pourront aussi disposer de données conséquentes sur la course, ce qui donnera ensuite la possibilité d'établir de nombreuses statistiques. A l'aide de ce logiciel, la légendaire course, qui a célébré sa 100ème édition l'année dernière, pourra proposer un aspect moderne qui se révélera à la hauteur de son prestige.

II. Champ d'application

L'outil développé dans le cadre de ce projet sera principalement un outil de gestion du tour de France. Il permettra d'insérer les données relatives à la course, tout au long du tour, aussi bien avant le tour que pendant son déroulement. Un niveau de granularité assez fin est exigé, permettant non seulement d'insérer des données à l'issu de chaque étape, mais aussi de faire connaître à l'outil une multitude de renseignements relevés au sein même de chaque étape du tour, à différents points et concernant différents types d'informations.

L'outil doit être capable de traiter plusieurs éditions du tour de France. Il doit de ce fait bénéficier d'une certaine généréricité. A priori, aucune modification ne doit être faite sur le logiciel pour gérer cette situation. Étant donné l'investissement qu'il représente, l'outil s'inscrit sur le long terme et doit être opérationnel aussi longtemps que possible, et ce avec un minimum de maintenance.

Enfin, un autre aspect du logiciel est dédié à la visualisation, en temps réel ou non, des données du tour de France. Cette partie doit aussi être capable d'offrir un niveau de détail adapté à la demande de l'utilisateur, de même que d'offrir un certain nombre de statistiques pertinentes concernant la course. Vu le nombre important d'informations susceptibles d'entrer dans le système, les interfaces avec les différents types d'utilisateurs devront être synthétiques et aussi intuitifs que possible.

III. Définitions et abréviations

a. Tour de France¹

- **Maillot jaune** : Il distingue le leader du classement général au temps.
- **Maillot vert** : Il est porté par le leader du classement par points. Les points sont en jeu à l'occasion du sprint intermédiaire et aux arrivées d'étape.
- **Maillot blanc à pois rouges** : Il revient au leader du classement de la montagne. Les points de la montagne sont attribués au passage aux sommets de toutes les difficultés classées.
- **Maillot blanc** : Il récompense le meilleur jeune de 25 ans ou moins du classement général au temps.
- **Prix de la combativité** : Il est remis à l'issue de chaque étape par un jury composé de huit spécialistes de cyclisme. Un " super-combatif " est désigné après la dernière étape du Tour.
- **Classement par équipe** : Il est établi par l'addition des temps des trois meilleurs coureurs de chaque formation sur chaque étape.
- **HC** : Hors Catégorie. Les cols sont classés selon leur dénivelé et le pourcentage de la côte.
- **UCI** : Union Cycliste Internationale. Il s'agit d'une organisation dont le but est développer et de promouvoir le cyclisme en coopération avec les fédérations nationales.²

b. Langage technique

- **UML** : Unified Modeling Language. UML est un langage de modélisation objet. Il permet de modéliser de objets et ainsi représenter une application sous forme de diagramme.³
- **SGBD** : Système de gestion de base de données. Logiciel qui permet de stocker des informations dans une base de données. Un tel système permet de lire, écrire, modifier, trier, transformer ou même imprimer les données qui sont contenus dans la base de données.⁴
- **SQL** : Structured Query Language. Langage informatique normalisé servant à exploiter des

1 Source : <http://www.letour.fr>

2 Source : http://fr.wikipedia.org/wiki/Union_cycliste_internationale

3 Source : <http://fr.openclassrooms.com>

4 Source : <http://sql.sh/sgbdb>

bases de données relationnelles⁵

- **PL/SQL** : Procedural Language / Structured Query Language . Il s'agit d'un langage propriétaire procédural et structuré. Il a été créé par Oracle et est utilisé dans le cadre de bases de données relationnelles. Sa particularité est de pouvoir combiner des requêtes et des instructions procédurales dans le but de créer des traitements complexes.
- **W3C** : World Wide Web Consortium. Il s'agit d'un organisme qui développe des standards pour le Web.
- **HTML** : Hypertext Markup Language. C'est un format de données conçu pour représenter les pages web.⁶
- **CSS** : Cascading Style Sheets. Elles forment un langage informatique qui décrit la présentation des documents HTML.⁷

IV. Organisation du document

Partie	Chapitre	Contenu
Partie 1 Spécifications	Description générale	Description sommaire et synthétique offrant une vision générale du projet et de ses problématiques, sans toutefois entrer dans un niveau de détail trop important.
	Besoins fonctionnels	Détail des besoins fonctionnels exprimés par les commanditaires du logiciel ainsi que l'équipe en charge du développement.
Partie 2 Conception	Architecture	Présentation des différents packages de l'application et leurs dépendances
	Conception détaillée	Présentation des diagrammes UML relatifs à cette partie (diagramme de classe, diagrammes de séquence <i>object-level</i> et diagrammes d'activités)
	Dictionnaire de données	Identification et spécification de toutes les données utilisées par le logiciel et intégrées à la base de données
	Modèle entité association	Modèle entité association découpé en sous-modèle utilisé pour concevoir l'architecture de la base de données
	Modèle logique relationnel	Modèle relationnel issu du modèle entité association, auquel vont être appliquées des techniques de normalisation puis d'optimisation.
	Optimisation spécifique ORACLE	Optimisation à l'aide des outils proposés par le SGBD tel que l'indexation et le partitionnement
	Architecture applicative	Aperçu de l'ensemble de l'application ainsi qu'une présentation différents composants physiques et logiques
Partie 3 Développement		

⁵ Source : http://fr.wikipedia.org/wiki/Structured_Query_Language

⁶ Source : http://fr.wikipedia.org/wiki/Hypertext_Markup_Language

⁷ Source : http://fr.wikipedia.org/wiki/Feuilles_de_style_en_cascade

Partie 1 Spécifications

I. Description générale

1. Domaine étudié

Tout d'abord, rappelons que le tour de France est une course qui sillonne l'hexagone chaque année au mois de juillet pour une période d'environ 25 jours. C'est une course par étapes, qui se comptent généralement entre 20 et 25 par édition. Chaque jour, en dehors des jours des repos, une étape est disputée et un classement est fait par ordre d'arrivée. Le premier cycliste à passer la ligne d'arrivée est le vainqueur de l'étape.

Le tour de France est une course par équipe. En dehors du classement par équipe, chaque coureur est indépendant et gagne individuellement, c'est à dire qu'une victoire d'étape ou une victoire sur le tour est remporté un coureur particulier. Bien évidemment, l'équipe apporte un côté stratégique à la course, puisque c'est elle qui donne aux individualités la possibilité de s'exprimer et de gagner.

En plus de ce classement par étape, le temps cumulé de chacun des coureurs tout au long du tour est établi et permet de définir le vainqueur de la course. Bien entendu, le cycliste qui aura terminé le tour dans une période de temps la plus courte sera désigné vainqueur. C'est le premier virtuel de la course qui porte le maillot jaune.

D'autres classements sont faits afin que des coureurs au profil différents aient un objectif sportif pour participer au tour. On pense au classement pour le maillot à poids (montagne) et le classement pour le maillot vert (sprint) qui sont des classements par point, le classement par équipe, le classement du meilleur jeune et les combattants d'étape ainsi que le super-combattant du tour. Ces deux derniers titres ne sont pas réellement des classements mais désignés par un jury d'experts.

2. Acteurs

Le diagramme suivant donne un aperçu des acteurs interagissant avec le système. Nous avons une vue d'ensemble sur le système évoluant dans son environnement avec une identification de chacun des flux liant le logiciel à ses utilisateurs. Avant de proposer le diagramme, il convient de développer chacun des acteurs et des interactions.

a. Les administrateurs

Les administrateurs du système possèdent un accès très large au système. Ils peuvent renseigner des nouvelles équipes, des nouveaux coureurs, des nouvelles villes ou des nouveaux cols dans la base. C'est eux qui vont également saisir le tracé du futur Tour de France après validation de la Direction.

Bien sûr, ils ne peuvent modifier les résultats et classement de la course en cours ou de celles passées.

b. Le commissaire de course

Il est responsable de la saisie de l'ensemble des informations liées à la course pour laquelle il a été nommé. Ses prérogatives se limitent cependant à la saisie des résultats de la course. Il inscrit met à jour le système avec les résultats des coureurs à la fin de chaque étape. Étant donné son statut de supérieur vis à vis des personnes chargées d'informer le système à chaque point de passage, il peut modifier les saisies de ces derniers.

A priori, le commissaire n'a la possibilité d'éditionner les données que lorsque la course est encore en cours, c'est à dire qu'il ne peut pas agir de façon rétroactive, à moins qu'un événement majeur change les résultats d'une course étant déjà terminée.

c. Les commissaires de point de passage

Les commissaires de point de passage relèvent les informations liées à un point de passage sur une étape, quand les coureurs y parviennent. Leur responsabilité est limitée à entrer les résultats, c'est à dire le temps et le classement de chacun des coureurs aux points de passage pour lesquels ils ont été désignés. Le commissaire de course peut modifier une information saisie par un commissaire, sous réserve de justification.

Encore une fois, le commissaire de point de passage ne dispose que de ses prérogatives en temps réel, et ne peut saisir et modifier les données que lors du passage des coureurs au point de passage. Un commissaire de point de passage peut être responsable de plusieurs de points de passage, à condition que ces derniers n'appartiennent pas à la même étape.

d. Les spécialistes

Les spécialistes forment un jury chargé d'élire le combattant du jour pour chaque étape, ainsi que le super-combattant désigné à l'issue de la course. Le jury reste le même tout au long du tour et chacun des membres est chargé de choisir un coureur. Le coureur ayant récupéré le plus de votes est ainsi désigné combattant du jour (ou super-combattant selon le cas). En cas d'égalité, c'est le commissaire de course qui départage les coureurs.

Il est important de préciser que la désignation du combattant à la fin de chaque étape et la désignation du super-combattant à la fin du tour sont deux choses distinctes. Si le procédé de sélection et le jury restent les mêmes, on ne retrouve aucun lien direct..

e. Les journalistes

Pour les journalistes qui couvrent le tour de France, disposer d'informations nombreuses et les plus exhaustives possibles sur le déroulement de la course est primordial. En effet, les informations recueillies par le système constituent la source d'information la plus importante et sera probablement considérée comme la plus fiable pour ceux-ci. Le système doit être capable de lui fournir les informations en temps réel, c'est à dire pendant la course et pendant le déroulement d'une étape, mais aussi à des fins d'historique, de prospection d'informations, ou d'élaborations de statistiques plus larges que le tour de France, à une période postérieure au déroulement de la course.

Inutile de préciser que les journalistes auront un accès en lecture seule aux informations contenues dans le système. Peut-être que les informations les plus critiques ne seront pas mis à sa disposition dans un premier temps. On pense notamment aux données relatives contrôles antidopage dont la diffusion doit être maîtrisée par les organisateurs de la course, pour des raisons de confidentialité par exemple.

f. Les spectateurs

Un accès pour de simples spectateurs à la recherche de renseignements et ne disposant pas de carte de presse doit aussi être implémenté. Tout comme les journalistes, ils accèdent également aux données contenues dans le système en lecture seule. Peut-être auront-ils un accès à un nombre d'informations restreintes en comparaison des journalistes.

On pourrait poser la question de la pertinence d'un tel acteur dans le cadres des interactions avec notre système, étant donné que le journaliste est censé servir d'intermédiaire entre les informations de la course et le spectateur lambda. Toutefois, pour des raisons de transparence, il a été convenu que chacun était libre d'accéder aux informations à la source, c'est à dire les données présentes dans ce système.

3. Fonctionnement du système

Le fonctionnement de certaines parties du système est à préciser; Le rôle de cette partie consiste à décrire les différents aspects du système. Ces derniers découlent directement de l'analyse des besoins.

a. Inscriptions

On gère l'inscription d'un coureur à un Tour par l'intermédiaire d'un participant. Un participant du Tour est identifié par un numéro de dossard. Les numéros de dossard sont attribués en fonction du classement du tour précédent.

b. Classement

La gestion du classement se fait à l'aide de points de passage. En effet, chaque étape est définie par un certain nombre de points de passage qui permettent de connaître le tracé exact de l'étape. Certains de ces points de passage sont particuliers et compte pour le classement du meilleur grimpeur ou du meilleur sprinteur. On considère la ligne d'arrivée comme un point de passage. Un classement est donc effectué lors de ces points de passage. A la fin de chaque étape, le temps cumulé et le temps de l'étape est relevé pour chaque coureur et équipe. Et chaque porteur de maillot est mis à jour.

c. Attribution des points (montagne et sprint)

Chaque point de passage particulier est associé à une certaine catégorie (col de catégorie 1,2,3,4 ou HC, sprint intermédiaire, sprint final). Chacune de ces catégories possède un barème pour l'attribution des points en fonction des places. Le nombre de places donnant droit à des points peut varier selon les catégories.

d. Prix du plus combatif

Un jury composé de 8 spécialistes est désigné pour l'ensemble du Tour. A chaque étape, les 8 spécialistes votent pour un coureur. Le coureur qui possède le plus de voix est élu « Coureur le plus combatif de l'étape ». A la fin du Tour, les 8 spécialistes votent pour élire le coureur « Super Combatif ».

e. Abandon

Afin d'avoir un traçage complet, lorsqu'un coureur abandonne, on relève le point de passage précédent qui l'il a franchi. On y ajoute alors le motif de l'abandon.

Si un coureur, abandonne en dehors de la course, on relève le dernier point de passage de l'étape précédente qui correspond à l'arrivée.

4. Flux

Cette partie s'intéresse aux interactions que les utilisateurs ont avec le système. A défaut de présenter un niveau de détail très important, il permet d'avoir un aperçu des différents liés entre utilisateurs et logiciel. Bien entendu, les cas d'utilisation seront traitées plus en détail dans une prochaine partie.

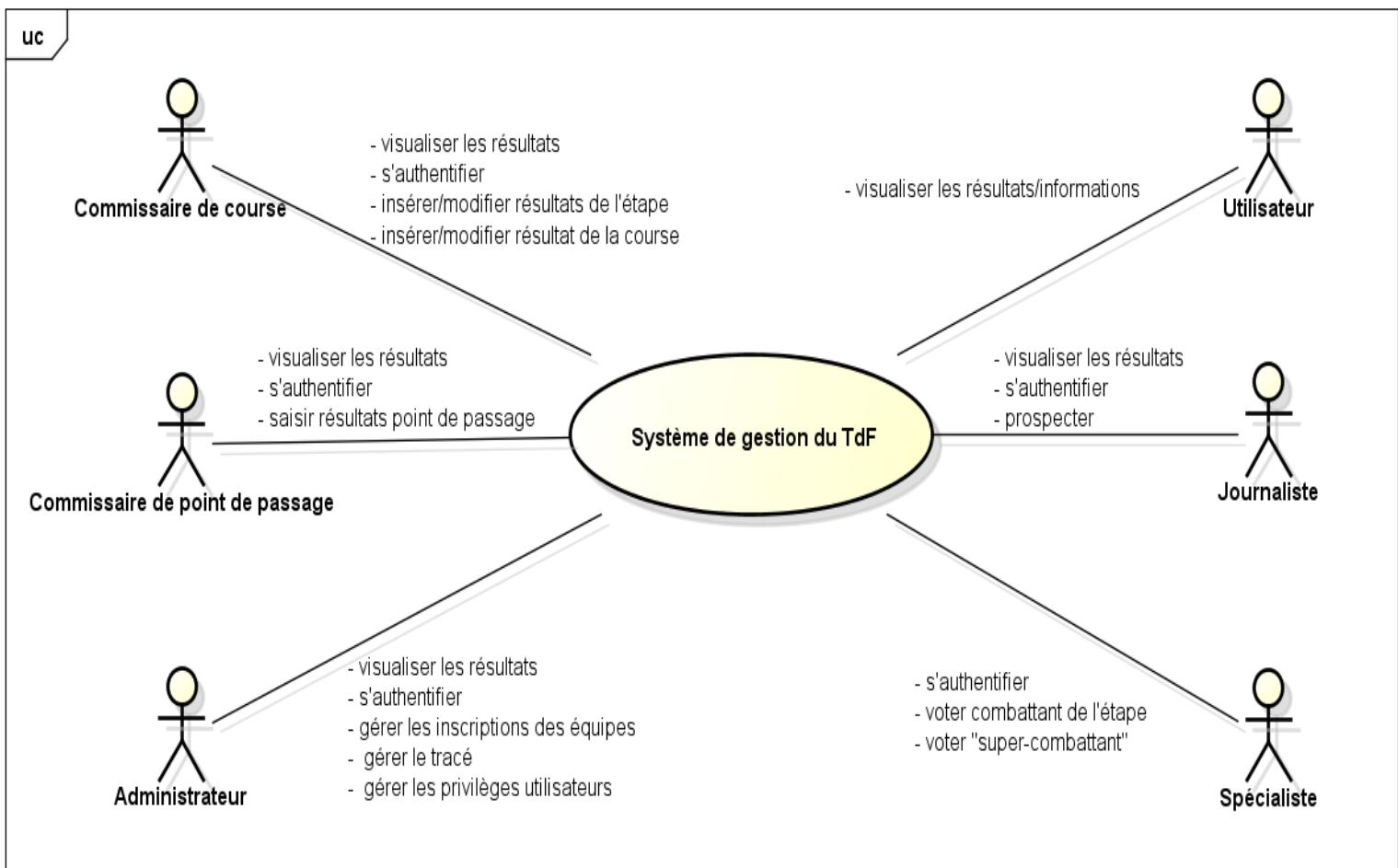


Illustration 1: Diagramme de contexte

powered by Astah

a. Administrateur

L'administrateur a pour responsabilité de gérer l'organisation de la course, c'est à dire de

gérer le tracé des prochains tours, de gérer les inscriptions des joueurs et des équipes, ainsi que d'attribuer les rôles à chacun des acteurs internes tel que le commissaire de la course et les commissaires des point de passage. Toute action entraînant la modifications des informations contenues dans le système nécessite préalablement une authentification.

Fonctionnalités	Description
S'authentifier	
Gérer le tracé	Ajouter et modifier les étapes et les points de passage
Gérer les inscriptions	Ajouter et modifier les équipes et coureurs inscrits pour la course
Gérer les rôles	Attribuer les responsabilités aux utilisateurs (commissaire de course et commissaires de point de passage)

b. Commissaire de course

Le commissaire de course va interagir avec le système pour saisir les résultats de l'étape et contrôler les résultats entrés par les autres acteurs dont il est le supérieur. On pense particulièrement aux commissaires de points de passage. Toute action entraînant la modifications des informations contenues dans le système nécessite préalablement une authentification.

Fonctionnalités	Description
S'authentifier	
Saisir les résultats de l'étape	Saisir le classement de l'étape, valider les classements généraux, ajouter un abandon éventuel.
Gérer les données existantes	Modifier les informations saisies par un commissaire de point de passage.

c. Commissaire de point de passage

Cet acteur va utiliser le système pour enregistrer les résultats liés au point de passage pour lequel il été désigné. Toute action entraînant la modification des informations contenues dans le système nécessite préalablement une authentification.

Fonctionnalités	Description
S'authentifier	
Saisir les résultats du point de passage	Saisir le classement et le temps de chaque coureur arrivant au point de passage.

d. Spécialiste

Le spécialiste compose le jury dans le cadre de la sélection du combattant du jour à l'issue de chaque étape, ainsi que la sélection du super-combattant désigné à la fin de la course. Bien entendu, chacun des spécialistes doit s'authentifier avant de saisir les coureurs qu'il a désigné (comme combattant du jour ou super-combattant).

Fonctionnalités	Description
S'authentifier	
Voter pour le combattant du jour	Voter pour le combattant de l'étape.

Fonctionnalités	Description
Voter pour le super-combattant	Voter pour le super-combattant de la course à la fin du tour.

e. Journaliste

Le journaliste peut accéder au système d'information pour récolter un certain nombre de renseignements liés à la course. Sa carte de presse lui confère un accès à un plus grand nombre d'informations qu'un utilisateur standard. Toute connexion au système nécessite une authentification au préalable.

Fonctionnalités	Description
S'authentifier	
Récolter des informations	Récupérer différents renseignement (coureurs, résultats étape, ...) liées au tour. Il accède au système en lecture seule.

d. Utilisateur

L'utilisateur n'a accès qu'à un nombre restreints d'information. Étant donné les faibles droits de l'utilisateur sur le système, aucune authentification n'est nécessaire.

Par un concept d'héritage qui n'apparaît pas dans cette section, il est important de considérer chacun des acteurs comme une spécialisation de l'utilisateur. Ainsi, tous les acteurs possèdent au minimum les droits de l'utilisateur.

Fonctionnalités	Description
Visualiser les résultats	Visualise les résultats de la course. Il accède au système en lecture seule.

5. Contraintes de développement

Le développement de l'application est soumis à certaines contraintes. Dans la phase de spécification le langage de modélisation utilisé doit être UML 2.0. Il s'agit non seulement d'un langage graphique mais de nombreuses descriptions afin de mieux comprend le fonctionnement du système seront nécessaires. Nous utiliserons donc des diagrammes d'utilisations, de séquences et de classes.

Pour la phase de conception, la méthode d'analyse qui doit être utilisée est MERISE. La structure du système d'information sera représentée à l'aide d'un Modèle Conceptuel de Données. La structure de données sera décrite grâce à un Modèle Logique de Données.

En ce qui concerne le SGBD, Oracle Database de l'entreprise Oracle nous a été imposé. Le choix de la version se fera entre 10g ou 11g. Bien que l'outil soit multi-plateforme, on privilégiera un système d'exploitation de type Windows a des fins de simplicité d'implémentation et de compatibilité.

L'exploitation de la base de données s'effectuera avec le langage PL/SQL. Il s'agit d'un langage propriétaire procédural et structuré. Il a été créé par Oracle et est utilisé dans le cadre de bases de données relationnelles. Sa particularité est de pouvoir combiner des requêtes et des instructions procédurales dans le but de créer des traitements complexes.

L'application doit être livrée sous forme de site Web. Le site respectera les normes imposées par le W3C. Il s'agit d'un organisme qui développe des standards pour le Web. Les langages utilisés

seront donc HTML, CSS et JavaScript.

Pour la phase de création, l'environnement de développement nécessaire est Oracle Application Express également appelé ApEx. Il permet un développement rapide d'applications Web utilisant une base de données Oracle.

6. Hypothèses et dépendances

La bonne réalisation du projet nécessite la mise en place d'un certains nombre de postulats. Si ces postulats venaient à être remis en question, il serait nécessaire de modifier, voir re-développer le logiciel. Plutôt que de proposer de réelles solutions à ce manque adaptabilité, il convient d'identifier les hypothèses et dépendances inhérentes au projet.

a. Modification de l'organisation de la course

Il est évident que si l'organisation structurelle du tour de France venait à changer de manière conséquente, il est fort probable que le système ne soit plus adapté et ne puisse plus être utilisé, du moins pas sans une phase d'analyse et de développement ou de refonte logicielle. Même s'il est peu probable que l'organisation du tour soit modifiée de manière importante, il faut tout de même l'envisager.

b. Modification du règlement de la course

Il peut arriver que le règlement de la course change. Cela est déjà arrivé dans le passé, notamment sur la manière dont sont établis les classements. Si cette modification est minime, peut-être faudra-t-il modifier légèrement le système. En revanche, si cette modification est majeure, elle pourrait remettre en cause le fonctionnement de l'ensemble du logiciel.

c. Modification du périmètre du système

Le périmètre a été défini afin de majorer les phase de conception et de développement associées à ce projet. Une remise en question de ce périmètre introduite par de nouveaux besoins ou/et de nouvelles contraintes aurait indubitablement un impact sur le logiciel. L'extension de périmètre alors que l'outil est déployé et fonctionnel nécessiterait non seulement une phase de développement supplémentaire, mais probablement une phase de conception associée et qui n'est pas à négliger.

d. Dépendances logicielles

Durant son développement et sa mise en place, le système va s'appuyer sur différents outils logiciels. Il est nécessaire de se questionner à propos de l'impact d'un changement de technologie sur le logiciel.

PL/SQL, qui est va être utilisé pour le développement du projet, est un langage propriétaire disponible uniquement sur les systèmes de gestion de base de données (SGBD) d'Oracle. Même si la plupart de ses concurrents proposent des technologies similaires, un changement de système de gestion de base de données nécessitera nécessairement une modification du système, donc une phase de développement pour adapter ce dernier.

e. Dépendances applicatives

De même que pour les dépendances logicielles, les dépendances applicatives sont amenées par l'utilisation d'outils de développement propriétaires et disponibles uniquement dans le cadre d'un interfaçage avec les systèmes Oracle. L'outil ApEx qui est utilisé dans le cadre de la création d'applications utilisant un système de gestion de base de données (SGBD) Oracle n'est pas portable sur les outils proposés par ses concurrents. Toute la partie applicative serait donc à réévaluer si un changement venait à être souhaité ou contraint.

7. Périmètre du système

Nous allons maintenant définir ce qui va être pris en compte dans le cadre de ce projet. Il convient de fixer les frontières du futur système dès maintenant pour s'assurer de sa conformité avec les besoins et les contraintes exprimées par les différents acteurs. Comme nous l'avons expliqué précédemment, on considère deux grandes parties dans le système. Ces parties seront utilisés par différents types d'acteurs, suivant leur rôle et leurs fonctions dans la course.

La première grande partie du logiciel est liée à la gestion de la course. Elle est utilisée par les organisateurs pour la création de la course et par commissaires qui saisissent les résultats au sein du système. Bien évidemment, chaque type d'acteur aura des priviléges différents suivant ses responsabilités et des prérogatives dans le cadre de l'événement. Chacun de ces utilisateurs aura un niveau d'accès personnalisé, évitant ainsi certaines manipulations, entraînant des informations erronées ou non cohérentes.

Une deuxième partie concernera la visualisation des différents détails de la course aux acteurs n'ayant accès au système que dans un but informatif, et n'ayant aucune responsabilité quant à la gestion en tant que telle de la course. Ces utilisateurs sont des personnes étrangères à l'organisation technique du tour de France, tel que les spectateurs souhaitant recueillir des informations dans un cadre strictement privé et non commercial. Les journalistes auront aussi un accès en lecture seule, mais avec pour objectif de les transmettre à un large public.

Ces deux parties porteront uniquement sur un Tour de France et plus précisément celui de l'année 2012. De ce fait, les modifications de point de règlement d'un Tour à l'autre ne sont pas gérées.

Une sous-partie du système permet également de gérer les contrôles anti-dopage. Cette gestion se limite à la saisie d'un contrôle à une date donnée pour un coureur. Le résultat de ce contrôle peut aussi être saisi.

8. Extensions

Les extensions possibles apportées ci-dessous ne sont présentes qu'à titre informatif. Par définition, elle ne seront pas implémentées et n'influeront aucunement sur le développement du projet. Bien que jugées pertinentes, elles dépassent amplement les limites qui ont été fixées pour le développement de ce système.

Il aurait été possible d'interfacer l'application avec des systèmes extérieurs dans le but de faciliter la saisie et d'automatiser un certain nombre de processus, réduisant par la même le risque d'erreurs et d'insertion de mauvaises données. On pense par exemple à des systèmes de bases de données géographique utiles pour automatiser la saisie des latitudes, longitudes et altitudes nécessaires à notre système. On aurait aussi pu intégrer un module permettant de récupérer automatiquement les informations météorologiques qui peuvent influer sur le déroulement de la

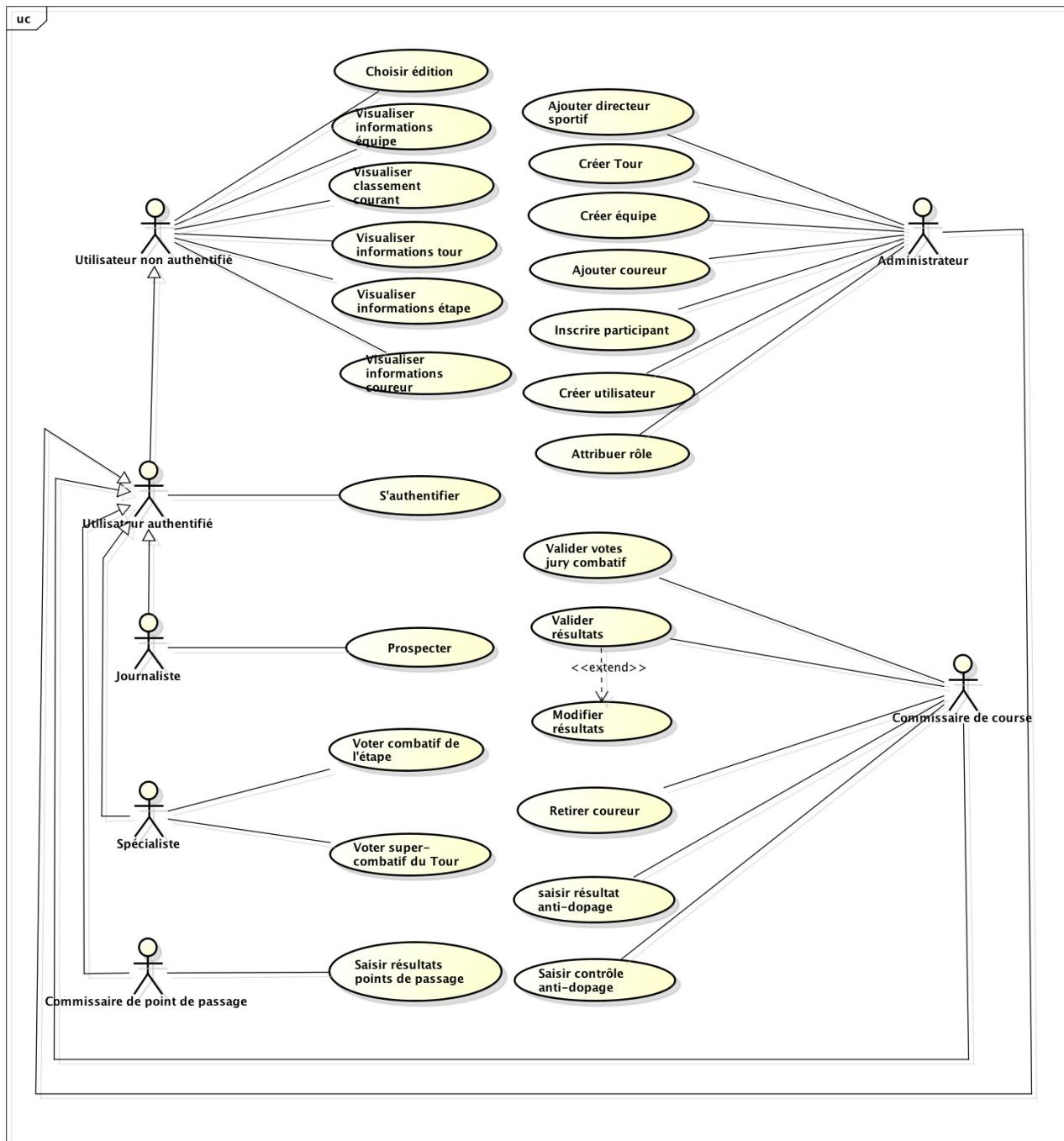
course ainsi que les résultats. Une autre possibilité aurait été d'ajouter un module qui aurait pour fonction d'envoyer des informations à un système de l'UCI tel que les résultats de la course ou les contrôles anti-dopage.

Afin de ne pas compliquer le système, il a été décidé de n'implémenter le système que pour le tour de France. On aurait pu envisager un plus haut niveau de généricité pour que le système soit utilisé dans le cadre d'autres courses cyclistes par étape (tour d'Espagne, tour d'Italie, ...) dont les règles et le mode de fonctionnement sont très proches. On aurait pu aller encore plus loin en proposant un système compatible avec d'autres courses cyclistes au fonctionnement totalement différent (les courses d'un jour par exemple).

Pour la gestion interne du Tour, il aurait été possible d'ajouter une gestion de la caravane. Cette sous-partie permettrait de gérer les partenaires présents dans la caravane, gérer l'ordre de passage des chars ainsi que les horaires. On pourrait également ajouter un grand nombre d'informations portant sur l'aspect « touristique ». Le public pourrait alors en savoir plus sur les villes et départements traversée comme un historique, les monuments à visiter ou alors des hôtels recommandés pour suivre le Tour à travers la France. La gestion des contrôles anti-dopage peut être affinée avec des informations sur les types de contrôles effectués, sur le lieu, sur le laboratoire effectuant les analyses ou sur la planification des contrôles pour le vainqueur d'étape et le maillot jaune. Avec une extension du système permettant de gérer plusieurs Tours, il faut pouvoir gérer les points de règlements qui peuvent changer d'une édition à l'autre. Enfin, une partie peut être dédiée à la gestion de l'organisation du Tour de France. Cette sous-partie offrirait la possibilité au équipe techniques d'obtenir des informations sur leur planning, leur logement tout au long du parcours.

II. Besoins fonctionnels

1. Diagramme des cas d'utilisation



powered by Astah

Illustration 2: Diagramme de cas d'utilisation

a. Description des acteurs

Actor description template	
Identification/	AC01
Itération	1
Name	Utilisateur non-authentifié
Stereotype	<<Human Being>>
Type	Abstract
Form	Initiator, Receiver
Responsabilities	(initiator, receiver, UC01 : Visualiser infirmations équipe) (initiator, receiver, UC02 : Visualiser classement courant) (initiator, receiver, UC03 : Visualiser infirmations Tour) (initiator, receiver, UC04 : Visualiser informations étapes) (initiator, receiver, UC05 : Visualiser infirmations coureur)
Description	Acteur souhaitant consulter les informations sur le tour et les résultats à titre informatif. Il n'a pas besoin de s'authentifier.
Relationships	(Child, AC02 : Utilisateur authentifié)
History	

Actor description template	
Identification/	AC02
Itération	1
Name	Utilisateur authentifié
Stereotype	<<Human Being>>
Type	Abstract
Form	Initiator, Sender, Receiver
Responsabilities	(initiator, sender, UC06 : S'authentifier)
Description	Acteur nécessitant une authentification pour accéder au système et ayant des priviléges correspondant à son profil
Relationships	(Parent, AC01 : Utilisateur non-authentifié) (Child, AC03 : Journaliste) (Child, AC04 : Spécialiste) (Child, AC05 : Commissaire de point de passage) (Child, AC06 : Commissaire de course) (Child, AC07 : Administrateur)
History	

Actor description template	
Identification/	AC03
Itération	1
Name	Journaliste
Stereotype	<<Human beeing>>

Actor description template	
Type	Concrete
Form	Initiator, Sender, Receiver
Responsabilities	(initiator,receiver, UC07 : Prospecter)
Description	Acteur souhaitant prospector et collecter une grande quantité d'informations. Il nécessite une authentification pour avoir un niveau de détail des informations plus élevé ou pour extraire des données.
Relationships	(Parent : AC02, Utilisateur authentifié)
History	

Actor description template	
Identification/	AC04
Itération	1
Name	Spécialiste
Stereotype	<<Human beeing>>
Type	Concrete
Form	Initiator, Sender, Receiver
Responsabilities	(initiator, sender, UC08 : Voter combatif de l'étape) (initiator, sender, UC09 : Voter super-combatif du Tour)
Description	Personne membre d'un jury et qui vote à la fin de chaque étape pour le coureur le plus combatif. A la fin du Tour, la personne vote également pour le coureur super-combatif.
Relationships	(Child, AC02 : Utilisateur authentifié)
History	

Actor description template	
Identification/	AC05
Itération	1
Name	Commissaire de point de passage
Stereotype	<<Human beeing>>
Type	Concrete
Form	Initiator, Sender, Receiver
Responsabilities	(initiator, sender, UC10 : Saisir résultats points de passage)
Description	Acteur se trouvant à un point de passage d'une étape. Il est habilité par l'organisation du Tour de France pour saisir le classement des coureurs pour ce point de passage. Il a besoin d'une authentification pour accéder au système.
Relationships	(Child, AC02 : Utilisateur authentifié)
History	

Actor description template	
Identification/	AC06
Itération	1
Name	Commissaire de course
Stereotype	<<Human beeing>>
Type	Concrete
Form	Initiator, Sender, Receiver
Responsabilities	(initiator, sender, UC11 : Saisir contrôle anti-dopage) (initiator, sender, UC12 : Saisir résultat contrôle anti-dopage) (initiator, sender, UC13 : Retirer coureur) (initiator, sender, UC14 : Valider résultats) (initiator, sender, UC15 : Modifier résultats) (initiator, sender, UC16 : Valider vote jury combatif)
Description	Acteur habilité par l'organisation du Tour de France pour gérer l'évolution de la course (classement/contrôle anti-dopage/abandon). Il a besoin d'une authentification pour accéder au système.
Relationships	(Child, AC02 : Utilisateur authentifié)
History	

Actor description template	
Identification/	AC07
Itération	1
Name	Administrateur
Stereotype	<<Human Being>>
Type	Concrete
Form	Initiator, Sender, Receiver
Responsabilities	(initiator, sender, UC17 : Attribuer rôle) (initiator, sender, UC18 : Créer utilisateur) (initiator, sender, UC19 : Incrire participant) (initiator, sender, UC20 : Ajouter coureur) (initiator, sender, UC21 : Créer équipe) (initiator, sender, UC22 : Créer Tour) (initiator, sender, UC23 : Ajouter directeur sportif)
Description	Acteur en charge de l'administration de la course et attribution des rôles : création utilisateur/droits, saisie du parcours et création d'étape, ajout de coureurs et équipes.. Il a besoin d'une authentification pour accéder au système.
Relationships	(Child, AC02 : Utilisateur authentifié)
History	

b. Description des cas d'utilisation

Use case description template	
Identification	UC01
Itération	1
Nom	Visualiser informations coureur
Type	Concrete
Description	Consultation des informations détaillées sur un coureur
Acteurs	Utilisateur non-authentifié
Pré-conditions	<ul style="list-style-type: none"> - Une édition est sélectionnée - Le coureur et l'équipe existent, U est l'utilisateur non-authentifié et affichage de la page d'accueil du système.
Post-conditions	
Flux	<ol style="list-style-type: none"> 1. U clique sur l'onglet « Coureurs » 2. Le système affiche la liste des coureurs participant à l'édition considérée (numéro de dossard, nom, prénom, pays, nom d'équipe), un champ de recherche sur le nom et un champ de recherche sur le prénom. <ol style="list-style-type: none"> 2. a [L'utilisateur utilise la liste] <ol style="list-style-type: none"> 2. a. 1. U clique sur un coureur présent dans la liste 2. a. 2. Le système affiche : numéro de dossard, nom, prénom, nom de l'équipe, la nationalité, le poids, la taille, le temps général, le classement général, les points et le classement pour l'obtention du maillot de meilleur grimpeur, les points et le classement pour l'obtention du maillot vert, le classement pour le maillot blanc si le coureur est en lice pour le classement considéré, s'il a déjà reçu le prix du plus combatif ou du super-combatif, ainsi qu'un historique des ces résultats sur les autres éditions. 2. b. [U utilise le champ de recherche] <ol style="list-style-type: none"> 2. b. 1. U saisit le nom et/ou le prénom 2. b. 2. U clique sur « Rechercher » <ol style="list-style-type: none"> 2. b. 2. a. [Aucun coureur n'est trouvé] <ol style="list-style-type: none"> 2. b. 2. a. 1. Le système renvoie sur la liste complète comme 2. avec un message d'erreur « Aucun coureur trouvé » 2. b. 2. b. [Un coureur est trouvé] <ol style="list-style-type: none"> 2. b. 2. b. 1. Le système affiche les détails du coureur (cf 2.a.2.) 2. b. 2. c. [Plusieurs coureurs sont trouvés] <ol style="list-style-type: none"> 2. b. 2. c. 1. Le système affiche la liste des coureurs correspondants selon le format décrit dans 2. 2. b. 2. c. 2. L'utilisateur clique sur un coureur de la liste (ou refait une recherche comme en 2.b.) 2. b. 2. c. 3. Le système affiche les détails du coureur (cf 2.a.2.)

Use case description template	
Identification	UC02
Itération	1
Nom	Visualiser classements courants
Type	Concrete
Description	Consultation des différents classements généraux (maillot jaune, maillot vert, maillot à pois, maillot blanc, classement par équipe)
Acteurs	Utilisateur non-authentifié
Pré-conditions	<ul style="list-style-type: none"> - Une édition est sélectionnée - U est l'utilisateur non-authentifié et affichage de la page d'accueil du système.
Post-conditions	
Flux	<ol style="list-style-type: none"> 1. U clique sur l'onglet « Classement » 2. Le système affiche les 10 premiers de chaque classement : <ul style="list-style-type: none"> - le rang, numéro de dossard, nom et prénom du participant, le nom de l'équipe, la position dans le classement, le temps ou le nombre de points associé au classement et l'écart avec le leader - pour chaque classement un lien vers une page contenant le classement complet. 2. a. [U clique sur une ligne concernant un participant] <ul style="list-style-type: none"> 2. a. 1. Le système affiche la page relative aux détails du participant 2. b. [U clique sur lien vers un classement détaillé] <ul style="list-style-type: none"> 2. b. 1. Le système affiche le classement complet.

Use case description template	
Identification	UC03
Itération	1
Nom	Créer équipe
Type	Concrete
Description	Ajout des informations d'une équipe inscrite à un Tour.
Acteurs	Administrateur
Pré-conditions	<ul style="list-style-type: none"> - Une édition est sélectionnée - Le Tour existe et les directeurs sportifs existent, A est l'administrateur et affichage de la page d'administration du système.
Post-conditions	
Flux	<ol style="list-style-type: none"> 1. A clique sur « Ajouter équipe » 2. Le système affiche un formulaire pour saisir les informations de l'équipe pour le tour considéré : <ul style="list-style-type: none"> - un champ pour saisir le nom de l'équipe - un champ pour saisir une description de l'équipe - une liste déroulante contenant les pays - un champ pour saisir le nom du sponsor - un champ pour saisir l'acronyme du sponsor - un champ pour saisir le lien du site web

Use case description template	
	<ul style="list-style-type: none"> - une liste déroulante contenant les noms et prénoms des directeurs sportifs <p>3. A remplit les champs.</p> <p>4. A clique sur « Valider »</p> <p>4. a. [Les valeurs sont correctes]</p> <p>4. a. 1. Le système ajoute les données dans la base</p> <p>4. b. [Au moins une valeur est erronée]</p> <p>4. b. 1. Le système revient au formulaire 2. en affichant un message d'erreur.</p>

Use case description template	
Identification	UC04
Itération	1
Nom	Inscrire participant
Type	Concrete
Description	Inscrire un nouveau participant pour un Tour
Acteurs	Administrateur
Pré-conditions	<ul style="list-style-type: none"> - Une édition a été sélectionnée - Le cycliste existe, l'équipe du participant existe, A est l'administrateur et affichage de la page d'administration du système.
Post-conditions	
Flux	<ol style="list-style-type: none"> 1. A clique sur « Ajouter participant» 2. Le système affiche un formulaire pour saisir les informations du participant pour le tour considéré : <ul style="list-style-type: none"> - une liste déroulante contenant la liste du nom des équipes inscrites pour le Tour - une liste contenant la liste des noms et prénoms des cyclistes - une liste déroulante contenant les pays - un champ pour saisir la taille du participant - un champ pour saisir le poids du participant 3. A sélectionne les valeurs ou remplit les champs vides 4. A clique sur « Valider » <p>4. a. [Les valeurs sont correctes]</p> <p>4. a. 1. Le système insère les données dans la base</p> <p>4. b. [Au moins une valeur est erronée]</p> <p>4. b. 1. Le système revient au formulaire 2. en affichant un message d'erreur.</p>

Use case description template	
Identification	UC05
Itération	1
Nom	Créer Tour
Type	Concrete
Description	Création d'une nouvelle édition du Tour de France

Use case description template	
Acteurs	Administrateur
Pré-conditions	A est l'administrateur et affichage de la page d'administration du système. Les villes des étapes et point de passage existent.
Flux	<p>1. A clique sur « Créer nouvelle édition du Tour»</p> <p>2. Le système affiche un formulaire pour saisir les informations relatives à l'édition</p> <ul style="list-style-type: none"> - une liste déroulante avec les années - un champ pour saisir le numéro de l'édition - un champ pour saisir la date de début - un champ pour saisir la date de fin - un champ pour saisir le nom de l'édition <p>3. A clique sur le bouton « Valider »</p> <p>3. a. [Les données sont correctes]</p> <ul style="list-style-type: none"> 3.a.1 Le système intègre les données dans la base [...] <p>3. b. [Au moins une donnée est erronée]</p> <ul style="list-style-type: none"> 3. b. 1. Le système revient sur 2. tout en affichant un message d'erreur. <p>3. a. 2. A clique sur « Ajouter une étape »</p> <p>3. a. 3. Le système affiche un formulaire pour saisir les informations relatives à l'étape</p> <ul style="list-style-type: none"> - le numéro de l'étape est pré saisi et non modifiable - le nom de l'étape - la date de l'étape - le type de l'étape <p>3. a. 4. A clique sur « Valider »</p> <p>3. a. 4. a. [Données correctes]</p> <ul style="list-style-type: none"> 3. a. 4. a. 1. Le système intègre les données dans la base [...] <p>3. a. 4. b. [Au moins une donnée est erronée]</p> <ul style="list-style-type: none"> 3. a. 4. b. 1. Le système revient sur 3. a. 3. tout en affichant un message d'erreur. <p>3. a. 4. a. 2. Le système revient sur la page de l'édition du Tour nouvellement ajouté avec la liste des étapes déjà créées</p> <p>3. a. 4. a. 3. A clique sur le nom d'une étape</p> <p>3. a. 4. a. 4. Le système affiche les détails de l'étape avec éventuellement les points de passages déjà créés.</p> <p>3. a. 4. a. 5. A clique sur « Ajouter un point de passage »</p> <p>3. a. 4. a. 6. Le système affiche un formulaire permettant de saisir les informations relatives au point de passage pour l'étape considérée</p> <ul style="list-style-type: none"> - le numéro du point de passage est pré-saisi - le nom du point de passage - le nombre de kilomètre depuis le départ - l'altitude - l'horaire de passage prévu - le nom de la ville dans une liste de sélection - la catégorie du point de passage dans une liste de sélection <p>3. a. 4. a. 7. A clique sur « Valider »</p> <p>3. a. 4. a. 7. a. [Données correctes]</p>

Use case description template	
	<p>3. a. 4. a. 7. a. 1. Le système intègre les données dans la base. [...] 3. a. 4. a. 7. b. [Au moins une donnée est erronée] 3. a. 4. a. 7. b. 1. Le système revient sur 3. a. 4. a. 4. tout en affichant un message d'erreur</p> <p>3. a. 4. a. 7. a. 1. Le système affiche les détails de l'étape avec le ou les points de passages déjà créés comme en</p>

2. Diagrammes de séquence

a. Visualiser informations coureurs

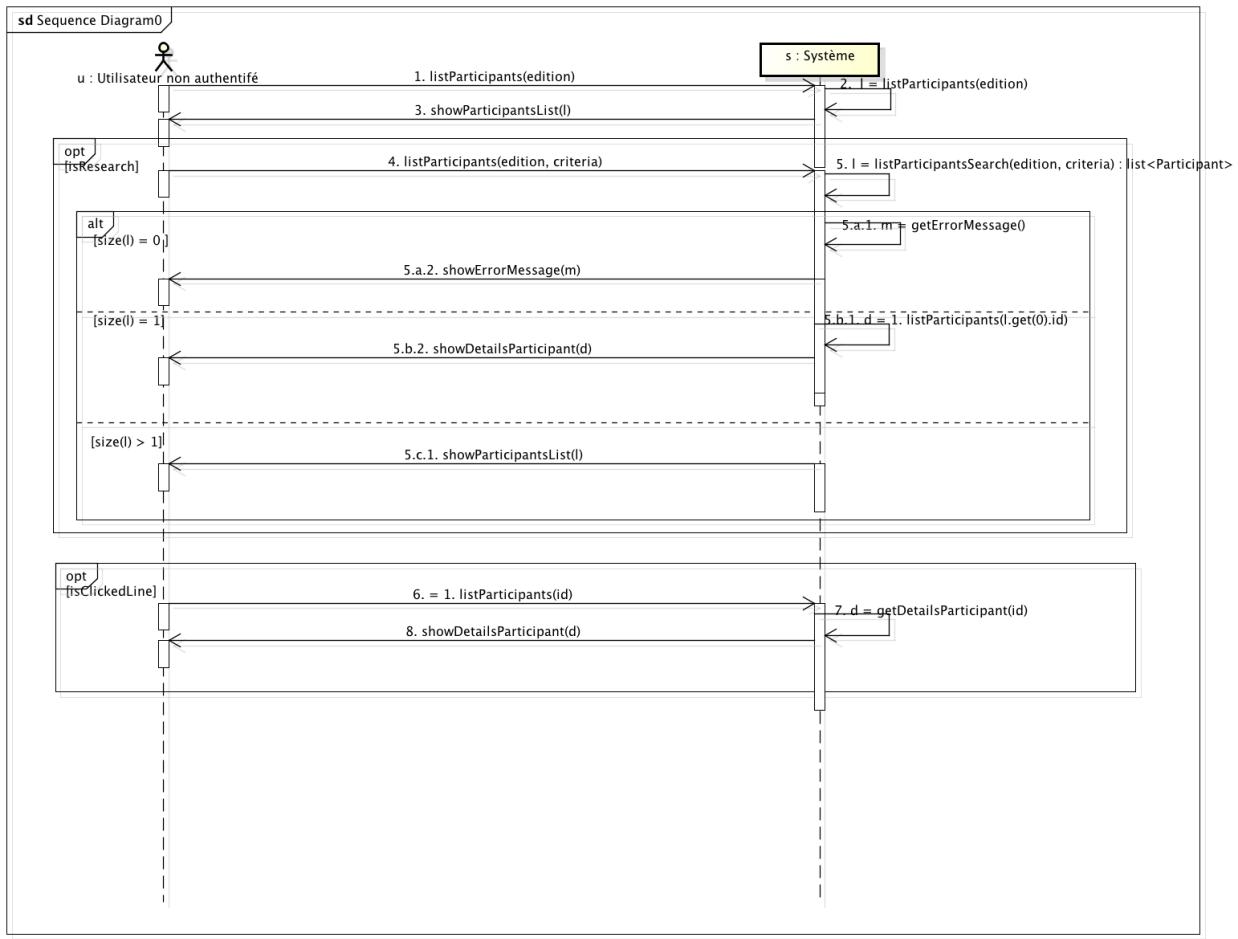


Illustration 3: Diagramme de séquence (System Level) - Visualiser Informations Coureur

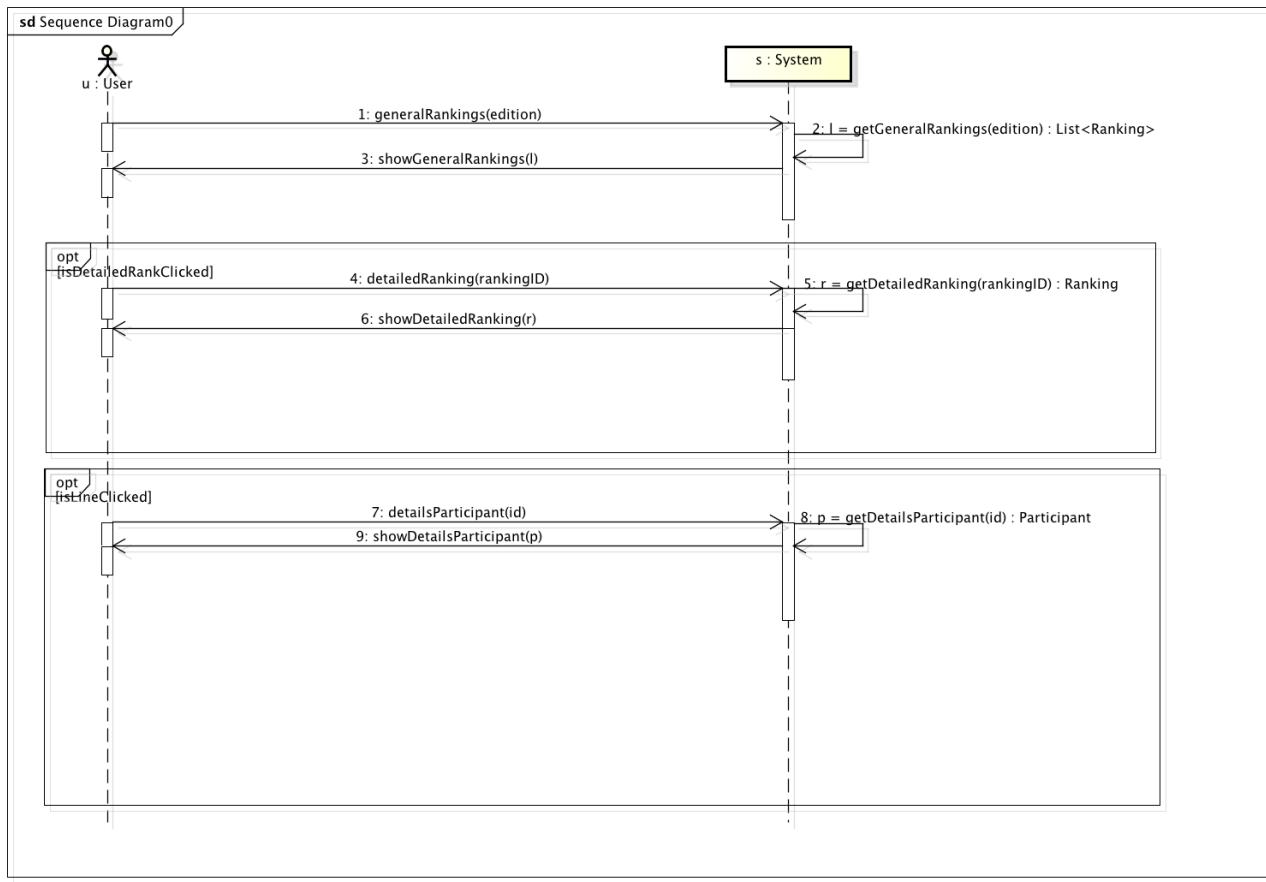
Lorsque l'utilisateur souhaite consulter les informations relatives aux coureurs engagés dans une course, il commence par afficher la liste de tous les participants. S'il souhaite obtenir des informations plus détaillées que celles contenues dans la liste, deux choix s'offrent à lui :

- Il peut cliquer directement sur un coureur dans la liste de départ, auquel cas une page affichant les détails spécifiques au coureur est renvoyée
- Il peut aussi utiliser le formulaire de recherche présent au dessus de la liste. Lorsque l'utilisateur exécute sa recherche, le système réagit différemment selon les résultats correspondant à ces critères. En effet, si la recherche ne produit aucun résultat (la liste des participants correspondants est vide), le système renvoie un message d'erreur ainsi que la page de départ, c'est à dire la liste de tous les coureurs inscrits au tour. Si le nombre de coureurs correspondants aux critères de recherche est égal à exactement un, alors le système redirige l'utilisateur la page affichant les détails spécifiques à ce coureur. Enfin, si les critères de recherches correspondent à plusieurs coureurs, alors le système renvoie logiquement une liste filtrée, à partir de laquelle on pourra également sélectionner un

coureur (en cliquant sur la ligne correspondante).

À l'exception de la page relative aux détails d'un coureur, l'utilisation du formulaire de recherche est possible, tant pour affiner une recherche précédente que pour faire une recherche différente. De même des qu'une liste de coureurs est retournée, l'utilisateur peut, s'il le souhaite consulter les détails relatifs à un participant en cliquant que la ligne correspondante.

b. Visualiser classement



powered by Astah

Illustration 4: Diagramme de séquence (System Level) - Visualiser Classement

Lorsque l'utilisateur souhaite consulter les différents classements relatifs au tour, le système affiche un bref récapitulatif des différents classements (maillot jaune, montagne, sprint, meilleur jeune). Seul les cinq premiers de chaque classement sont affichés afin de ne pas inonder l'utilisateur avec un trop grand nombre d'informations.

L'utilisateur peut éventuellement choisir d'utiliser le lien en dessous de chacun des classements afin que le système lui retourne une page dédié au classement sélectionné. Dans ce cas, la totalité du classement considéré est affiché.

Comme pour le diagramme de séquence précédent, l'utilisateur peut choisir à tout moment de consulter les détails relatifs à un utilisateur en cliquant sur la ligne correspondante. L'utilisateur sera ensuite redirigé vers la page présentant de façon détaillée un participant.

c. Créer équipe

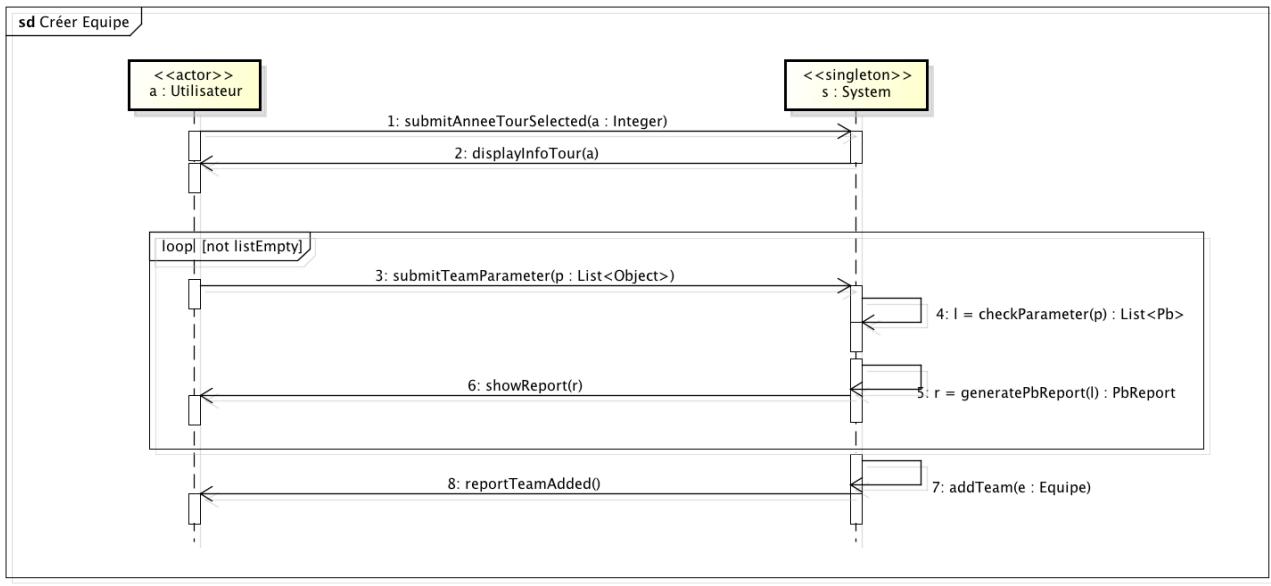


Illustration 5: Diagramme de séquence (System Level) - Crée Équipe

powered by Astah

Après avoir sélectionné l'année du Tour que l'administrateur souhaite modifier, un formulaire d'ajout d'équipe pour ce Tour s'affiche. Il suffit d'envoyer au système l'année du Tour à l'aide d'un entier.

L'administrateur aura donc la charge de saisir des données dans différents champs tel que le nom de l'équipe ou le nom du directeur sportif par exemple.
Les données saisies seront donc que des chaînes de caractères ou des numéros. Il se s'agira pas d'objets complexes.

Ces données envoyées sous forme de liste au système sont vérifiées c'est à dire que le système s'assure de l'intégrité des données. En effet, le format d'un site web soit être conforme, le pays de l'équipe doit exister ou le directeur sportif saisi existe.

Le système stocke les erreurs ou problèmes rencontrés lors de cette vérification. Il renvoie un rapport d'erreur pour informer l'utilisateur des modifications à apporter. Quand il n'y a plus d'erreurs le système crée l'équipe et indique à l'utilisateur que l'ajout s'est bien passé.

d. Incrire Participant

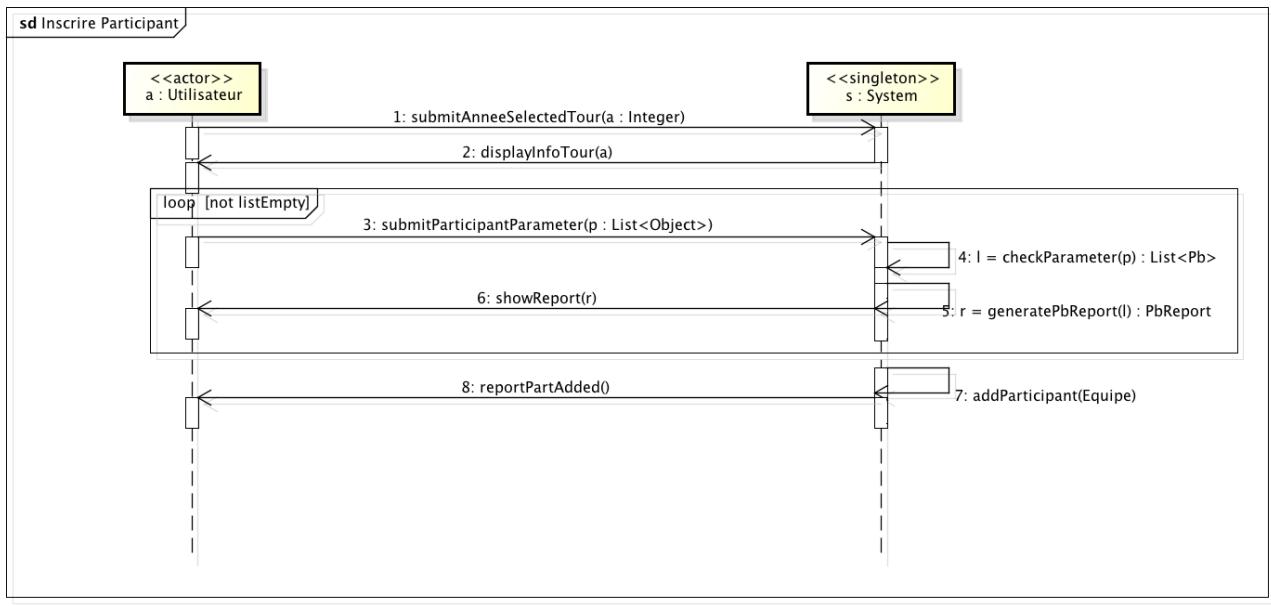


Illustration 6: Diagramme de séquence (System Level) - Incrire Participant

powered by Astah

Après avoir sélectionné l'année du Tour que l'administrateur souhaite modifier, un formulaire d'ajout de participant pour ce Tour s'affiche. Il suffit d'envoyer au système l'année du Tour à l'aide d'un entier.

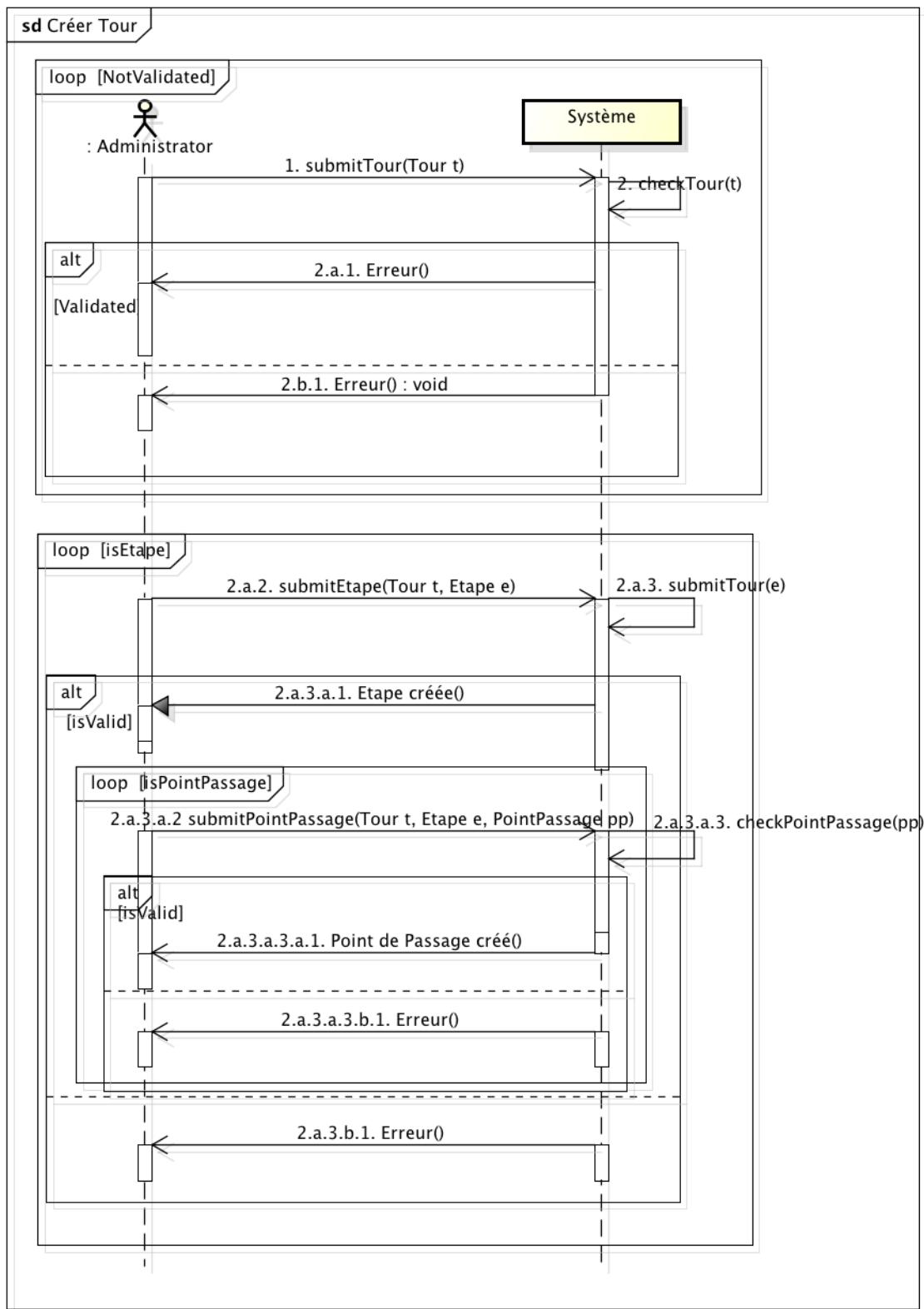
L'administrateur aura donc la charge de saisir des données dans différents champs tel que le cycliste associé à ce participant, l'équipe à laquelle il appartient ou son poids.

Ces données envoyées sous forme de liste au système sont vérifiées c'est à dire que le système s'assure de l'intégrité des données. En effet, le cycliste doit exister ou le poids ne peut excéder 200 kilos par exemple.

Le système stocke les erreurs ou problèmes rencontrés lors de cette vérification. Il renvoi un rapport d'erreur pour informer l'utilisateur des modifications à apporter.

Quand il n'y a plus d'erreurs le système ajoute le participant au système et indique à l'utilisateur que l'ajout s'est bien passé.

e. Créer Tour



powered by Astah

Illustration 7: Diagramme de séquence (System Level) - Crée Tour

Comme énoncé dans la description du cas d'utilisation associé, la création d'une édition du tour de France suit un processus bien précis. Étant donné le nombre conséquent de données à saisir, ce processus est divisé en plusieurs phases, tout en essayant de rester le plus intuitif possible pour l'utilisateur.

L'administrateur commence par créer un tour, en renseignant les informations générales relatives à celui-ci, c'est à dire son nom et son édition, et éventuellement le commissaire de course, les autres informations tel que les dates de début et de fin étant mise à jour automatiquement en fonction des étapes créées. Ensuite, l'administrateur peut créer les étapes une par une et dans l'ordre. Le nom de l'étape, la date et le type d'étape sont à indiquer, les autres données pouvant être déduites des points de passage associés à l'étape. Enfin, pour chaque étape, l'administrateur peut créer les points de passage un par un et dans l'ordre. Là encore, il faudra saisir le nom du point de passage, son altitude, sa distance par rapport à son point de départ.

Il est important de noter que l'administrateur peut s'arrêter à tout moment. Les informations seront tout de même sauvegardées dans la base. Le processus de création de tour pourra être repris à tout moment.

3. Domain Model

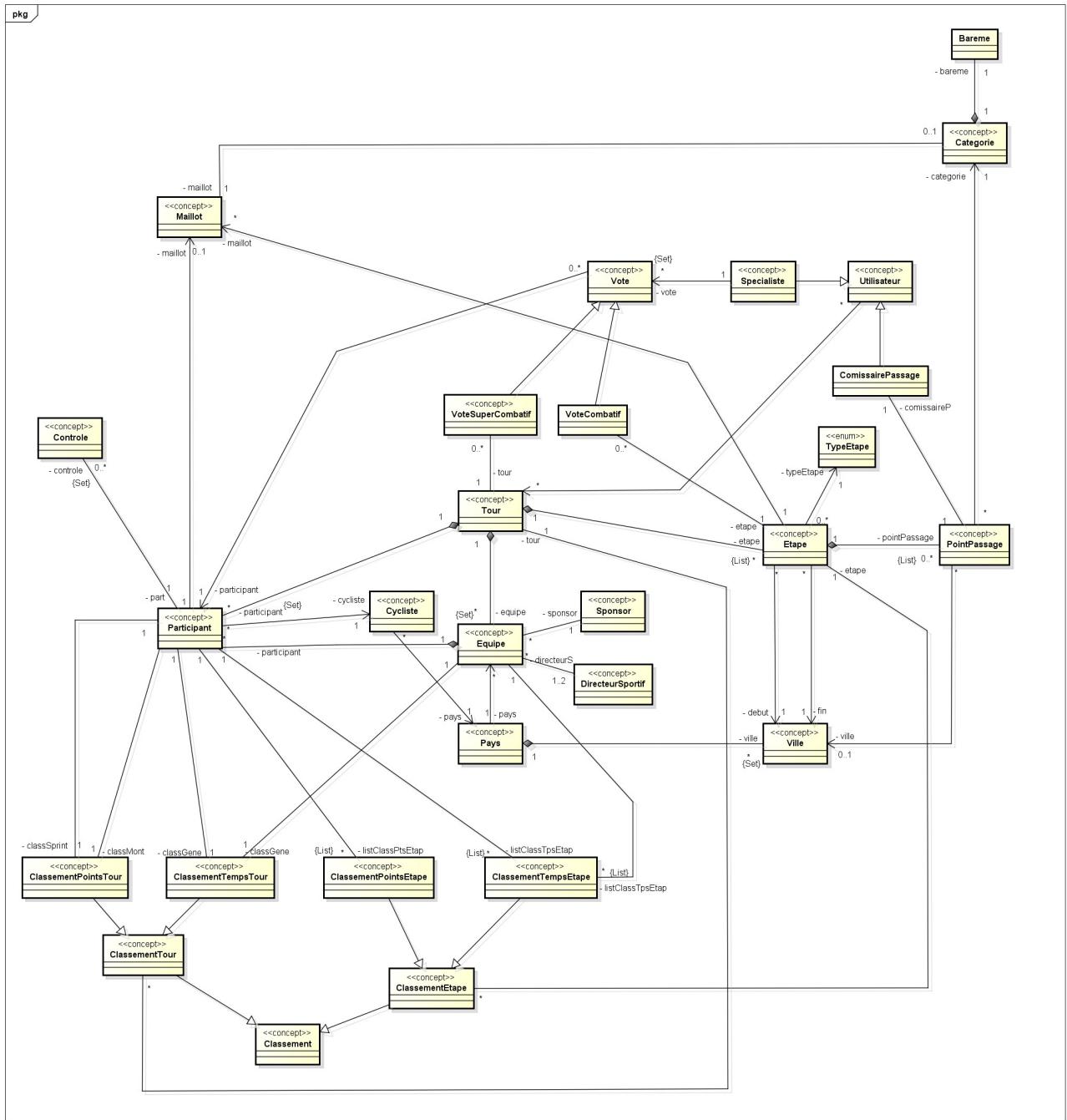


Illustration 8: Domain Model

powered by Astah

Le « domain model » permet de modéliser les concepts qui sont présents dans le système. Il s'agit de représenter un problème donné à l'aide d'un formalisme standardisé. Ici, nous utiliserons les éléments du diagramme de classe.

La classe *Tour* est au cœur du système. C'est depuis cette classe que tous les éléments importants du système vont dépendre.

Ce qui caractérise un *Tour* est un ensemble d'étapes, de participants et d'équipes. Sans le *Tour* ces éléments n'ont pas lieu d'être et cela justifie l'utilisation de la composition pour relier *Tour*

à *Participant*, *Equipe* et *Etape*. *Tour* aura donc comme attributs des « SetCollection» correspondant au nombre d'équipes et de participants d'un Tour. L'utilisation de « SetCollection » est justifié par le fait qu'un participant et un équipe sont uniques lors d'un Tour. La notion d'ordre n'a pas lieu d'être dans ce cas. *Tour* aura également comme attribut une « ListCollection » pour regrouper les étapes. La notion d'ordre peut être intéressante par rapport à la chronologie des étapes.

Bien entendu un participant, un équipe ou une étape est propre à un seul Tour.

Un *Participant* du *Tour* est associé à un unique *Cycliste*. Un *Cycliste* n'est pas censé avoir connaissance de son association avec un *Participant*, c'est pourquoi l'association est orientée.

Lors d'un Tour, un participant est présent dans différents classements (général, montagne, sprint) en fonction de points ou du temps. Pour gérer la notion de classement, nous avons opté pour une distinction des classements en classement relatif à Tour ou à une seule étape. Ces deux types de classement héritent de la classe *Classement*. L'héritage est utilisé vu que des attributs et méthodes seront communes mais les classes enfants auront des informations supplémentaires en rapport au *Tour* ou à l'*Etape* associé.

ClassementTour et *ClassementEtape* sont une nouvelle fois distingués en classement utilisant des points ou des classements utilisant des temps. L'héritage est à nouveau requis pour cette situation.

Participant possède donc un *ClassementPointsTour* pour le classement du meilleur sprinteur et un autre *ClassementPointsTour* pour le meilleur grimpeur. De même, il possède un *ClassementTempsTour* pour le classement général. Pour chaque étape un *ClassementPointsEtape* et un *ClassementTempsEtape* est instancié et ils sont gérés grâce aux « ListCollection ». La notion d'ordre est importante pour assurer une cohérence des classements.

La gestion des classements pour les équipes est similaires sauf que les équipes ne sont pas présentent dans les classements par points.

Une étape est d'un certain type. Ce type est défini à l'avance et ne change jamais. L'utilisation des énumérations permet d'avoir une liste de valeur possible.

Un *PointPassage* n'a pas lieu d'exister sans une étape. La relation avec *Etape* est donc une composition.

La modélisation de la gestion des utilisateurs se fait grâce à l'héritage. Un *Utilisateur* accède à un *Tour* donné. Certains utilisateurs particuliers sont en charge d'un point de passage. Il s'agit de *CommissairePassage*. Il possède les mêmes attributs que *Utilisateur* mais possède en plus les informations relatives au point de passage dont il a la charge. La relation entre les deux classes n'est pas orientée étant donné qu'on souhaite également savoir quel commissaire de point de passage était en charge d'un point de passage donné.

Partie 2 Conception

Cette partie sera dédiée à la conception architecturale et détaillée proposée pour répondre aux besoins et aux contraintes exprimées dans la première partie de ce rapport. Tout d'abord, nous présenterons l'architecture ainsi que les différents diagrammes UML relatifs à la conception détaillée. Ensuite, les différents phases inhérentes à la réalisation du projet seront décrites et argumentées.

I. Architecture

Notre système est articulé autour de cinq modules, chacun regroupant les classes dont le rôle est de fournir une fonctionnalité du système. Ceci-étant, les cinq modules sont intrinsèquement liés et ne sont pas réellement dissociables et indépendants les uns des autres.

Module Système

Ce module regroupe les éléments de plus haut niveau ainsi que les objets transversaux n'étant pas directement liées à l'une ou l'autre des fonctionnalités qu'il implémente celui-ci.

Classe	Description
Système	Classe de plus haut niveau pour la gestion de l'application
Pays	Classe modélisant un pays présent dans le système
Ville	Classe modélisant une ville présente dans le système
TourManager	Classe regroupant tous les tours référencés dans l'application

Module Utilisateurs

Ce module permet de gérer les utilisateurs de notre système, aussi bien les spécialistes que les commissaires de passage.

Classe	Description
Utilisateur	Classe représentant un utilisateur du système
CommissairePassage	Classe modélisant un commissaire de point de passage
Specialiste	Classe modélisant un spécialiste intervenant dans la désignation du combattif / super-combattif

Module Course

Représente la partie la plus importante de notre système puisque c'est dans cette partie que seront regroupée les classes relatives à la course.

Classe	Description
Tour	Classe modélisant un tour
Etape	Classe modélisant une étape d'un tour particulier
Point de passage	Classe modélisant un point de passage
Participant	Classe modélisant un participant correspondant à un tour particulier
Equipe	Classe modélisant une équipe

Module Résultats

C'est au sein de ce module que seront regroupées les classes nécessaires à la génération et au calcul des résultats.

Classe	Description
ClassementCollection	Classe pouvant regrouper plusieurs collection de <i>Classement</i> , Chacun étant une liste de résultat associé à un objet (une étape, un tour, une équipe ou un participant). Chaque collection peut être interprétée comme un classement particulier (maillot vert, maillot jaune, maillot à pois, ...)
Classement	Peut être vu comme un résultat (une place à un classement)
ClassementTour	Objet modélisant un résultat pour un classement attaché à un tour
ClassementEtape	Objet modélisant un résultat pour un classement attaché à une étape
ClassementPointsTour	Objet modélisant un résultat pour un classement attaché à un tour de type points
ClassementTempsTour	Objet modélisant un résultat pour un classement attaché à un tour de type temps
ClassementPointsEtape	Objet modélisant un résultat pour un classement attaché à une étape de type points
ClassementTempsEtape	Objet modélisant un résultat pour un classement attaché à une étape de type temps
Barème	Entité modélisant l'attribution d'un nombre de points en fonction du classement à points donné et de la position à laquelle le participant s'est classé.

Module Dopage

Ce petit module est utilisé pour la gestion des contrôles anti-dopages ainsi que de leur résultats.

Classe	Description
Contrôle	Classe utilisée pour la gestion des contrôles anti-dopage

Interactions et dépendances

Finalement, les interactions et les dépendances qu'entretiennent les différents packages les uns avec les autres peuvent être représentées avec le diagramme suivant.

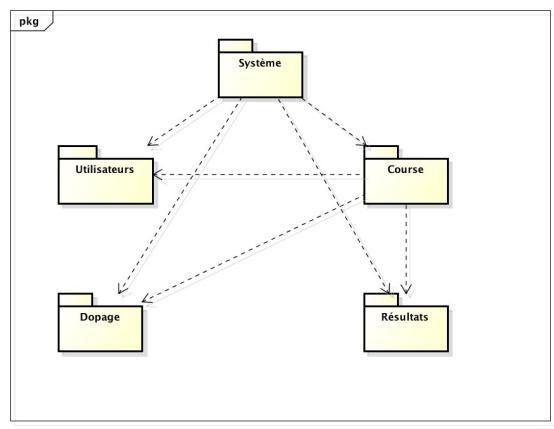


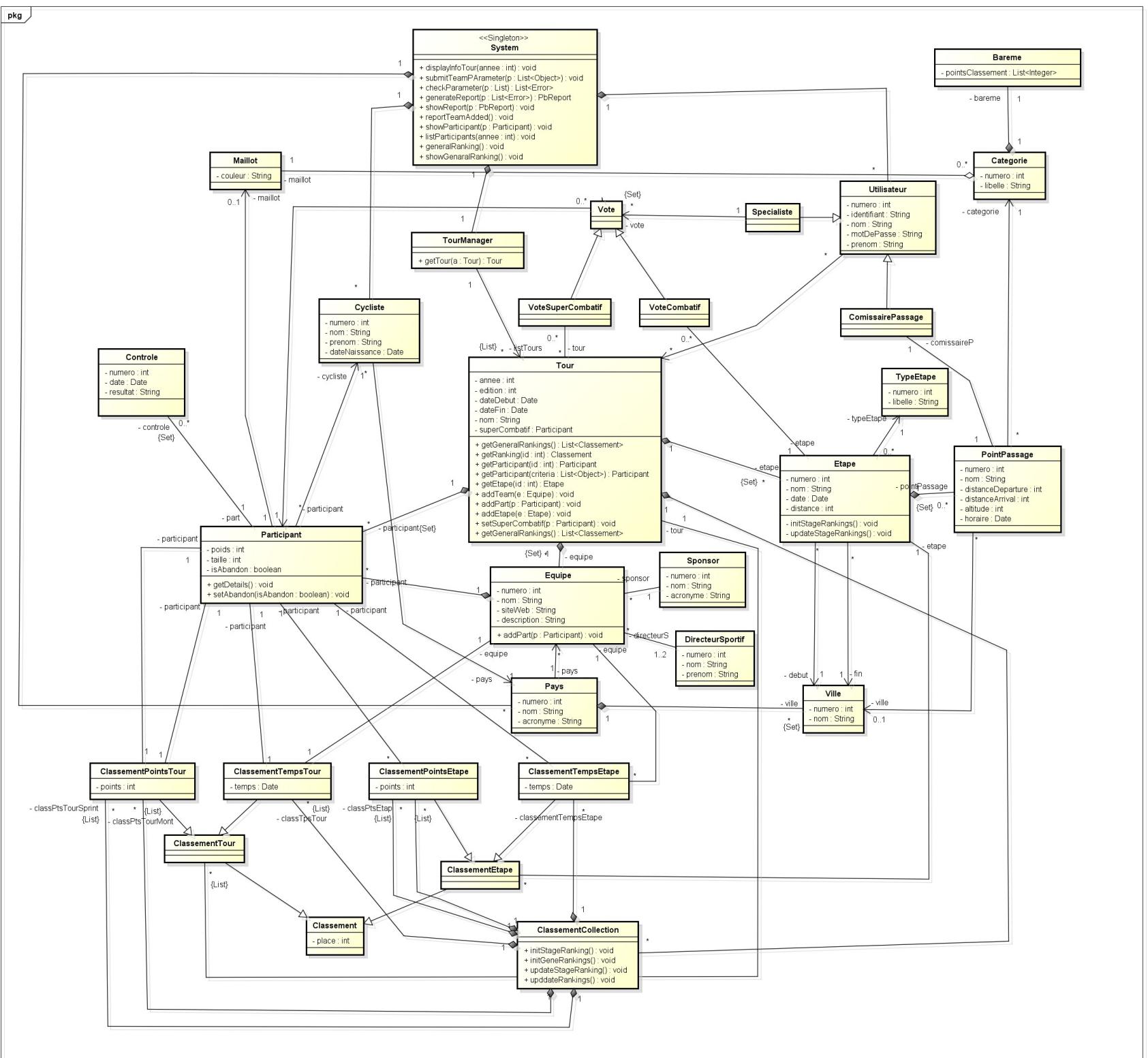
Illustration 9: Packages de l'application

II. Conception détaillée

1. Diagramme de classe

Le diagramme de classe est calqué sur le diagramme *Domain-Model* précédemment présenté. Comme on peut le voir, une classe *System* a été implémentée. Elle constitue la classe de plus haut niveau hiérarchique et permet de regrouper l'ensemble des objets de l'application au même endroit. Bien entendu il n'existe qu'une seule instance de cet objet au sein de l'application ; c'est pourquoi le stéréotype *Singleton* lui a été attribué. Cette classe permettra de gérer des données qui ne sont pas liées à un tour en particulier, mais plus à l'ensemble du système, tel que les différents utilisateurs, ou encore les participants. C'est aussi par cette classe que *TourManager* sera accessible.

Même si la classe *System* est considérée comme à la racine de notre logiciel, beaucoup des traitements se produisant durant l'application sont attachés au niveau de la classe *Tour*. En effet, les classements, les participants et les équipes sont liés à un tour particulier et il n'est pas nécessaire d'accéder à d'autres informations que celles contenues (directement ou indirectement) dans cet objet.



2. Diagrammes de séquence

Dans la partie précédente, des diagrammes de séquence ont déjà été présentées. Comme nous l'avons vu ces diagrammes étaient de niveau système (*system level*) et s'attachaient chacun à un cas d'utilisation particulier. Il est à présent question de décrire sous forme de séquence chacun de ces cas d'utilisation, mais en décomposant le système afin d'en présenter les rôles des différents modules et des différentes classes. Ainsi chacun des diagrammes de séquence de niveau objet (*object level*) correspond à un diagramme de séquence de niveau système.

a. Visualiser informations coureurs

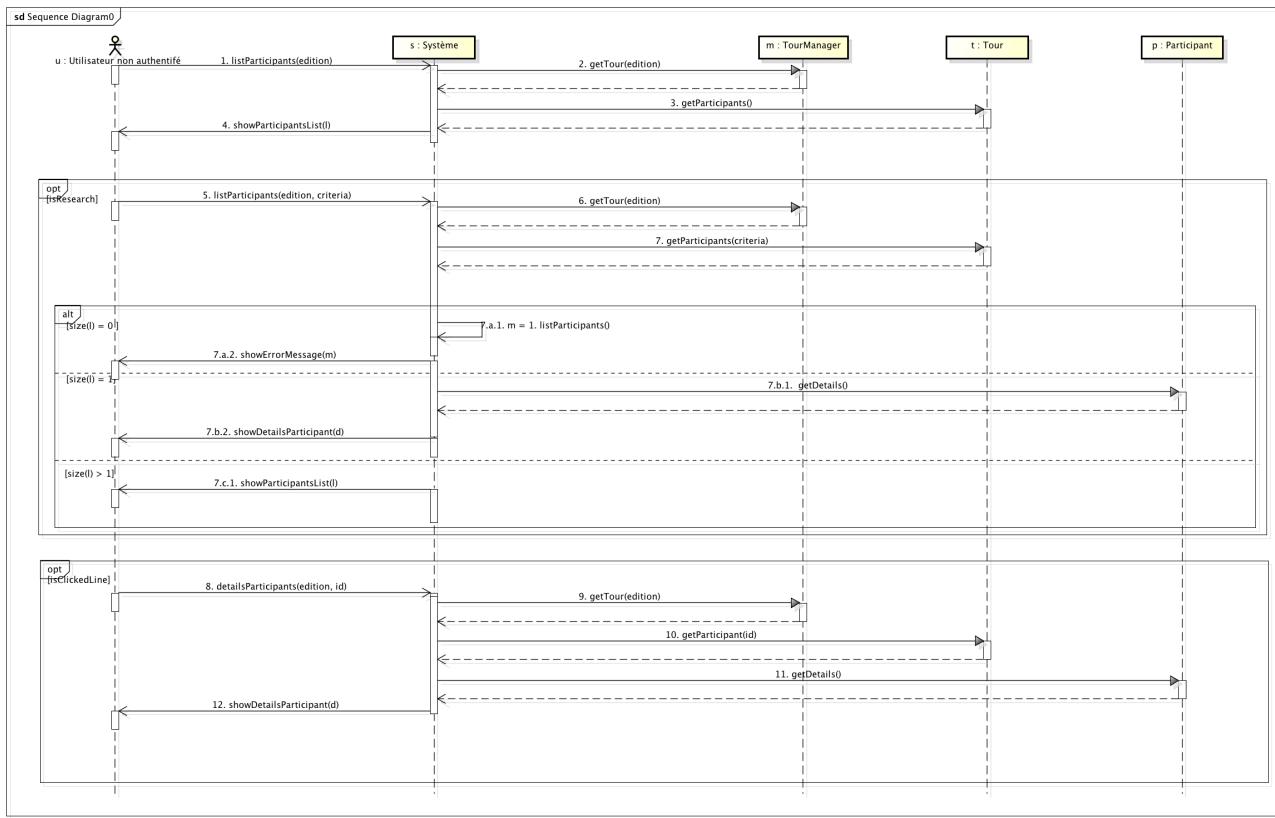


Illustration 11: Diagramme de séquence - Visualiser informations coureur – Object Level

Logiquement, le diagramme de séquence au niveau objet reprend la même logique que le diagramme correspond au niveau système. Il est cependant nécessaire de donner des explications quand au rôle de chacun des objets et en quoi consistent les méthodes invoquées.

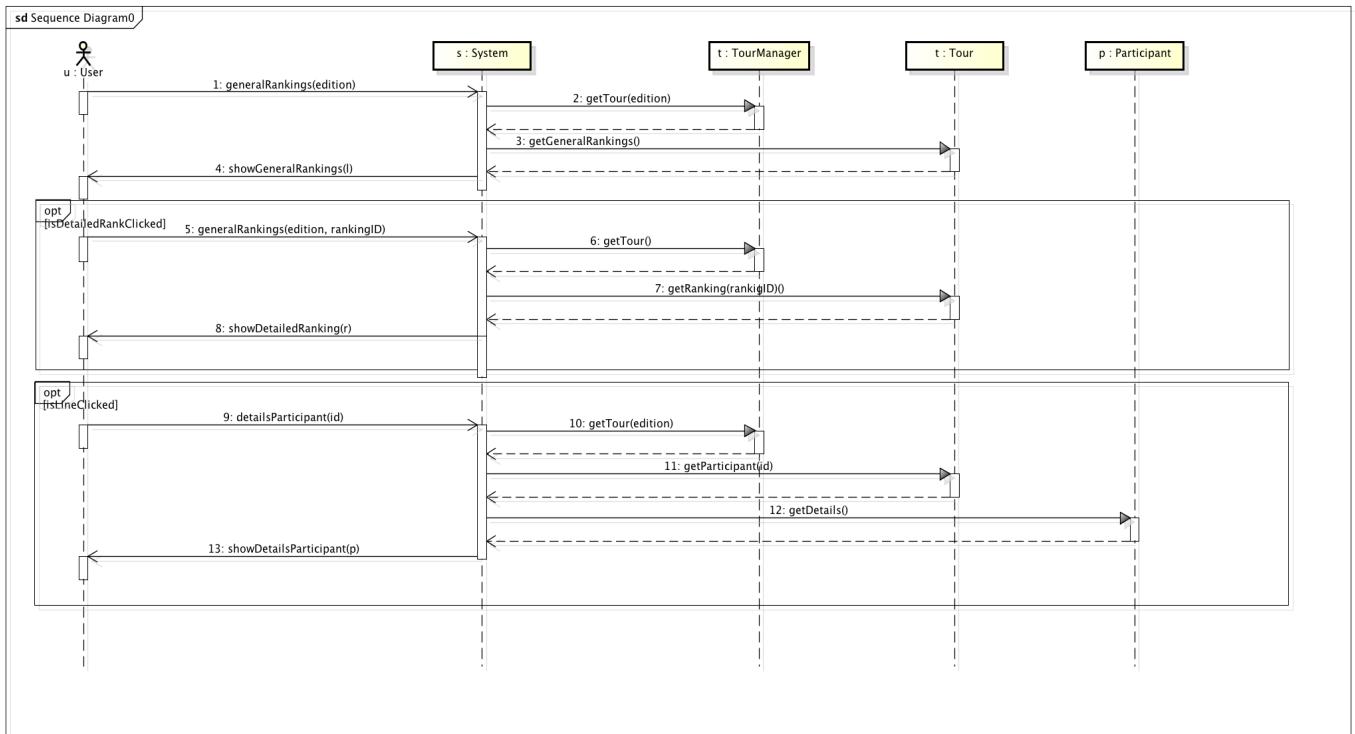
Le *TourManager* encapsule toutes les informations relatives aux éditions de course. En réalité, on peut dire que cet objet encapsule une liste d'instance de *Tour*, et met à disposition quelques méthodes permettant leur manipulation, notamment *getTour*, qui en fonction de l'année d'édition donnée en paramètre, retournera l'instance du tour correspondante.

La classe *Tour* encapsule toutes les données relatives au tour, tel que les étapes, les équipes ou encore les participants. Une méthode *getParticipants* est utilisée pour retourner une liste de l'ensemble des participants inscrits au tour. Il est possible de passer à cette fonction un paramètre afin de ne récupérer qu'une sous-ensemble des participants correspondant à certains critères.

La classe *Participant* regroupe les données relative à un participant. La méthode associée

getDetails retourne l'ensemble de ces informations formatées pour leur affichage.

b. Visualiser classement



powered by Astah

Illustration 12: Diagramme de séquence - Visualiser classement – Object Level

On retrouve dans ce diagramme les mêmes objets que dans le diagramme de séquence précédent. La méthode *getGeneralRankings* d'un tour est utilisé pour récupérer une liste des différents classements courants de manière abrégée (i.e. les cinq premiers de chaque classement par exemple). La méthode *getRanking* retourne un classement particulier, mais de manière complète ; le numéro de classement souhaité est alors fourni en paramètre.

c. Créer équipe

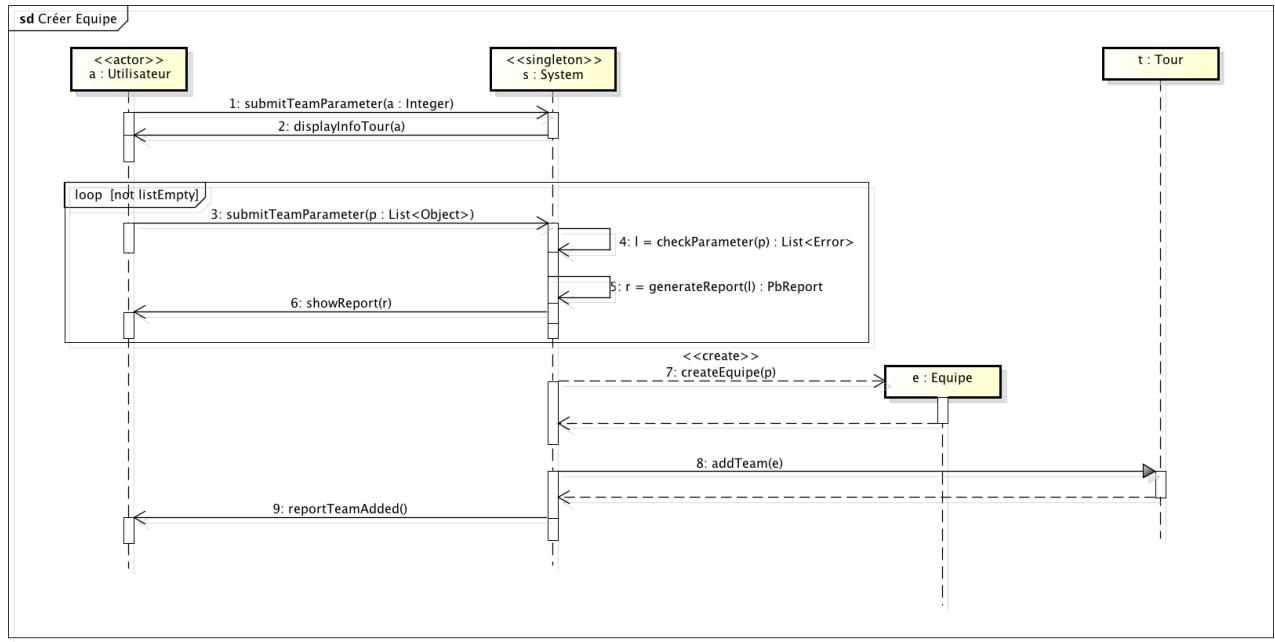


Illustration 13: Diagramme Séquence - Crée équipe - Object Level

powered by Astah

Les actions de l'administrateur sont les mêmes que celles présentes dans la description accompagnant le diagramme de séquence System Level. Les compléments d'informations portent sur les actions entre les objets du système.

Quand le système valide les informations saisies par l'administrateur, une instance de l'objet "Equipe" est créée à l'aide des paramètres envoyés via le formulaire. Une fois cet objet créé, le System ajoute l'équipe au Tour correspondant. En effet, celui a été sélectionné avant la saisie des informations. Les équipes sont stockées sous forme de liste dans l'instance de Tour.

d. Créer participant

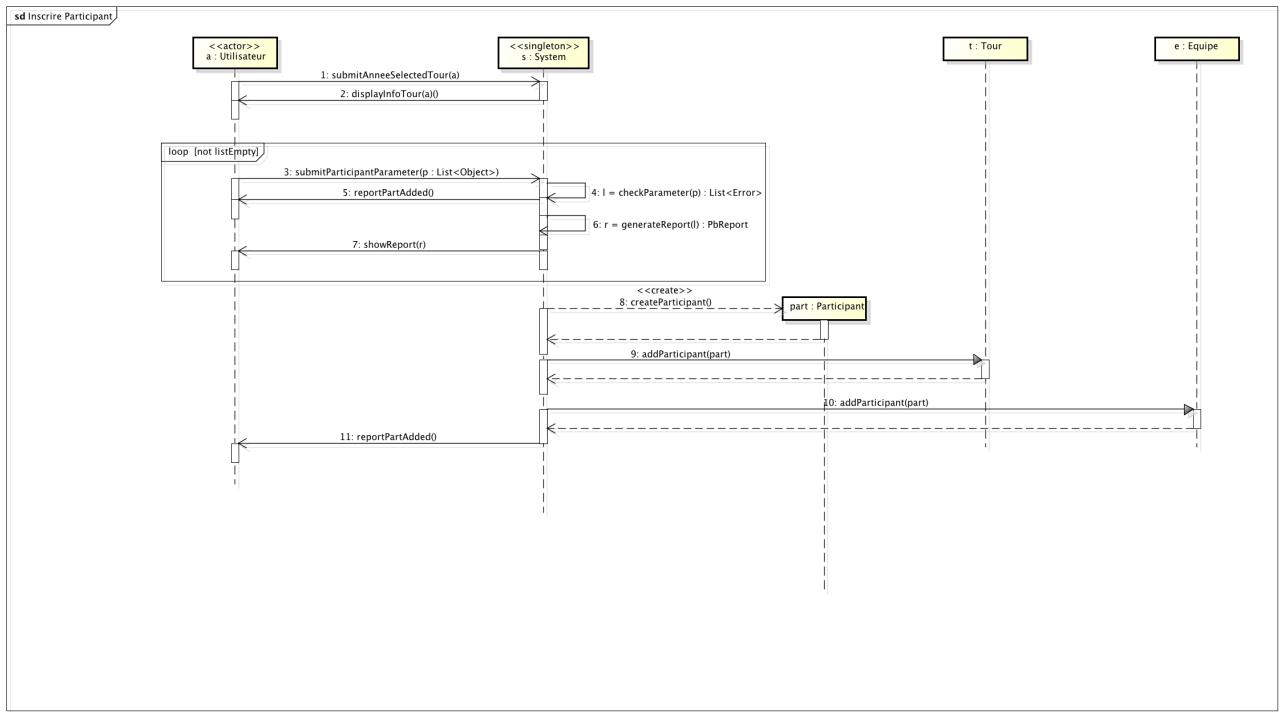


Illustration 14: Diagramme séquence - Incrire Participant - Object Level

powered by Astah

Les actions de l'administrateur sont les mêmes que celles présentes dans la description accompagnant le diagramme de séquence System Level. Les compléments d'informations portent sur les action entre objets du système.

Quand le système à valider les informations saisies par l'administrateur, une instance de l'objet *Participant* est créée à l'aide des paramètres envoyés via le formulaire.

Le système ajoute le participant au Tour sélectionné et à l'équipe correspondante. Le Tour stockent les participants sous forme de liste.

e. Créer Tour

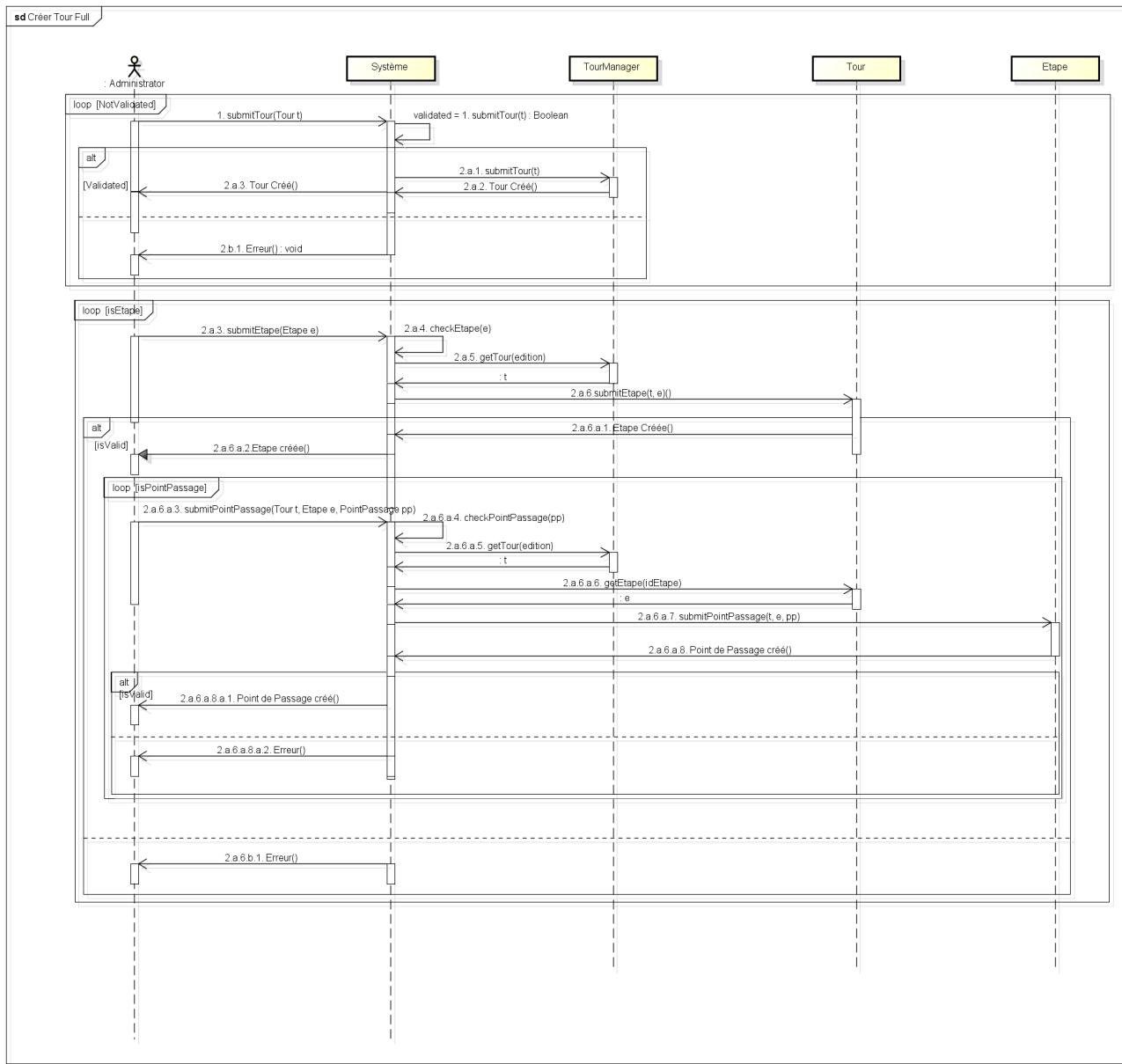


Illustration 15: Diagramme séquence - Crée Tour - Object Level

powered by Astah

Les échanges entre l'utilisateur et le système sont décrits en compléments du diagramme de séquence System Level.

Pour ce qui est des échanges entre les instances des objets du système, ce dernier commence par créer une instance de Tour si les paramètres saisis dans le formulaire sont corrects.

L'administrateur peut s'arrêter à ce moment ou ajouter des étapes s'il le souhaite.

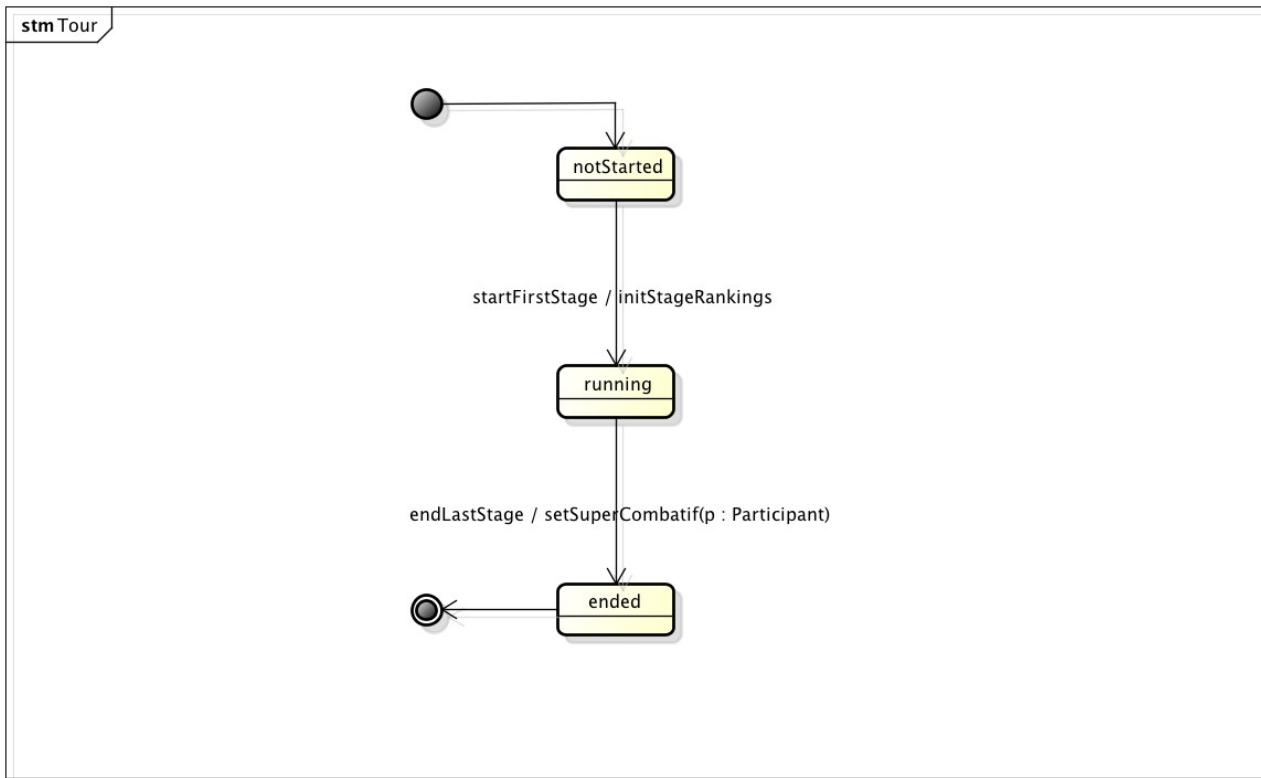
Si les informations saisies pour la création d'étape sont correctes, l'étape est ajoutée au Tour sélectionné.

Le processus est le même pour les points de passage. Si les informations saisies pour la création du point de passage sont correctes, il est ajoutée à l'étape sélectionnée.

3. Diagrammes d'état-transition

Les diagrammes d'états transition permettent de décrire l'évolution des différents objets du système. Nous avons retenu quatre éléments dont la description du comportement paraît pertinente.

a. Tour



powered by Astah

Illustration 16: Diagramme d'état-transition - Tour

Un tour peut avoir trois états :

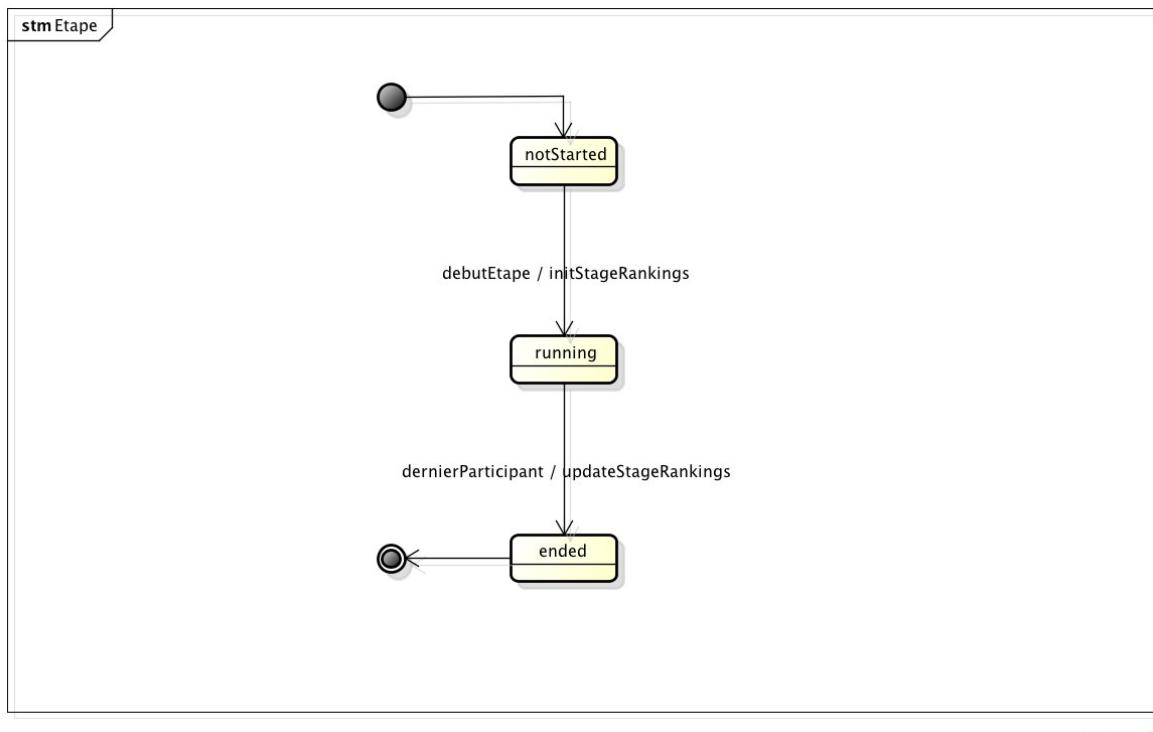
notStarted : le tour n'a pas encore démarré

running : le tour est en cours (lorsque la première étape a commencée)

ended : le tour est terminé (lorsque la dernière étape s'est terminée)

La transition d'un état à l'autre déclenche un certain nombre de traitements, qui concerne par exemple l'initialisation du classement général au départ de la course ou encore l'élection du super combatif n'intervenant que lorsque le tour arrive à son terme.

b. Étape



powered by Astah

Illustration 17: Diagramme d'état-transition - Etape

Le diagramme d'état transition décrivant le comportement de l'objet *Etape*. Il est très similaire au diagramme d'état transition concernant l'objet *Tour*, avec trois états possibles

noStarted : l'étape n'a pas débuté

running : l'étape a débuté est au moins un participant est toujours en course

ended : tous les participants sont arrivées (ou on abandonné).

Les transition d'états vont déclencher un certain nombre de traitements. La transition de l'état *running* à *ended* va par exemple permettre la création du classement relatif à l'étape et la mise à jour du classement courant pour le tour.

c. Participant

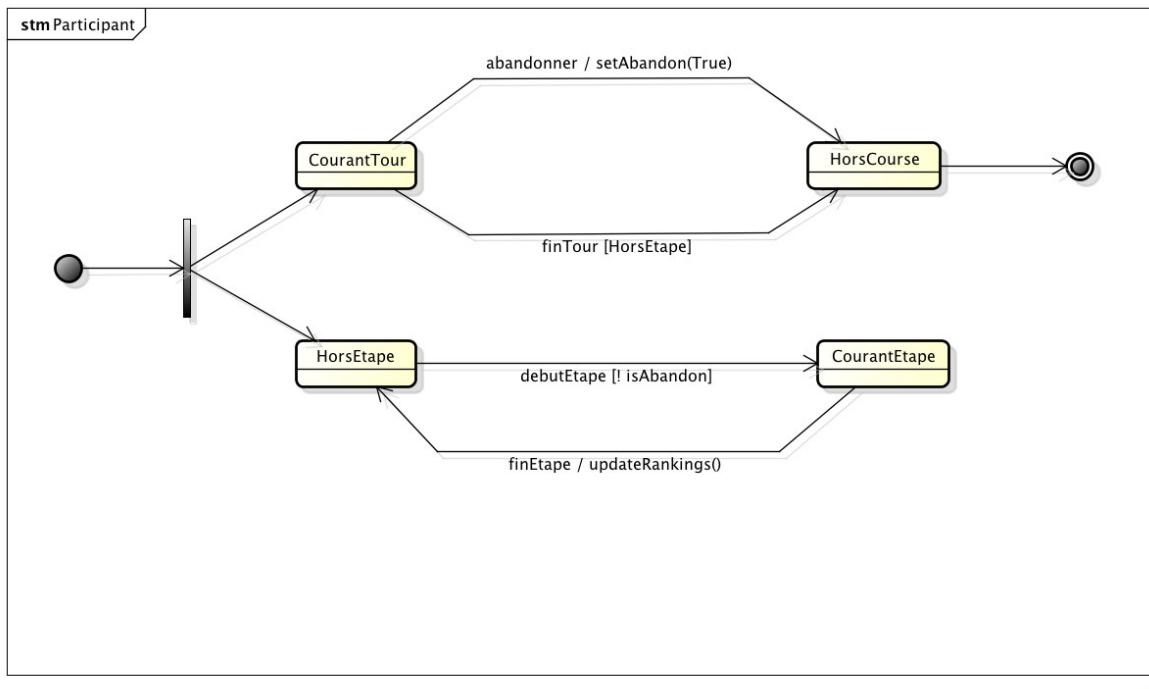


Illustration 18: Diagramme d'état-transition - Participant

Un participant a deux états simultanés. Un des états décrit son statut par rapport au tour, et l'autre est statut par rapport à l'étape. Par exemple, le participant peut toujours être en course (*courantTour*) mais *horsEtape* s'il y a un jour de repos.

CourantTour : le participant est dans le tour (c'est à dire que le participant n'est pas éliminé et la course n'est pas terminée).

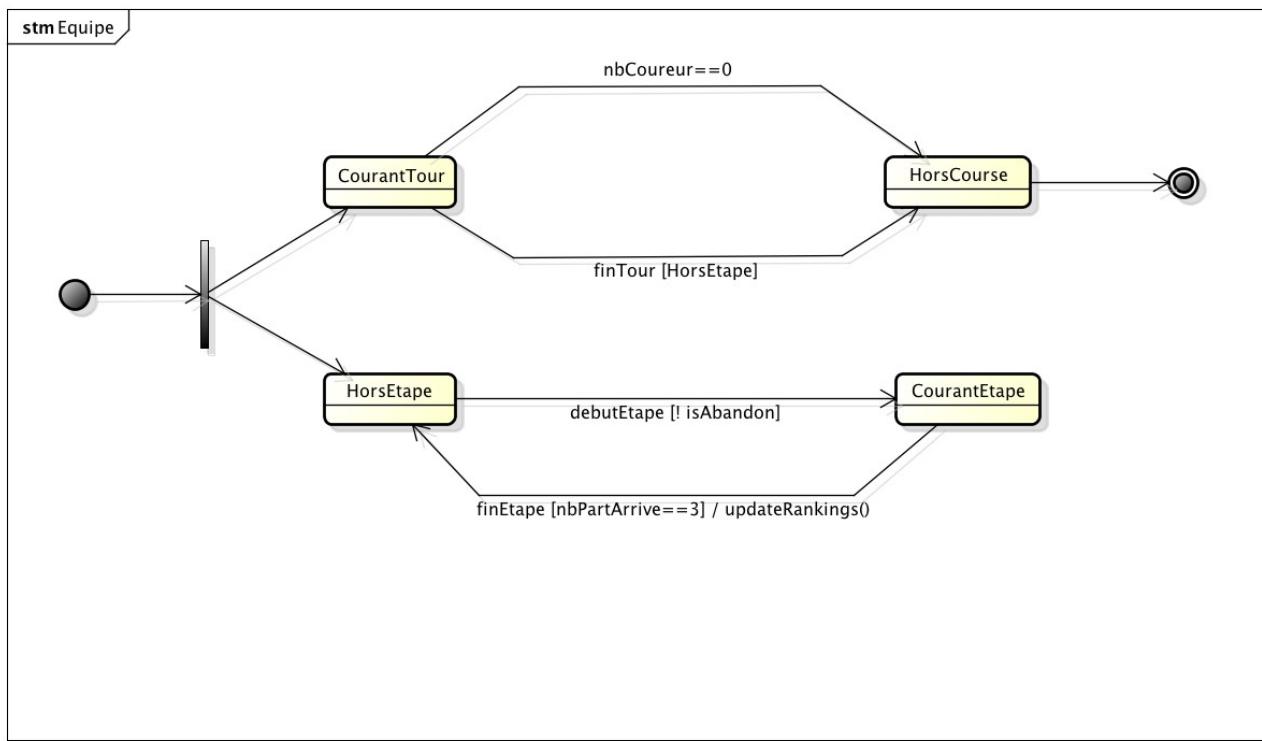
HorsCourse : le participant n'est plus dans la course (c'est à dire que le participant a été éliminé ou le tour est terminé).

CourantEtape : le participant est dans une étape (c'est à dire que l'étape a débuté et qu'il n'est pas encore arrivé) .

HorsEtape : le participant n'est pas dans une étape (c'est à dire que cette dernière n'a pas encore débuté ou le participant est déjà arrivé).

Là encore, un certain nombre de traitements sont effectués au passage de certaines transitions, tel que la mise à jour des différents classements.

d. Équipe



powered by Astah

Illustration 19: Diagramme d'état transition - Équipe

La dynamique d'une équipe est très proche de celui d'un participant, si ce n'est une subtilité ; en effet, on considère qu'une équipe a terminé une étape au moment où trois de ces cyclistes ont franchi la ligne d'arrivée. En effet, seul les trois premiers coureurs d'une équipe sont utilisés pour le calcul du classement par équipe.

III. Dictionnaire de données

Nom Conceptuel	Nom Logique ou Alias	Type E/Ca/Co	Nature	Longueur	Type Win'Design	Remarques
Acronyme	SPON_ACRON	E	AN	4	VA4	
Altitude Point Passage	PT_PASS_ALT	E	N	4	N4	
Année Tour	TOUR_ANNEE	E	N	4	N4	
Couleur Maillot	MAILLOT_COULEUR	E	AN	16	VA16	
Date Contrôle	CONTR_DATE	E	D	8	D8	
Date début Tour	TOUR_DATED	E	D	8	D8	
Date Etape	ETAPE_DATE	E	D	8	D8	
Date Fin Tour	TOUR_DATEF	E	D	8	D8	
Date Naissance Cycliste	CYCLISTE_DATEN	E	D	8	D8	
Description Equipe	EQUIPE_DESC	E	AN	100	VA100	
Distance Etape	ETAPE_DISTANCE	E	N	3	N3	
Edition Tour	TOUR_EDITION	E	N	3	N8	
Horaire prévu	PT_PASS_HORAIRE	E	H	4	H4	Heure approximative d'arrivée de la tête de la course
Km Arrivée	PT_PASS_KM_ARR	E	N	3	N3	Nombre de km depuis le début de l'étape
Km Départ	PT_PASS_KM_DEP	E	N	3	N3	Nombre de km restant jusqu'à la fin de l'étape
Libellé Catégorie	CAT_LIB	E	AN	30	VA30	
Libellé Profil	PROFIL_LIB	E	AN	40	VA40	
Libellé Type Etape	TETAPE_LIB	E	AN	30	VA30	
Mot de passe	UTIL_MDP	E	AN	20	VA20	
Nom Cycliste	CYCLISTE_NOM	E	AN	30	VA30	
Nom Directeur Sportif	DIRS_NOM	E	AN	30	VA30	
Nom Equipe	EQUIPE_NOM	E	AN	30	VA30	
Nom Etape	ETAPE_NOM	E	AN	30	VA30	
Nom Pays	PAYS_NOM	E	AN	30	VA30	
Nom Point Passage	PT_PASS_NOM	E	AN	30	VA30	
Nom Spécialiste	SPE_NOM	E	AN	30	VA30	

Nom Conceptuel	Nom Logique ou Alias	Type E/Ca/Co	Nature	Longueur	Type Win'Design	Remarques
Nom Sponsor	SPON_NOM	E	AN	30	VA30	
Nom Tour	TOUR_NOM	E	AN	30	VA30	
Nom Utilisateur	UTIL_NOM	E	AN	30	VA30	
Nom Ville	VILLE_NOM	E	AN	30	VA30	
Numéro Catégorie	CAT_NUM	E	N	4	N4	
Numéro Contrôle	CONTR_NUM	E	N	4	N4	
Numéro Cycliste	CYCLISTE_NUM	E	N	4	N4	
Numéro Directeur Sportif	DIRS_NUM	E	N	4	N4	
Numéro Dossard	PART_NUM	E	N	3	N3	
Numéro Equipe	EQUIPE_NUM	E	N	4	N4	
Numéro Etape	ETAPE_NUM	E	N	4	N4	
Numéro Pays	PAYS_NUM	E	N	N	N4	
Numéro Point Passage	PT_PASS_NUM	E	N	4	N4	
Numéro Profil	PROFIL_NUM	E	N	4	N4	
Numéro Spécialiste	SPE_NUM	E	N	4	N4	
Numéro Sponsor	SPON_NUM	E	N	4	N4	
Numéro Type Etape	TETAPE_NUM	E	N	4	N4	
Numéro Utilisateur	UTIL_NUM	E	N	4	N4	
Numéro Ville	VILLE_NUM	E	N	4	N4	
Place Barème	BAREME_PLACE	E	N	2	N2	
Poids Participant	PART_Poids	E	N	3	N3	
Points Barème	BAREME PTS	E	N	2	N2	
Points Montagne	PART PTS_MONT	Ca	N	3	N3	
Points Sprint	PART PTS_SPRINT	Ca	N	3	N3	
Prénom Cycliste	CYCLISTE_PRENOM	E	AN	30	VA30	
Prénom Directeur Sportif	DIRS_PRENOM	E	AN	30	VA30	
Prénom Spécialiste	SPE_PRENOM	E	AN	30	VA30	
Prénom Utilisateur	UTIL_PRENOM	E	AN	30	VA30	
Pseudonyme Utilisateur	UTIL_PSEUDO	E	AN	20	VA20	
Site Internet Equipe	EQUIPE_WEB	E	AN	30	VA30	
Taille Participant	PART_TAILLE	E	N	3	N3	
Temps Equipe Général	EQUIPE_TPS_GENE	Ca	N	8	N8	Timestamp correspondant au nombre de dixième de secondes
Temps Général	PART_TPS_GENE	Ca	N	8	N8	Timestamp correspondant au nombre de dixièmes de secondes

Étant donné le nombre conséquent d'informations présentes dans le tableau, nous allons maintenant apporter une précision sommaire quant à la signification de chacun des éléments qui vont être les entités de notre modèle conceptuelle

a. Barème

Utile pour connaître le nombre de point associé à une place donnée pour une certaine catégorie. Par exemple, on pourra savoir que la 4ème place pour un passage comptant pour le classement du meilleur grimpeur sera valorisée à x points

b. Catégorie

Définit les différents types de classements par points, c'est à dire le classement du meilleur sprinteur et le classement du meilleur grimpeur.

c. Contrôle

Concerne les contrôle anti-dopage effectués sur les coureurs dans le cadre du tour de France. On conserve la date de réalisation du contrôle.

d. Cycliste

Cette entité représente une personne physique. Elle contient les informations relatives à chaque personne, tel qu'un identifiant unique, le nom et le prénom.

e. Directeur

Cette entité représente les directeurs sportifs qui dirigent une équipe de coureurs lors d'un tour. Chaque directeur est identifié par un numéro unique et possède un prénom et un nom.

f. Équipe

Le tour de France impose que chaque coureur fasse partie d'une équipe. Une équipe a un identifiant unique, un nom et un site internet.

g. Étape

Le tour de France est constitué de plusieurs étapes, chacune étant organisée sur une journée. Elle est qualifiée par un numéro d'étape (sa position dans l'ensemble de la course), un nom d'étape, une date où elle est courue et la distance séparant la ligne de départ à la ligne d'arrivée.

h. Maillot

Chaque maillot est associé à un prix dans le tour de France. Par exemple, le maillot vert est associé au meilleur sprinteur courant depuis le début de la course, et le maillot jaune est porté par le coureur le plus rapide depuis le début de la course.

i. Participant

Le participant correspond à un coureur particulier lors d'une course donnée. Un participant est identifié par son numéro de dossard, unique pour un tour donné. Le poids et la taille sont également associés au participant plutôt qu'au coureur, car il peuvent varier d'une année sur l'autre.

j. Pays

Cette entité est utile pour qualifiée des éléments du modèle à une nationalité où une localisation géographique. On pense aux coureurs, aux directeurs d'équipe, aux équipes ou encore aux villes.

k. Point de passage

Constitue le niveau de granularité le plus fin dans une course. Les points de passage se situent tout au long du parcours de la course et permet de segmenter chaque étape pour obtenir un tracé et des données très précises. Un point de passage est identifié par un numéro (sa position dans l'étape), et possède une altitude, une longitude et une latitude.

l. Sponsor

Un sponsor est généralement une société qui finance une équipe. Un sponsor est identifié par un numéro et un nom (généralement la marque de la société), et possède aussi un acronyme.

m. Spécialiste

Un spécialiste est une personne désignée par l'organisation de la course pour un tour particulier afin de composer un jury chargé de désigner le combatif de l'étape et le super-combatif (le combatif désigné pour l'ensemble du tour).

n. Tour

Cette entité représente une instance du tour de France. Il est identifiée par l'année à laquelle il se déroule, et est qualifiée d'un numéro d'édition (différent de l'année), et d'une date de départ (la date de déroulement de la première étape) et d'une date d'arrivée (la date de déroulement de la dernière étape).

o. Type étape

Il existe différents types d'étapes lors d'une course, chacun ayant des règles différentes. On recense les étapes en ligne (standard) et les étapes de contre la montre (individuel ou par équipe).

p. Utilisateur

Un utilisateur est une personne pouvant accéder au système. Les utilisateurs ont différents profils leur attribuant certains priviléges sur le système.

q. Profil utilisateur

Cette entité correspond à type un niveau d'accès au système, définissant ainsi des droits pour les différents types d'utilisateur

IV. Modèle entité association

Dans un souci de lisibilité, le modèle entité association a été décomposé en sous-modèles, chacun correspondant à une fonction ou un aspect du système. Le modèle complet est disponible en annexe.

1. Gestion des inscriptions

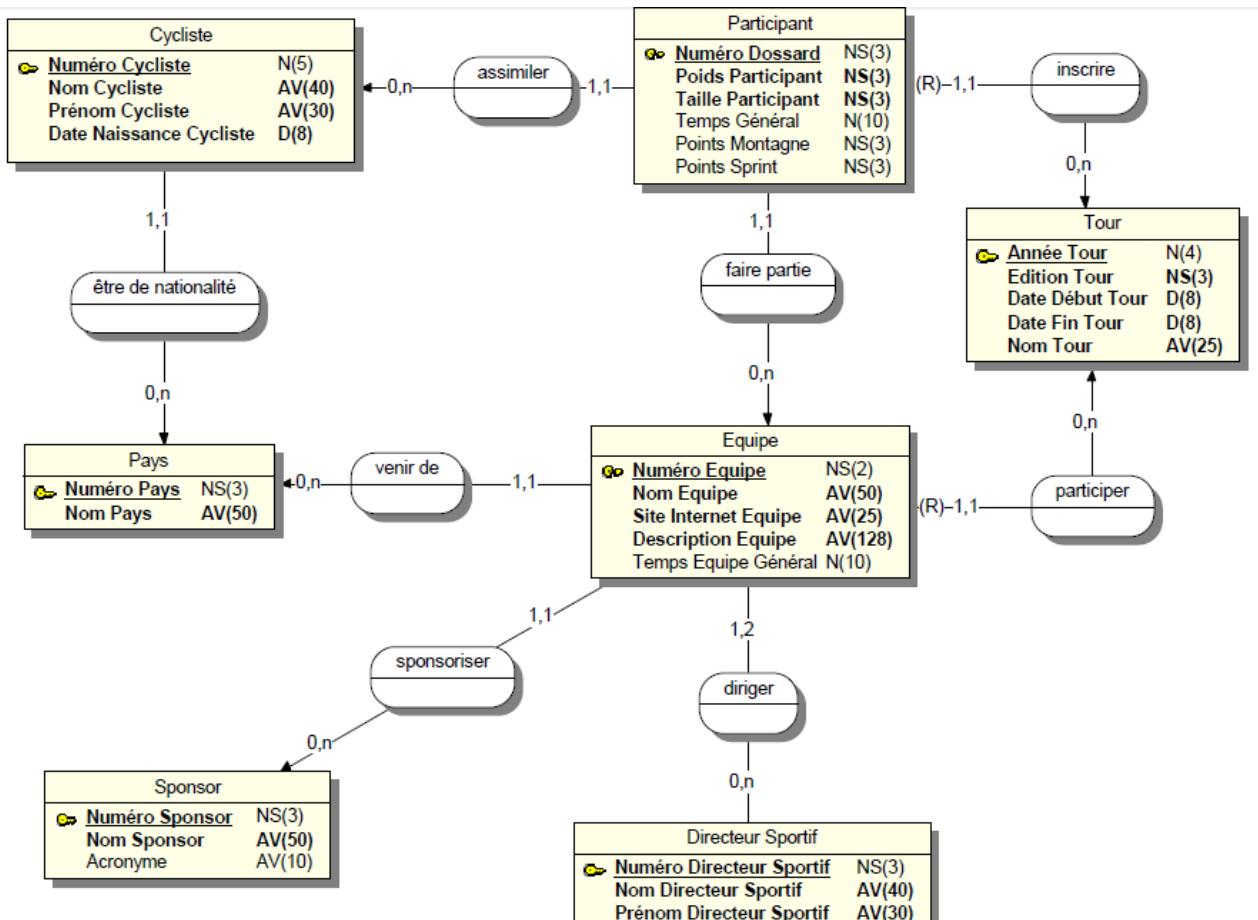


Illustration 20: MEA Gestion des inscriptions

Commentaires

Un Tour est identifié de manière unique par l'année durant laquelle il se déroule. Il est également caractérisé par le numéro de son édition, une date de début et de fin. Pour un Tour, 0, 1 ou plusieurs équipes et participant sont inscrits.

Un participant est identifié de manière relative avec l'identifiant du Tour et son numéro de dossard. Pour chaque Tour, on relève le poids et la taille du participant. En effet, son poids peut évoluer en fonction des années. On conserve également pour chaque participant son temps total, ses points pour le classement de la montagne et ses points pour le classement du meilleur sprinteur. Un participant est associé à un et un seul cycliste.

Les équipes sont aussi identifiées de manière relative avec l'identifiant du Tour et un numéro

d'équipe. Une équipe est dirigé par 1 ou 2 directeurs sportifs. Elle représente un seul sponsor et est issue d'un pays. Le sponsor est décrit à l'aide d'un numéro, d'un nom et d'un acronyme si nécessaire.

Un cycliste est caractérisé par un nom, un prénom et une date de naissance. Un cycliste est associé à un participant lorsqu'il est inscrit à un Tour. Il est d'une seule nationalité.

Un pays est caractérisé par un numéro et un nom.

2. Gestion de la course

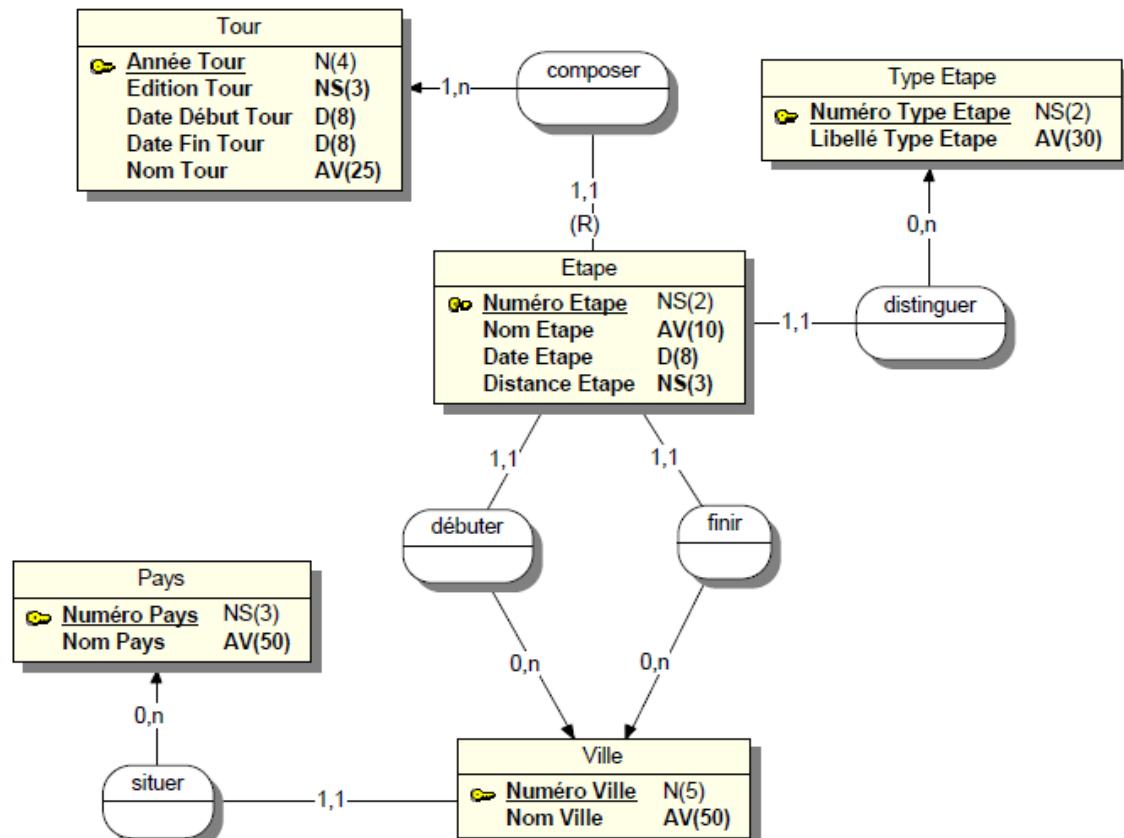


Illustration 21: MEA Gestion de la course

Commentaires

Un Tour est composé de une ou plusieurs étapes. La numérotation des étapes commence pour chaque édition à 1.

Une étape est identifiée de manière relative avec son numéro et l'identifiant du Tour. Elle est caractérisée par un nom, une date et une distance en kilomètre. Elle débute dans une et une seule ville et se termine dans une et une seule ville. Chaque ville est caractérisée par un numéro et un nom de ville. Chaque étape est d'un certains type comme par exemple étape en ligne, contre la montre individuel ou contre la montre par équipe. Chaque type est caractérisé par un numéro et un libellé.

Un ville se situe dans un et un seul pays.

3. Gestion des classements

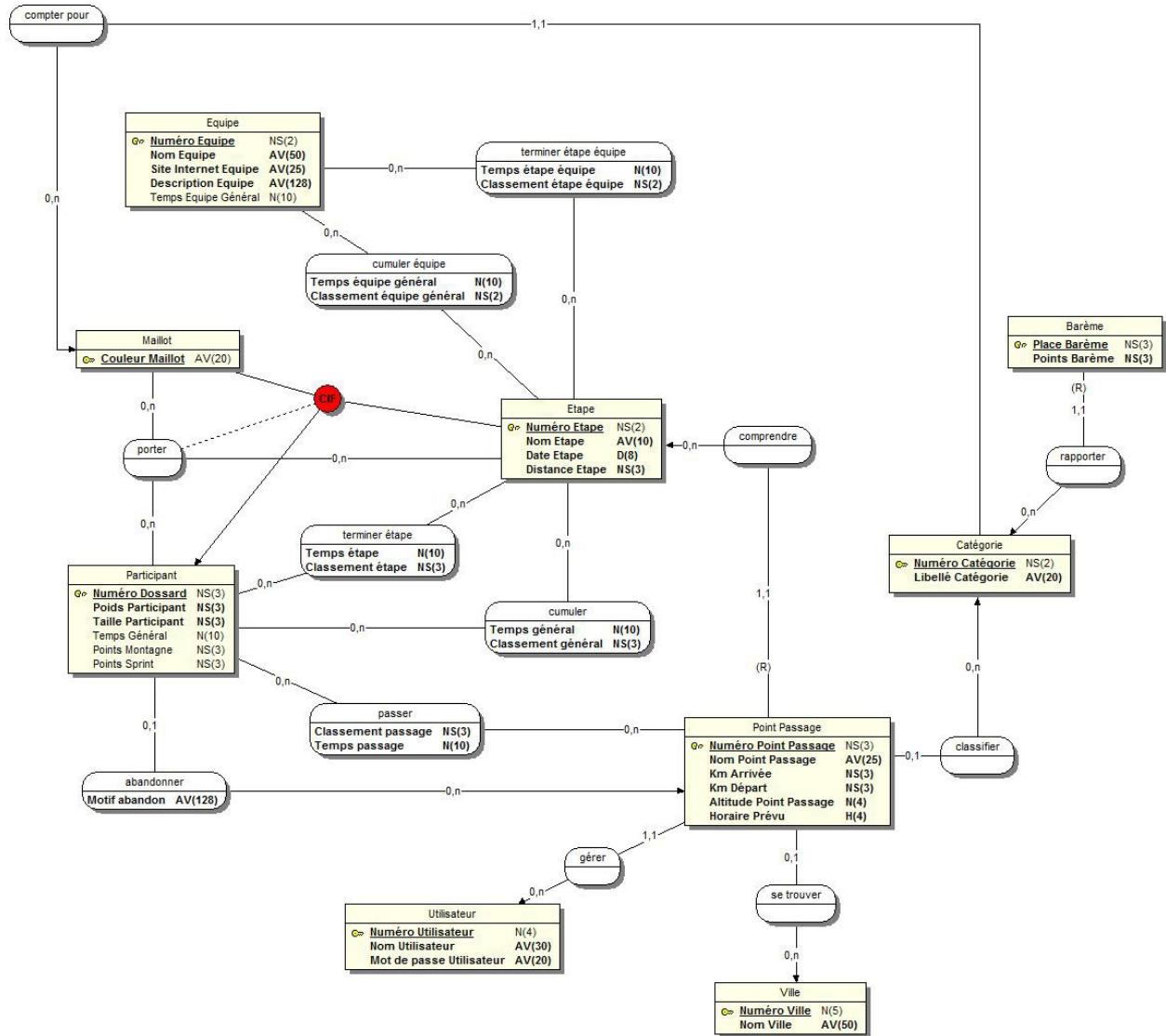


Illustration 22: MEA Gestion des classements

Commentaires

Chaque étape est composée de plusieurs points de passage. Les points de passage permettent de connaître le tracé exacte de l'étape. Ils sont caractérisés par un numéro, un nom, la distance qui le sépare de l'arrivée, la distance qui le sépare du départ, une altitude et un horaire de passage prévu. Ils peuvent être situés dans une ville.

Certains points de passage sont particuliers. Ils sont alors présents dans une catégorie donnée. Ces catégories correspondent aux différentes classifications des cols et des sprint à savoir col de catégorie 1,2,3,4 et HC et sprint intermédiaire et sprint final.

A chaque catégorie correspond un barème. Le barème est identifié de manière relative avec le numéro de la catégorie et le numéro de la place qui rapporte des points. En effet, pour chaque place un certains nombre de points est attribué. Le nombre de place donnant droit à des points peut varier selon les catégories.

Une catégorie compte pour l'obtention d'un maillot spécifique comme le maillot blanc à pois rouge ou le maillot vert. Un maillot est identifié par sa couleur.

Une équipe termine une étape avec un temps donné et un classement. On conserve également le temps cumulé et la classement général de chaque équipe à chaque fin d'étape. Cela permet de connaître la situation d'une équipe à une étape donnée. Une équipe a la possibilité de ne pas prendre part à une étape ou de ne pas la terminer.

De la même manière, le participant termine l'étape avec un classement et un temps donné. On relève aussi son temps cumulé et sa position au classement général. Un coureur a la possibilité de ne pas prendre part à une étape ou de ne pas la terminer.

Une étape peut être annulée et de se fait, aucun coureur ou équipe ne prend part à celle-ci.

Pour connaître le classement des coureurs, un relève pour chaque point de passage le classement de chaque participant.

A chaque étape, on note les participants porteurs de maillot. Pour une étape donnée et un maillot, il n'y a qu'un seul porteur.

4. Gestion du plus combatif

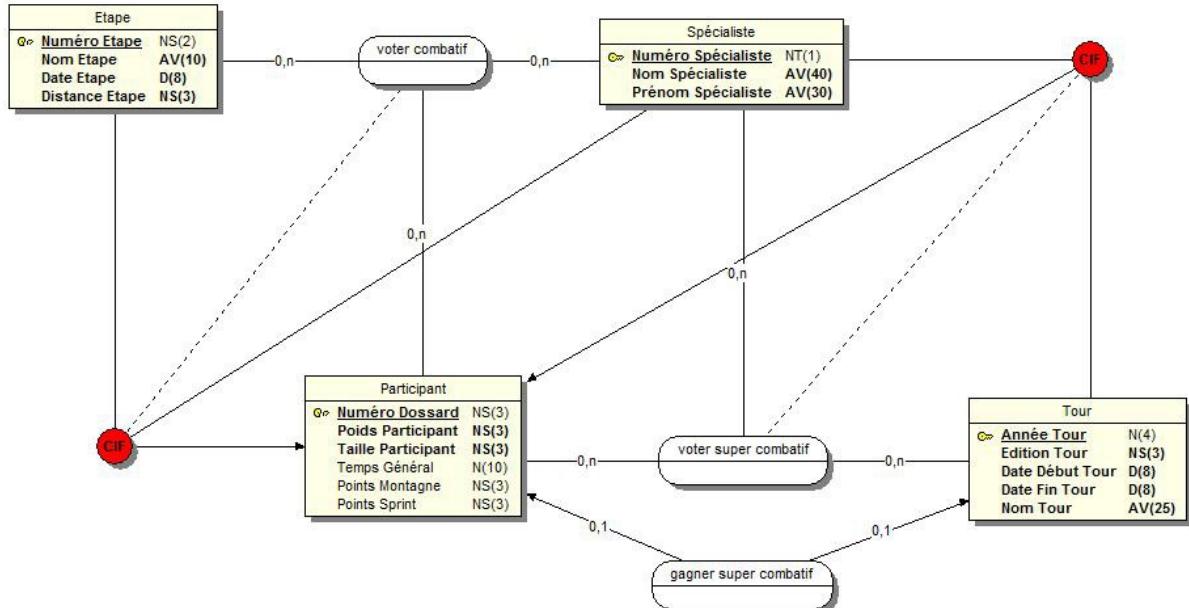


Illustration 23: MEA Gestion du plus combatif

Commentaires

A chaque fin d'étape, un participant reçoit le titre du plus combatif. Ce titre est décerné par un jury. Chaque spécialiste membre du jury vote pour un participant. Un spécialiste est caractérisé par un numéro, un nom et un prénom. Pour un étape et un spécialiste donné, un seul participant peut être nommé.

A la fin du tour, les même spécialistes votent pour décerner le prix du « super-combatif ». Pour un tour et un spécialiste donné, un seul coureur peut être nommé.

5. Gestion des contrôles anti-dopage

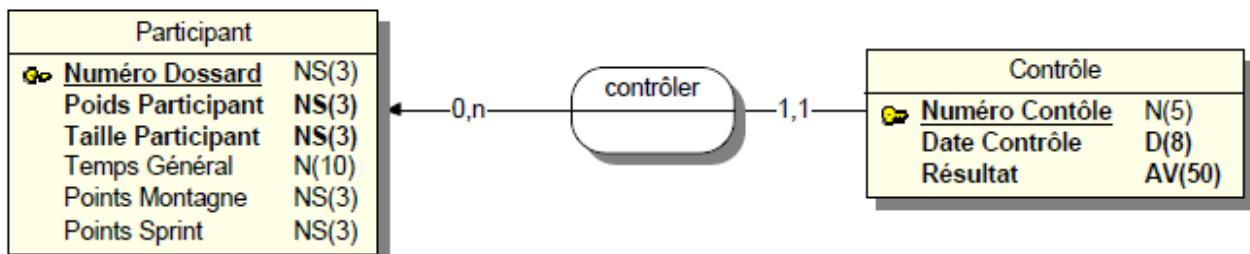


Illustration 24: MEA Gestion contrôles anti-dopage

Commentaires

Les participants du Tour peuvent être soumis à 0, 1 ou plusieurs contrôles anti-dopage. Un contrôle est caractérisé par un numéro unique, une date et un résultat. Un contrôle est associé à un seul participant.

6. Gestion des utilisateurs

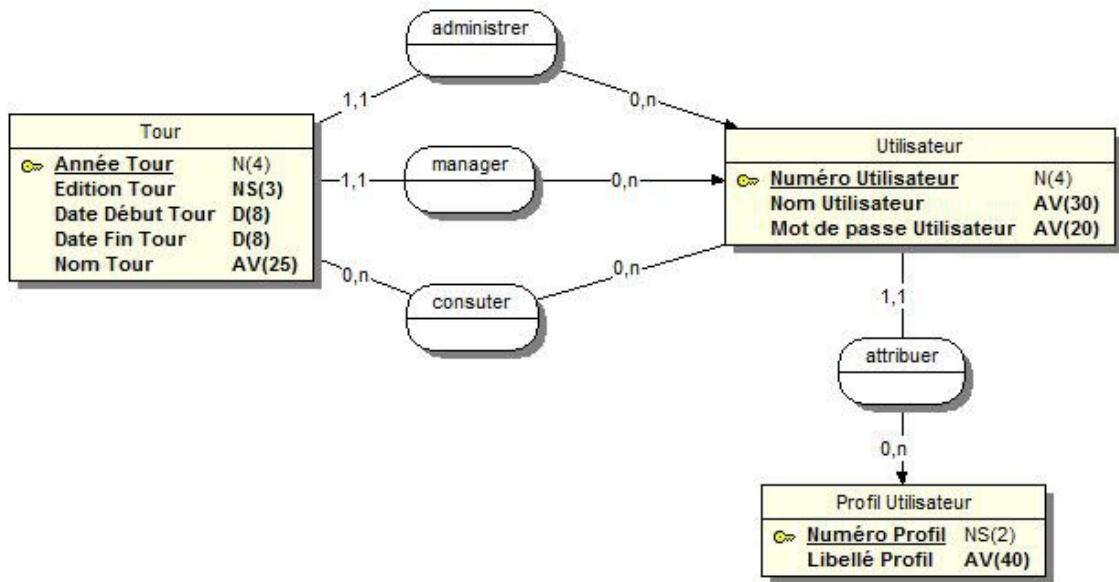


Illustration 25: MEA Gestion des utilisateurs

Commentaires

Un utilisateur se voit attribuer un et un seul profil utilisateur. Un profil utilisateur peut être attribué à 0, 1 ou plusieurs utilisateurs. Un profil utilisateur est caractérisé par un numéro de profil et un libellé. Un utilisateur est caractérisé par un numéro d'utilisateur, un nom et un mot de passe.

Un tour est administré par un et un seul utilisateur. Un utilisateur peut administrer 0, 1 ou plusieurs tours. Un tour est managé par un seul utilisateur (le commissaire de course). Un utilisateur peut être nommé commissaire de 0, 1 ou plusieurs courses. Enfin, un tour est consulté par 0, 1 ou plusieurs utilisateurs. Un utilisateur a les droits de consultation sur 0, 1 ou plusieurs tours.

V. Modèle Logique Relationnel

1. MLR Normalisé

a. Gestion des inscriptions

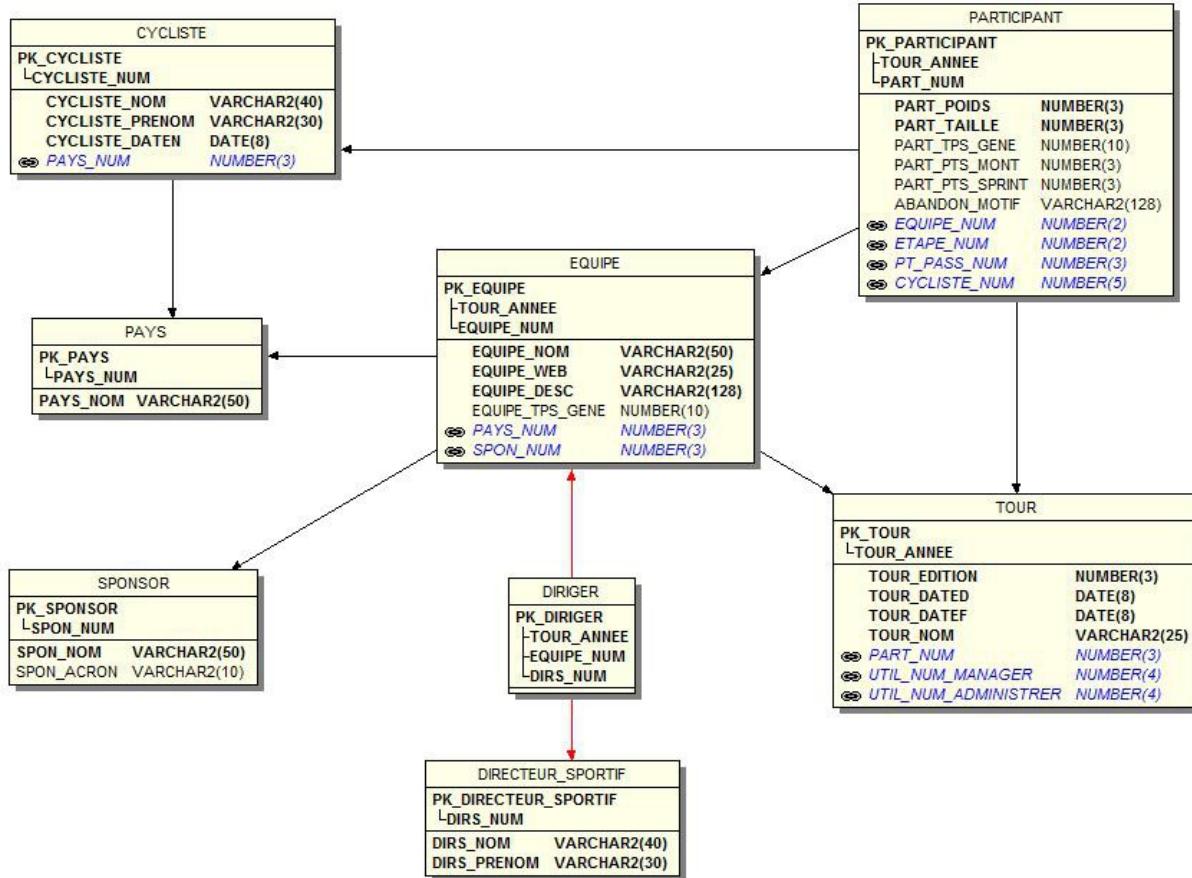


Illustration 26: MLR Normalisé - Gestion des inscriptions

b. Gestion de la course

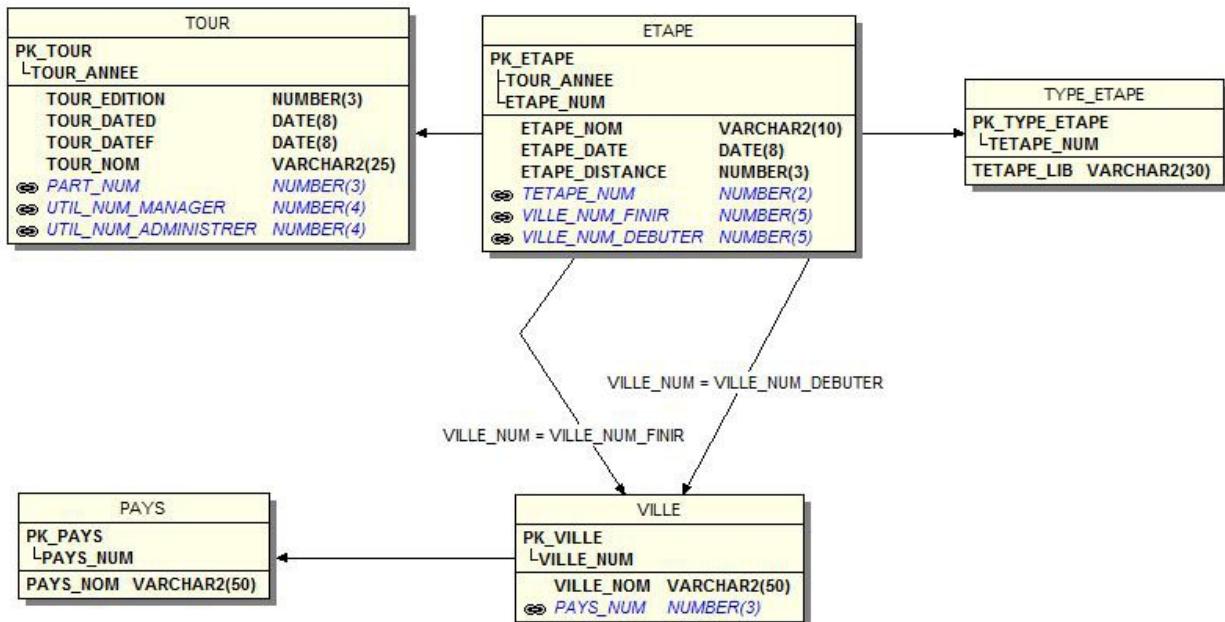


Illustration 27: MLR Normalisé - Gestion de la course

c. Gestion du classement

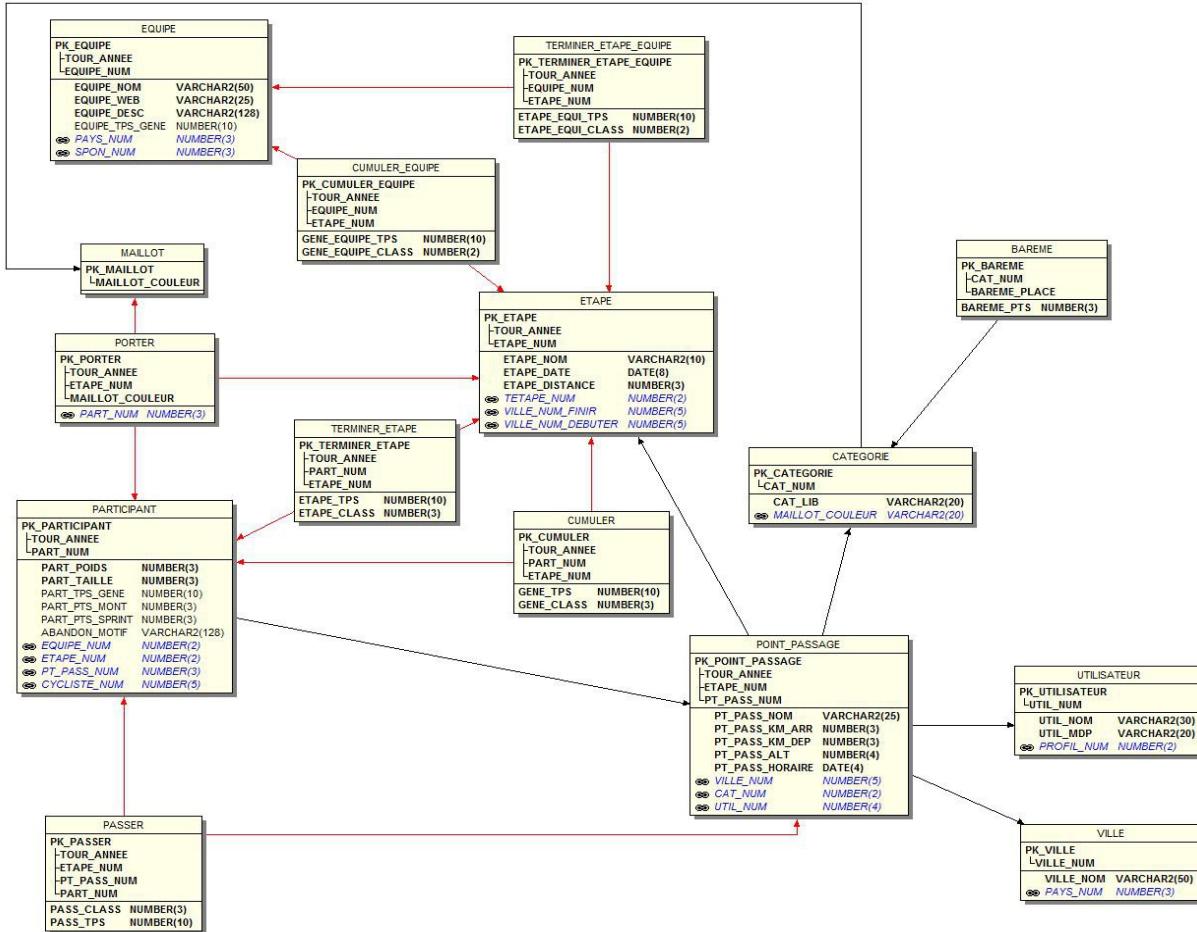


Illustration 28: MLR Normalisé - Gestion du classement

d. Gestion du plus combatif

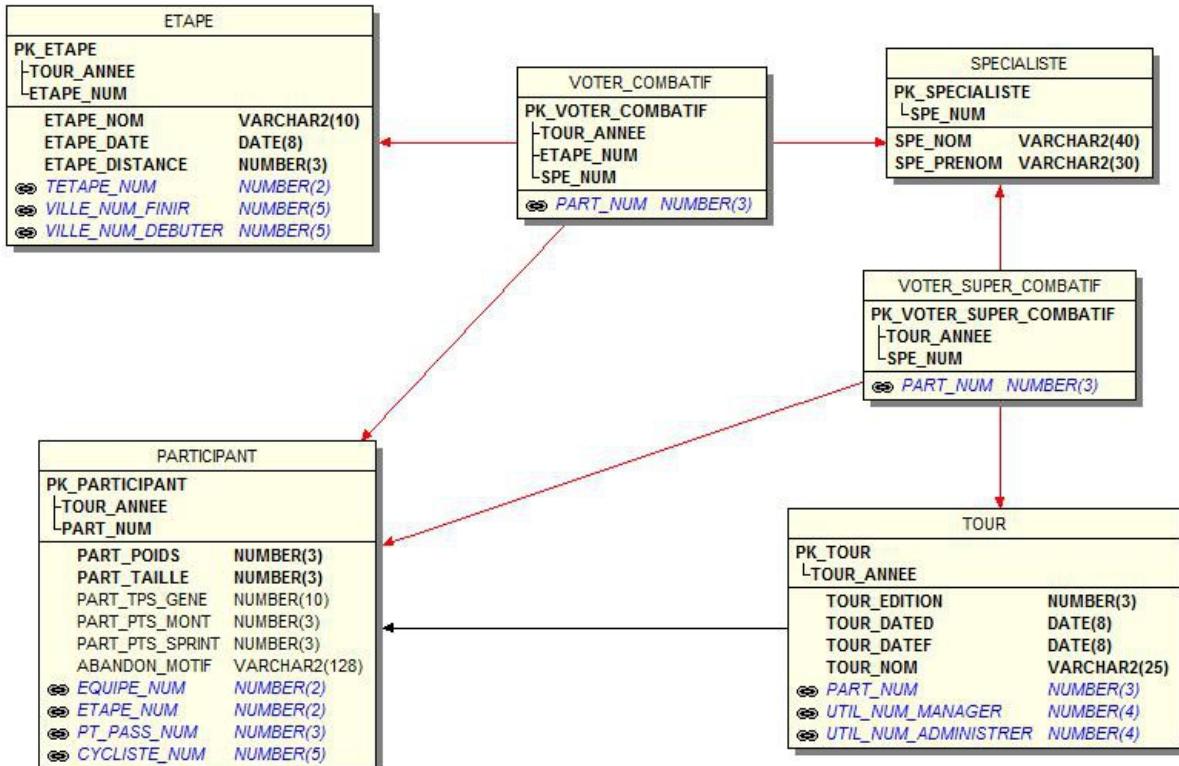


Illustration 29: MLR Normalisé - Gestion combatif

e. Gestion dopage



f. Gestion des utilisateurs

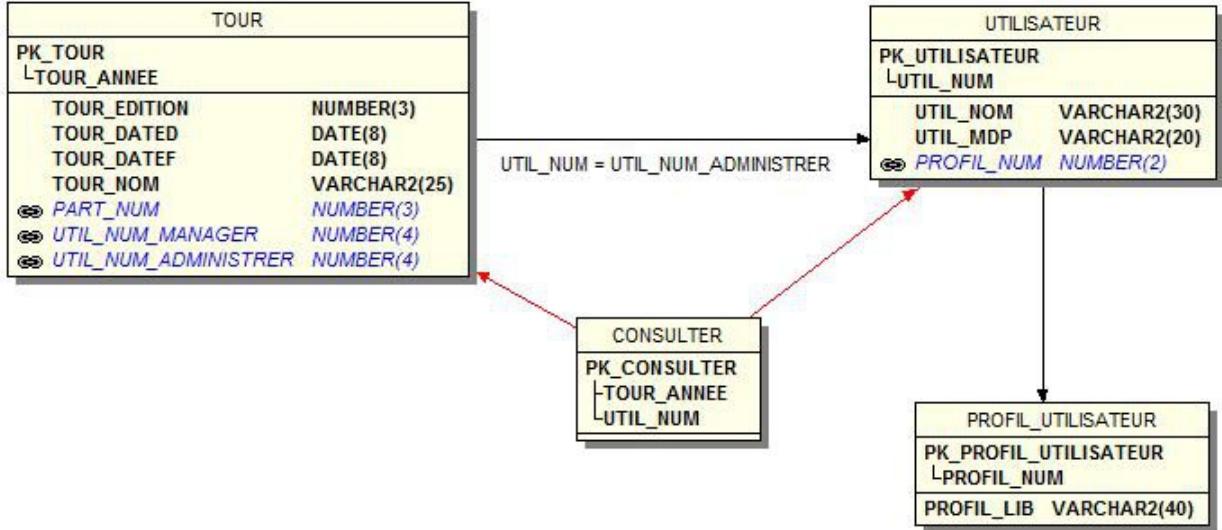


Illustration 30: MLR Normalisé - Gestion des utilisateurs

g. Justification de la normalisation

Le MCD est bien en première forme normale. En effet, pour toutes les tables du modèle les constituants sont fonctionnellement dépendants de la clé c'est à dire que dans un tuple, il ne peut exister plusieurs valeurs pour un même constituant.

Exemple de la table « ETAPE » : Il ne peut y avoir qu'un numéro d'étape, qu'une ville de début, qu'une ville d'arrivée, qu'un nom, qu'une date et qu'une distance pour une étape donnée.

Le MCD est bien en deuxième forme normale étant donné que pour toutes les tables, les constituants qui ne font pas partie de la clé sont en dépendance fonctionnelle élémentaire par rapport à la clé c'est à dire que tout attribut non clé ne dépend pas d'une partie de cette clé.

Exemple de la table « PORTER » : Le numéro du participant dépend bien de l'année du tour, du numéro de l'étape et de la couleur du maillot.

Le MCD est bien en troisième forme normale vu que pour toutes les tables, les constituants non clé sont en dépendance fonctionnelle élémentaire et directe par rapport à la clé c'est à dire que tout attribut non clé ne dépend pas d'un autre attribut non clé.

Exemple de la table « POINT DE PASSAGE » : Le numéro de la ville, le numéro de catégorie, le numéro de l'utilisateur en charge de ce point, le nom, le nombre de kilomètre depuis le départ, le nombre de kilomètre jusqu'à l'arrivée, l'altitude, et l'horaire prévu dépendent bien de la clé primaire.

2. Techniques d'optimisation

a. Fusions de tables

Suppression des tables qui ne comportent qu'une clé primaire

Tout d'abord, il est indispensable de supprimer les tables ne contenant que la clé primaire, car elle n'apporte aucune information supplémentaire au système. En effet, la clé primaire est déjà référencée en temps que clé étrangère pour les tables concernée. Dans notre schéma, c'est la table *MAILLOT* qui va donc disparaître. Bien entendu, les autres tables qui utilisaient la clé primaire (*MAILLOT_COULEUR*) de cette table ne seront plus référencées comme des clé étrangères, sans pour autant être supprimées. Dans notre cas, il s'agit des tables *CATEGORIE* et *PORTER*.

Afin d'assurer l'intégrité des données une contrainte *CHECK_VALUE* sera ajoutée pour les champs *MAILLOT_COULEUR* des différentes tables, afin que le champ ne puisse être renseigné qu'avec des maillots réellement existants

Fusion des tables ayant une clé primaire identique

Même si les données ne sont pas logiquement associées, les contraintes de performances nous obligent à agréger les données de tables ayant la même clé primaire. Dans notre cas, les tables *TERMINER_ETAPE* et *CUMULER* seront regroupées dans une seule et même table *TERMINER_ETAPE*. Pour les mêmes raisons, *TERMINER_ETAPE_EQUIPE* et *CUMULER_EQUIPE* seront regroupées dans la table *TERMINER_ETAPE_EQUIPE*.

b. Dénormalisation

Afin d'améliorer le temps d'accès aux données du système, mais aussi de simplifier l'écriture de requêtes, des tables vont être supprimées. On aura une redondance de données ce qui découlera sur une dénormalisation du modèle.

Il convient de savoir comment le système sera utilisé et les données seront exploitées afin de ne proposer une redondance que pour les données qui présentent un intérêt. Seul certaines données sont à la fois compatibles avec une technique de dénormalisation et entraîneront un réel gain en terme de temps d'accès ou en termes de complexité de recherche.

Suppression de la table SPONSOR

Une équipe possède un seul sponsor. Il est très facile d'intégrer les données de cette table directement dans l'équipe. Pour cela, on ajoutera les champ *SPON_NOM* et *SPON_ACRO* dans la table *EQUIPE*.

Suppression de la table TYPE_ETAPE

Une étape possède un type. Le libellé de ce type pourra être directement intégré à la table *ETAPE* sans trop de problème. Ainsi, une colonne « *TETAPE_LIB* » y sera ajouté. Afin d'assurer l'intégrité des données une contrainte *CHECK_VALUE* sera ajoutée pour le champ *TETAPE_LIB* afin que le champ ne puisse être renseigné qu'avec des types d'étape existants.

Suppression de la table PROFIL_UTILISATEUR

La table PROFIL_UTILISATEUR joue le rôle de table de libellé. Il est donc très facile de supprimer cette table en intégrant le libellé de profil directement dans table UTILISATEUR.

c. Redondance sélective

Dans un but de gain d'optimisation et de simplification des requêtes, un certain nombre de champs vont être redondés, c'est à dire ajoutés dans une table alors même que ces données existent déjà dans d'autres tables. Bien que cela contribue à la dénormalisation du système, l'accès aux différentes données ayant un lien logique fort sera grandement simplifié (moins de jointures). D'un autre côté, appliquer cette redondance aura pour conséquence une augmentation de la taille de la base de données, et va aussi ajouter un risque en ce qui concerne la cohérence des données, puisque les données se trouvant à différents endroits devront être les mêmes ; une modification dans une table devra automatiquement être répercutée sur tous les champs de la base de données dont la sémantique est la même. Afin de gérer la cohérence des données redondées, la mise en place de procédures *PL/SQL* ainsi que de *TRIGGERS* permettront d'assurer un état fonctionnel du système.

Bien évidemment, les données redondées ont été sélectionnées en fonction d'un certain nombre de principes, et correspondent à la façon dont ces données seront utilisées par la partie applicative de notre système. Dans le tableau suivant, les colonnes *Table destination* et *Colonne destination* correspondent respectivement à la table et à la colonne dans lesquelles les données vont être redondées i.e. elles vont être ajoutées, et les colonnes *Table source* et *Colonne source* correspondent à l'endroit où les données se situent dans le modèle normalisé.

Table destination	Colonne destination	Table source	Colonne source
PARTICIPANT	CYCLISTE_NOM	CYCLISTE	CYCLISTE_NOM
	CYCLISTE_PRENOM	CYCLISTE	CYCLISTE_PRENOM
	CYCLISTE_DATEN	CYCLISTE	CYCLISTE_DATEN
	CYCLISTE_PAYS	PAYS	PAYS_NOM
	EQUIPE_NOM	EQUIPE	EQUIPE_NOM
EQUIPE	EQUIPE_NOM	EQUIPE	EQUIPE_NOM
	EQUIPE_PAYS	PAYS	PAYS_NOM
ETAPE	NOM_VILLE_DEBUTER	VILLE	VILLE_NOM
	NOM_VILLE_FINIR	VILLE	VILLE_NOM
TERMINER_ETAPE	CYCLISTE_NOM	CYCLISTE	CYCLISTE_NOM
	CYCLISTE_PRENOM	CYCLISTE	CYCLISTE_PRENOM
	CYCLISTE_DATEN	CYCLISTE	CYCLISTE_DATEN
	CYCLISTE_PAYS	CYCLISTE	CYCLISTE_PAYS
	EQUIPE_NOM	EQUIPE	EQUIPE_NOM
TERMINER_ETAPE_EQUIPE	EQUIPE_NOM	EQUIPE	EQUIPE_NOM
	EQUIPE_PAYS	EQUIPE	EQUIPE_PAYS
POINT_PASSAGE	PT_PASS_VILLE_NOM	VILLE	VILLE_NOM

d. Données calculées

Dans cette partie nous allons définir les données calculées de notre système. En réalité, ces données apparaissent déjà au niveau du MLR normalisé, car elles ont été prises en compte en amont, lors de la réalisation du modèle entité-association. Ces données, bien qu'étant accessibles à travers le système via des calculs, nécessitent la rédaction de requêtes complexes et fastidieuses comprenant elles-mêmes plusieurs jointures, ce qui alourdit le système en terme de performance. Ces informations étant sollicitées fréquemment, cela justifie leur insertion directe dans le système.

A l'instar des données redondantes, les données calculées seront mises à jour au travers de procédures PL/SQL déclenchées par des *TRIGGERS* lors d'événements nécessitant leur modification. Le choix des événements sera déterminé par un compromis entre performances et consistance des données. En effet, mettre à jour trop souvent ces données va être négatif car les accès en écriture seront fréquents, et ne pas les mettre à jour assez souvent produira des résultats obsolètes. L'idéal est de mettre à jour ces données au moment où cela est vraiment nécessaire et lorsque le coût en terme de performances est le plus bas. Par exemple, le nombre de points pour le classement du meilleur grimpeur pour un participant et pour une étape ne sera calculé que lorsque le coureur aura passé la ligne d'arrivée. Ainsi, ce calcul n'aura besoin d'être fait qu'une fois par étape, et non pas chaque fois qu'un participant passe un point de passage, l'inconvénient étant qu'on ne pourra pas observer l'accumulation des points en temps réel lorsqu'une étape est en cours.

Table PARTICIPANT

Champ	Description	Données utilisées	Méthode MAJ	Fréquence
PART_TPS_GENE	Temps général d'un participant (en dixièmes de secondes)	TERMINER_ETAPE (GENE_TPS)	TRIGGER	Lorsque TERMINER_ETAPE (GENE_TPS) est mis à jour
PART_CLASS_GENE	Classement général d'un participant	PARTICIPANT (PART_TPS_GENE)	TRIGGER	Lorsque PARTICIPANT (PART_TPS_GENE) est mis à jour
PART_TPS_ECART	Écart entre le meilleur temps et le temps du participant	PARTICIPANT (PART_TPS_GENE)	TRIGGER	Lorsque PARTICIPANT (PART_TPS_GENE) est mis à jour
PART PTS MONT	Total des points pour le classement du meilleur grimpeur	TERMINER_ETAPE (ETAPE PTS MONT)	TRIGGER	Lorsque TERMINER_ETAPE (ETAPE PTS MONT) est mis à jour
PART PTS SPRINT	Total des points pour le classement du meilleur sprinteur	TERMINER_ETAPE (ETAPE PTS SPRINT)	TRIGGER	Lorsque TERMINER_ETAPE (ETAPE PTS SPRINT) est mis à jour
PART_CLASS_MONT	Rang courant pour le classement du meilleur grimpeur	PARTICIPANT (PART PTS MONT) de tous les participants	TRIGGER	Lorsque PARTICIPANT (PART PTS MONT) est mis à jour
PART_CLASS_SPRINT	Rang courant pour le classement du meilleur sprinteur	PARTICIPANT (PART PTS SPRINT) de tous les participants	TRIGGER	Lorsque PARTICIPANT (PART PTS SPRINT) est mis à jour

Table EQUIPE

Champ	Description	Données utilisées	Méthode MAJ	Fréquence
EQUIPE_TPS_GENE	Temps général d'une équipe (en dixièmes de secondes)	TERMINER_ETAPE_EQUIPE (GENE_EQUIPE_TPS)	TRIGGER	Lorsque TERMINER_ETAPE_EQUIPE (GENE_EQUIPE_TPS) est mis à jour
EQUIPE_CLASS_GEN_E	Classement général d'une équipe	TERMINER_ETAPE_EQUIPE (GENE_EQUIPE_TPS)	TRIGGER	Lorsque TERMINER_ETAPE_EQUIPE (GENE_EQUIPE_CLASS) est mis à jour

Table TERMINER_ETAPE

Champ	Description	Données utilisées	Méthode MAJ	Fréquence
ETAPE_TPS	Temps du participant pour l'étape	PASSER(PASS_TPS)	TRIGGER	Lorsque le participant passe le dernier point de passage de l'étape
ETAPE_CLASS	Classement du participant pour l'étape	PASSER(PASS_TPS) de l'ensemble des participants pour l'étape	TRIGGER	Lorsque le participant passe le dernier point de passage de l'étape
ETAPE PTS MONT	Nombre de points pour le classement du meilleur grimpeur engrangés pendant l'étape	PASSER(PASS_CLASS), POINT_PASSAGE(CAT_NUM), BARÈME(BAREME PTS)	TRIGGER	Lorsque le participant passe le dernier point de passage de l'étape
ETAPE PTS SPRINT	Nombre de points pour le classement du meilleur sprinteur engrangés pendant l'étape	PASSER(PASS_CLASS), POINT_PASSAGE(CAT_NUM), BARÈME(BAREME PTS)	TRIGGER	Lorsque le participant passe le dernier point de passage de l'étape
GENE_TPS	Temps général d'un participant à la fin de l'étape	PASSER(PASS_TPS)	TRIGGER	Lorsque le participant passe le dernier point de passage de l'étape
GENE_CLASS	Temps général d'un participant à la fin de l'étape	TERMINER_ETAPE(GENE_TPS) pour tous les participants	TRIGGER	Lorsque TERMINER_ETAPE (GENE_TPS) est mis à jour
GENE PTS MONT	Rang général du participant pour le classement du meilleur grimpeur à l'issue de l'étape	TERMINER_ETAPE(ETAPE PTS MONT)	TRIGGER	Lorsque TERMINER_ETAPE (ETAPE PTS MONT) est mis à jour
GENE CLASS MONT	Rang du participant pour le classement du meilleur grimpeur à l'issue de l'étape	TERMINER_ETAPE(GENE PTS MONT)	TRIGGER	Lorsque TERMINER_ETAPE (GENE PTS MONT) est mis à jour
GENE PTS SPRINT	Total des points pour le classement du meilleur sprinteur à l'issue de l'étape	TERMINER_ETAPE(ETAPE PTS SPRINT)	TRIGGER	Lorsque TERMINER_ETAPE (ETAPE PTS SPRINT) est mis à jour
GENE CLASS SPRINT	Rang général du participant pour le classement du meilleur sprinteur à l'issue de l'étape	TERMINER_ETAPE(GENE PTS SPRINT) de	TRIGGER	Lorsque TERMINER_ETAPE (GENE PTS SPRINT) est mis à jour

Champ	Description	Données utilisées	Méthode MAJ	Fréquence
	l'issue de l'étape	l'ensemble des participants		

Table TERMINER_ETAPE_EQUIPE

Champ	Description	Données utilisées	Méthode MAJ	Fréquence
ETAPE_EQUI_TPS	Temps de l'équipe pour l'étape	TERMINER_ETAPE (ETAPE_TPS)	TRIGGER	Lorsque TERMINER_ETAPE (ETAPE_TPS) est mis à jour
ETAPE_EQUI_CLASS	Classement de l'équipe pour l'étape	TERMINER_ETAPE_EQUIPE (ETAPE_EQUI_TPS) de l'ensemble des participants pour l'étape	TRIGGER	Lorsque TERMINER_ETAPE_EQUIPE (ETAPE_EQUI_TPS) est mis à jour
GENE_EQUI_TPS	Temps total de l'équipe à l'issue de l'étape	TERMINER_ETAPE_EQUIPE (ETAPE_EQUI_TPS)	TRIGGER	Lorsque TERMINER_ETAPE_EQUIPE (ETAPE_EQUI_TPS) est mis à jour
GENE_EQUI_CLASS	Classement général de l'équipe à l'issue de l'étape	TERMINER_ETAPE_EQUIPE (GENE_EQUI_TPS) de toutes les équipes	TRIGGER	Lorsque TERMINER_ETAPE_EQUIPE (GENE_EQUI_TPS) est mis à jour

Table PASSER

Champ	Description	Données utilisées	Méthode MAJ	Fréquence
PASS_CLASS	Classement au point de passage	PASSSER (PASS_TPS) de tous les participants	TRIGGER	Lorsque PASSER (PASS_TPS) est mis à jour

Table POINT_PASSAGE

Champ	Description	Données utilisées	Méthode MAJ	Fréquence
PT_PASS_KM_ARR	Distance entre le point de passage et l'arrivée	POINT_PASSAGE (POINT_PASS_KM_DEP)	TRIGGER	Lorsque TERMINER_ETAPE_EQUIPE (GENE_EQUI_TPS) est mis à jour

VI. Optimisation spécifique Oracle

1. Stratégie d'indexation

Afin d'améliorer les performances, en particulier dans le cadre de critères de recherche, de tri, de regroupement, et de jointure, ORACLE, à l'instar de la plupart des SGBD du marché, permet la création d'index. Afin de réellement gagner en performance, la table sur laquelle l'index va être appliqué doit être relativement conséquente concernant son nombre d'enregistrements, et le champ indexé doit évidemment être minutieusement choisi.

L'inconvénient de l'indexation est l'augmentation du volume de la base, puisque les index sont physiquement stockés. De plus, il faudra prendre en compte le fait que les index doivent être reconstruits à chaque modification des données. La reconstruction d'index n'est pas négligeable en terme de temps et de performance lorsqu'il s'agit d'un volume de données conséquent et que la fréquence de reconstruction est importante. Lorsqu'une série de mise à jour est planifiée, on pourra proposer de supprimer temporairement les index avant la mise à jour, puis de les rétablir à la fin la procédure ; ainsi l'index ne sera reconstruit qu'une fois pour le lot.

a. Indexation sur les clés étrangères

Si un index est automatiquement construit sur les clés primaires, ce n'est pas le cas pour les clés étrangères. Afin d'accélérer les jointures, un index sera créé sur les champs jouant le rôle de clé étrangère. Un gain en performance sera ainsi rendu possible, en particulier dans le cadre de requêtes complexes faisant intervenir un nombre de jointure relativement important.

Table	Colonne
CYCLISTE	PAYS_NUM
PARTICIPANT	CYCLISTE_NUM
	EQUIPE_NUM
	TOUR_ANNEE, ETAPE_NUM
	TOUR_ANNE, ETAPE_NUM, PT_PASS_NUM
EQUIPE	PAYS_NOM
TOUR	PART_NUM
	UTIL_NUM_MANAGER
	UTIL_NUM_ADMINISTRER
ETAPE	VILLE_NUM_DEBUTER
	VILLE_NUM_FINIR
POINT_PASSAGE	VILLE_NUM
	CAT_NUM
	UTIL_NUM
VILLE	PAYS_NUM

Table	Colonne
PORTR	PART_NUM
VOTER_COMBATIF	PART_NUM
VOTER_SUPER_COMBATIF	PART_NUM
CONTROLE	TOUR_ANNEE
	PART_NUM

b. Indexation sur les données utilisées pour des critères de recherche

Dans le cadre de notre application, il paraît évident que les noms des coureurs et les noms des équipes ainsi que le noms de villes seront fréquemment utilisés dans le cadre de recherche. Il sera donc nécessaire d'indexer les champs correspondants. Pour les données relatives au nom et au prénom, du cycliste on va proposer un index composé sur le nom et le prénom. En admettant que le critère de recherche principal reste le nom, ceci déterminera l'ordre de la composition ; l'index sera donc composé du nom, puis du prénom. Enfin, sachant que le classement du maillot blanc se calcule grâce à l'âge du participant, il faudra aussi mettre un index sur le champs renseignant la date de naissance du cycliste, dans la table *PARTICIPANT* (où la donnée sera prélevée pour effectuer le calcul).

Table	Colonne
CYCLISTE	CYCLISTE_NOM, CYCLISTE_PRENOM
PARTICIPANT	CYCLISTE_NOM, CYCLISTE_PRENOM
	CYCLISTE_DATEN
	EQUIPE_NOM
EQUIPE	EQUIPE_NOM
VILLE	VILLE_NOM
ETAPE	ETAPE_NOM
TERMINER_ETAPE	CYCLISTE_DATEN

c. Indexation sur les classements

Les classements constituent une information essentielle dans une course, et il paraît donc judicieux de les indexer. Par exemple, les différents classements courants et de fin d'étape n'étant modifiés qu'à la fin d'une étape, ils semblent être de bon candidats à l'indexation dans le sens où la fréquence de reconstruction des index ne sera pas trop élevée. De plus, ce sont ces champs qui vont être fortement sollicités afin de visualiser le classement général.

Table	Colonne
PARTICIPANT	PART_CLASS_GENE
	PART_CLASS_MONT
	PART_CLASS_SPRINT
EQUIPE	EQUIPE_CLASS_GENE
TERMINER_ETAPE	ETAPE_CLASS
	GENE_CLASS

Table	Colonne
	GENE_CLASS_MONT
	GENE_CLASS_SPRINT
TERMINER_ETAPE_EQUIPE	ETAPE_EQUI_CLASS
	GENE_EQUI_CLASS

d. Table organisées en index (IOT)

Une table standard contient des données stockées sans ordre particulier. Dans une table organisée en index, les données sont stockées dans une structure de type B-Tree, triée sur la clé primaire. Ce type d'organisation convient parfaitement pour les requêtes dont les termes sont basés exclusivement sur les éléments de la clé primaire. On pense plus particulièrement aux tables faisant le lien entre plusieurs tables et ayant une clé primaire composée de clé étrangères. Ce sont typiquement les tables qui correspondaient à des associations avec une cardinalité 0,n des deux côtés dans le modèle conceptuel de données

Tables organisées en index (IOT)
PORTEUR
PASSER
TERMINER_ETAPE
TERMINER_ETAPE_EQUIPE
DIRIGER
VOTER_COMBATIF
VOTER_SUPER_COMBATIF
CONSULTER

2. Stratégie de partitionnement

Afin d'optimiser notre système, une stratégie de partitionnement est plus que conseillée. Là encore, une sélection méthodique des tables qui seront partitionnées ainsi que la technique de partitionnement qui leurs sera appliquée doivent être réfléchis. Ces choix sont bien évidemment influencés par la manière dont les tables seront accédées, et le nombre d'enregistrement prévu pour les tables. En effet, le partitionnement ne présente un intérêt que si le nombre d'enregistrements est plutôt conséquent.

Dans notre cas, il est plutôt rare que les données relatives à des tours différents soient utilisées pour une même requête. La clé de partitionnement la plus naturelle est donc l'identifiant de Tour. De plus, cet attribut est présent dans toutes les tables requérant un partitionnement, étant donné qu'il est utilisé pour l'identification relatives de beaucoup d'enregistrements. Le partitionnement n'en sera que facilité ; on évitera le partitionnement par référence au profit d'un partitionnement par plage, beaucoup moins contraignant et beaucoup plus simple à mettre en place.

Toutefois, le partitionnement pour les tables *EQUIPE*, *PARTICIPANT* et *ETAPE*, *TERMINER_ETAPE_EQUIPE* bien que pouvant être de bon candidats au partitionnement en raison du faible lien logique entre les données de différents tour, ne seront pas partitionnées en raison du faible nombre d'enregistrements. Par exemple, en partitionnant la table *PARTICIPANT* par Tour, nous nous retrouverions avec des partitions avec environ 200 enregistrements (le nombre de

participants pour un tour), ce qui est dérisoire et ne présente donc aucun bénéfice, en terme d'optimisation des performances du moins.

a. Table TERMINER_ETAPE

Cette table est importante pour notre système. De plus le nombre d'enregistrements sera probablement relativement important. Par exemple, si le nombre de participants est de 200 et que le nombre d'étapes est de 23, le nombre d'enregistrements sera de 4600, et ce uniquement pour un Tour. Même si ce nombre n'est pas énorme, on imagine bien que si on saisi les données de tous les Tours précédent depuis sa création, on risque de terminer avec un nombre d'enregistrements assez important.

Nous opterons donc pour un partitionnement par plage sur l'identifiant de Tour, i.e tous les enregistrements liés au même tour appartiendront à la même partition. La clé de partitionnement sera donc *TOUR_ANNEE*.

b. Table POINT_PASSAGE

Dans notre système un niveau de granularité plutôt fin est souhaité pour le parcours d'une étape. Le nombre de points de passage sera de l'ordre de la centaine par étape. Avec 100 points de passage et 23 étapes, nous aurons donc 4600 point de passage par Tour. Tours, la table présentera donc un nombre important d'enregistrements.

Il convient donc de partitionner cette table par plage en associant tous les points de passage d'une même année à une même partition. La clé de partitionnement sera donc *TOUR_ANNEE*.

c. Table PASSER

C'est certainement la table qui possédera le plus grand nombre d'enregistrement. En effet, si une étape est composée de 100 points de passage et le nombre de participants pour le tour considéré s'élève à 200, le nombre d'enregistrement pour une seule étape s'élèvera logiquement au produit cartésien, c'est à dire à 20 000. Pour obtenir le nombre d'enregistrements par tour, il faudrait multiplier ce nombre par le nombre d'étapes. On pourrait ainsi avoir plus de 400 000 enregistrements, uniquement pour une année ! Au vu de ces informations, partitionner cette table n'est pas seulement préférable, c'est indispensable.

Comme pour les autres tables, nous allons opter pour un partitionnement par plage. Là encore, toutes les données relatives à un même Tour seront stockées dans la même partition. La clé de partitionnement utilisée sera *TOUR_ANNEE*.

d. Partitionnement d'index

Dans un but d'optimisation de performances et de simplicité dans leur mise en oeuvre, les tables partitionnées verront leurs index suivre un équipartitionnement. En effet, l'équipartitionnement est préconisé lorsque les conditions le permettent, en particulier sur les clés primaires et les clés étrangères.

Ainsi, les index seront partitionnés de la même façon que les leur table associée. Autrement dit, pour une partition d'index sera liée à une et une seule partition de la table.

3. MLR Optimisé

a. Gestion des inscriptions

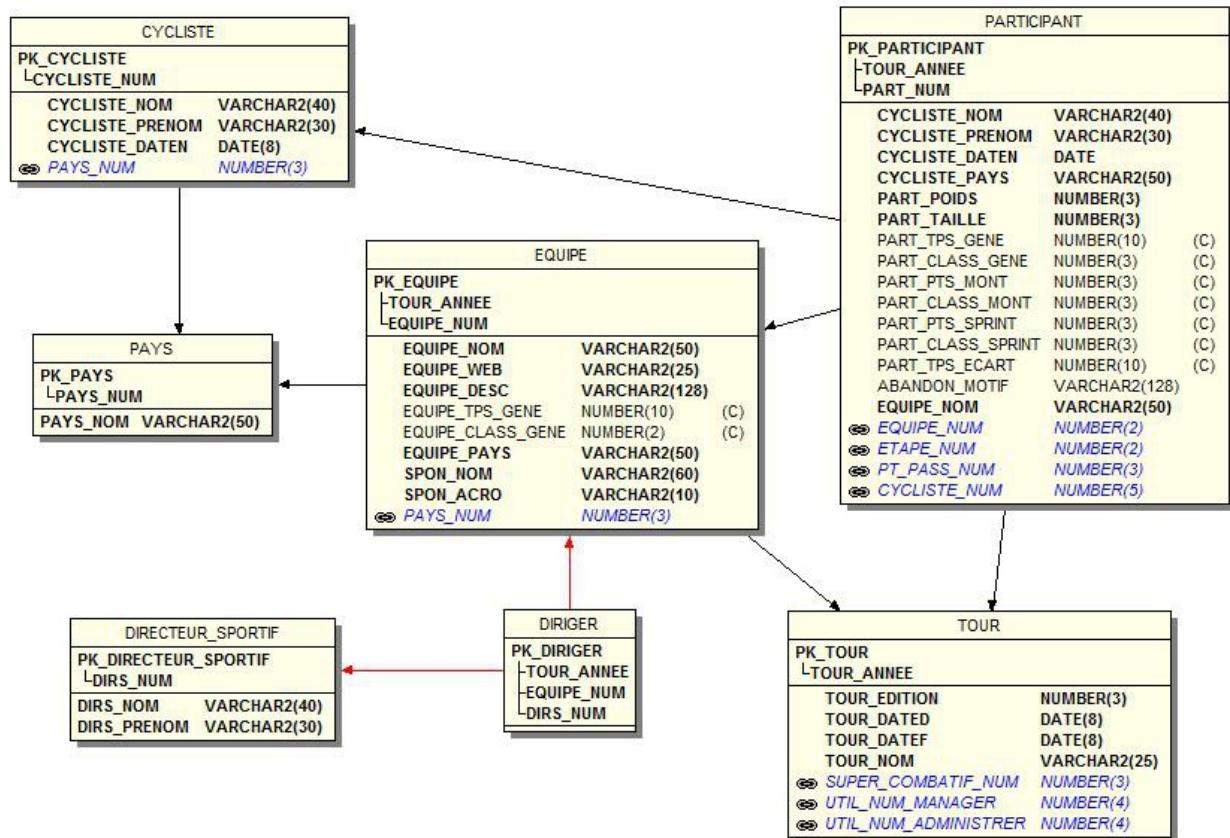


Illustration 31: MLR Optimisé - Gestion des inscriptions

b. Gestion de la course

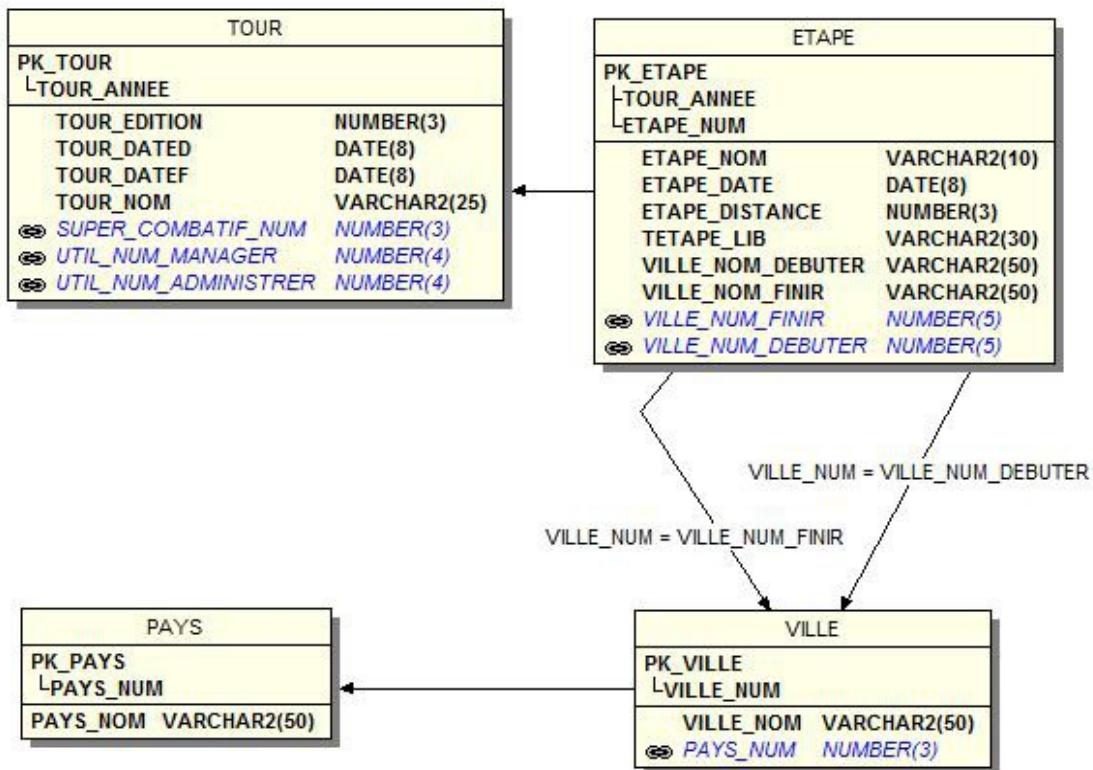


Illustration 32: MLR Optimisé - Gestion de la course

c. Gestion du classement

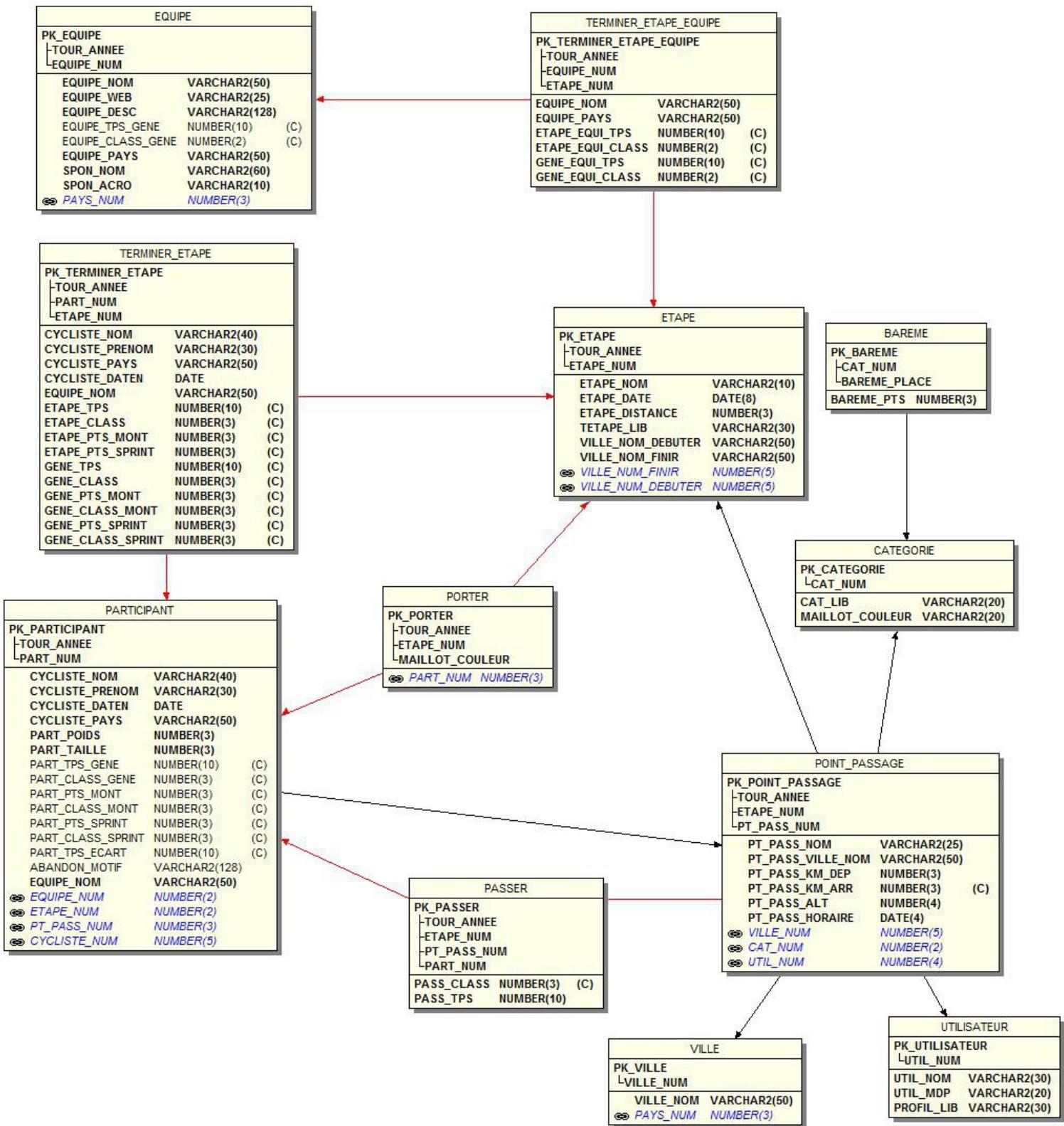


Illustration 33: MLR Optimisé - Gestion du classement

d. Gestion des utilisateurs

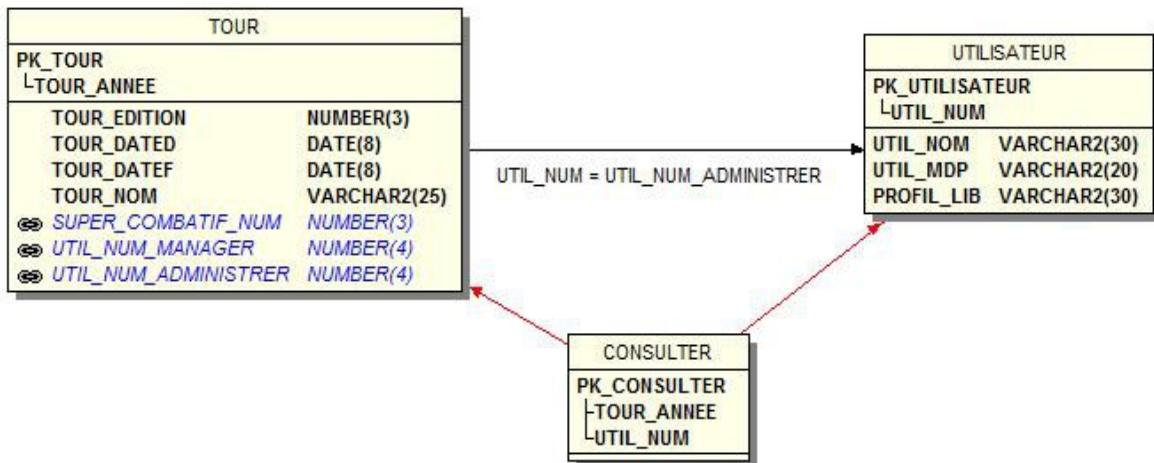


Illustration 34: MLR Optimisé - Gestion des utilisateurs

e. Gestion dopage

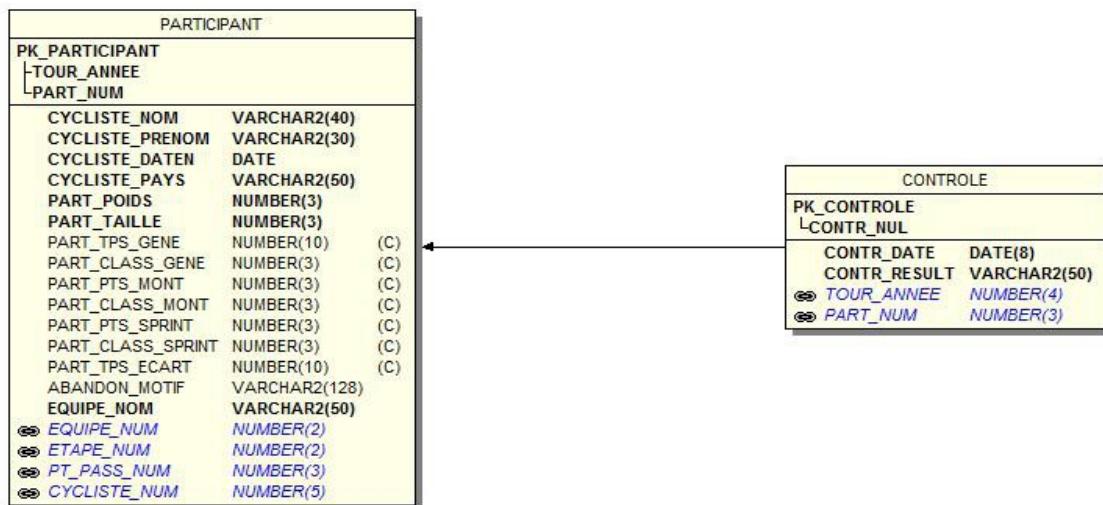


Illustration 35: MLR Optimisé - Gestion dopage

f. Gestion combatif

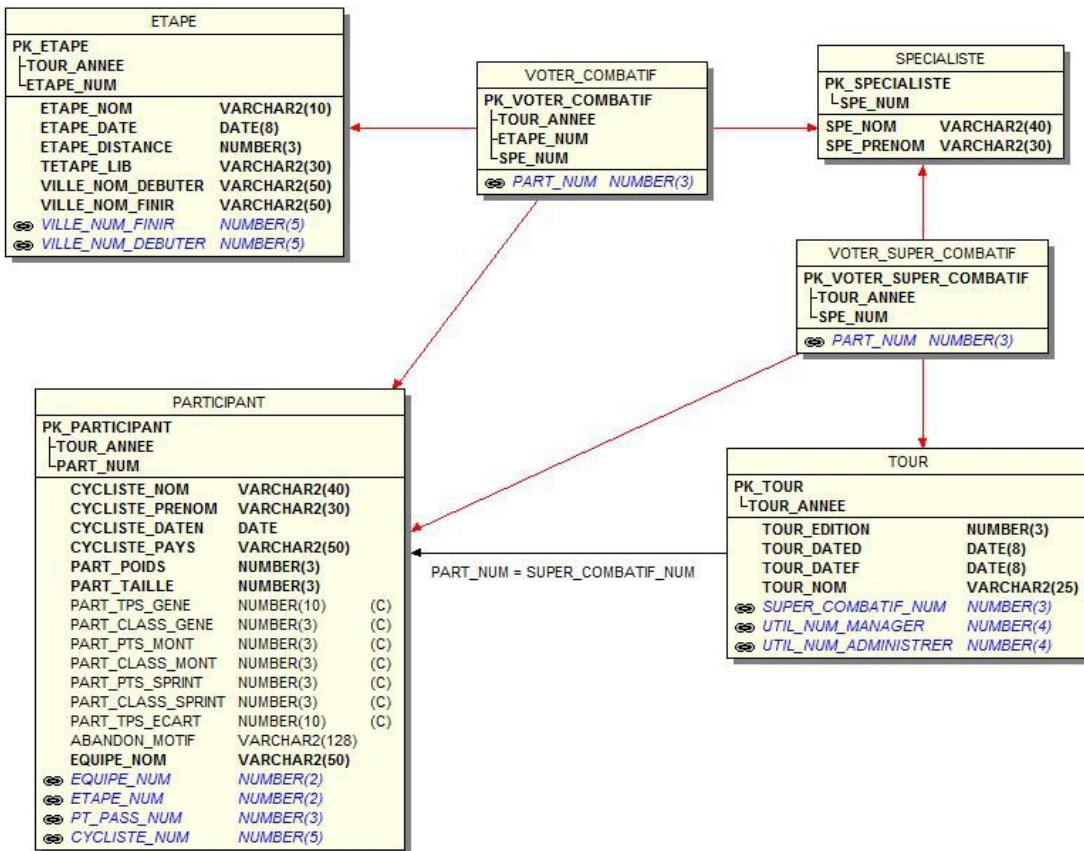


Illustration 36: MLR Optimisé - Gestion combatif

VII. Architecture applicative

1. Architecture technique

La structure générale de notre système d'information consiste en une architecture trois tiers (trois niveaux) :

- **Couche présentation de données** : permet l'affichage sur le navigateur web et la communication avec l'utilisateur.
- **Couche Traitement métier des données** : correspond à la partie fonctionnelle de l'application. Elle implémente la logique du SI et décrit les opérations que l'application opère sur les données en fonction des requêtes des utilisateurs, effectuées au travers de la couche présentation. Dans le cadre de notre système, ces tâches sont implémentées par le serveur d'application Oracle AS.
- **Couche accès aux données** : définit la partie gérant l'accès à la base de données. Dans notre cas, système de gestion de base de données relationnel (SGBDR) Oracle remplira cette fonction.

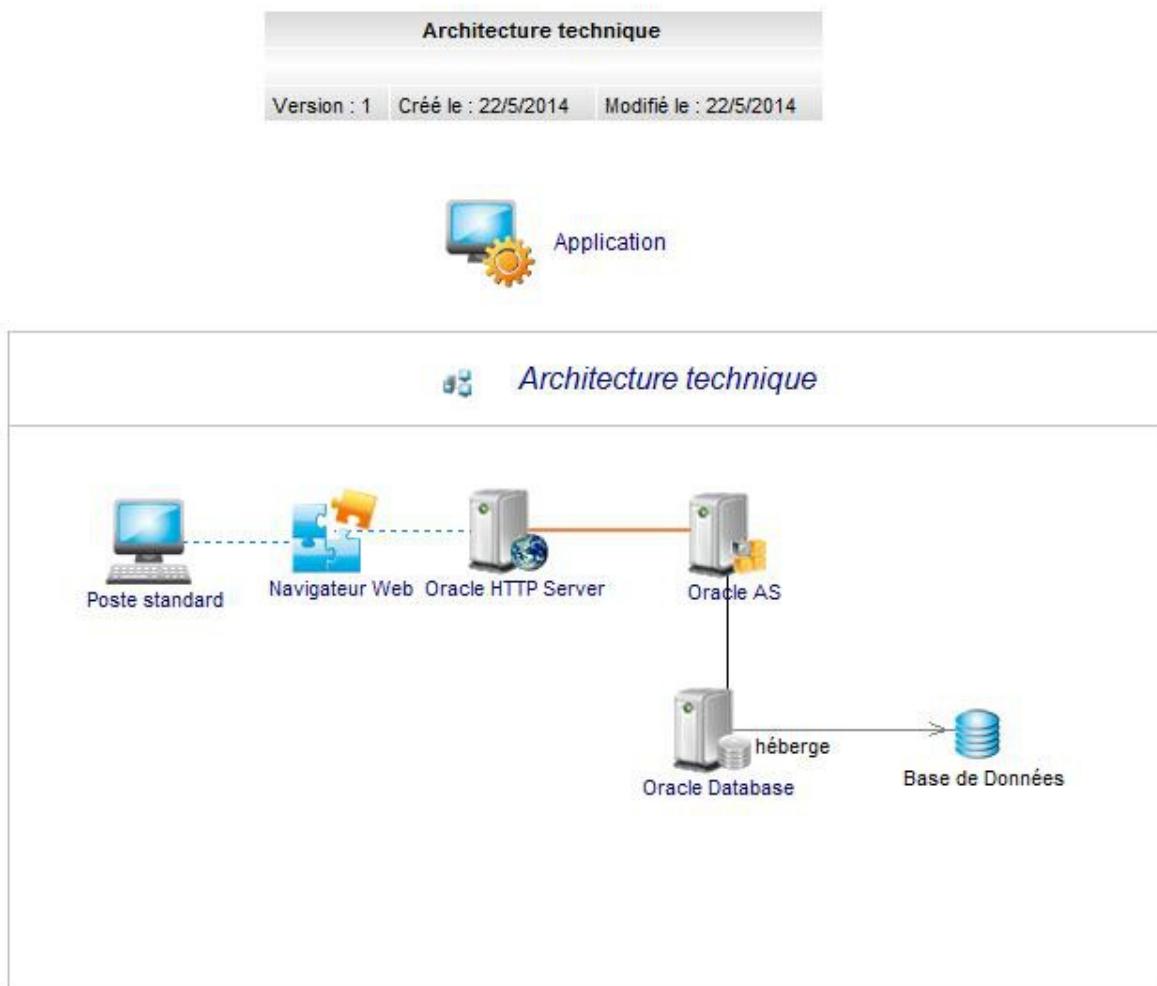


Illustration 37: Architecture technique

2. Cartographie fonctionnelle par module

Notre système d'information est décomposé en plusieurs modules, chacun assurant des fonctionnalités sémantiquement liées. Cette architecture offre une meilleure lisibilité et assure une meilleure évolutivité.

Au sein de notre logiciel, les différents modules seront :

- **Module résultats** : Module permettant de gérer et de visualiser les résultats de la course. C'est à travers ce module que le commissaire de point de passage saisira les résultats et que le commissaire de course les validera.
- **Module accès** : Utiliser pour gérer les accès des différents utilisateurs au système, notamment à l'aide de du profil utilisateur qui définit les rôles et les droits.
- **Module inscriptions** : Utiliser pour l'inscription des coureurs et des équipes à la course. Si nécessaire, les coureurs et les équipes seront d'abord ajoutées dans la base de données.
- **Module dopage** : Module utilisé pour la gestion des contrôles anti-dopage.
- **Module vote combatif** : Module utilisé pour le vote du combatif et du super-combatif. C'est à l'aide de ce module que les spécialistes feront connaître leur décision, qui seront ensuite

validées par le commissaire de course.

- **Module gestion de parcours :** Module utilisé pour la partie administrative de la course, c'est à dire la création de l'édition et du parcours. C'est grâce à ce module que les tours, les étapes et les points de passages seront ajoutées au système.

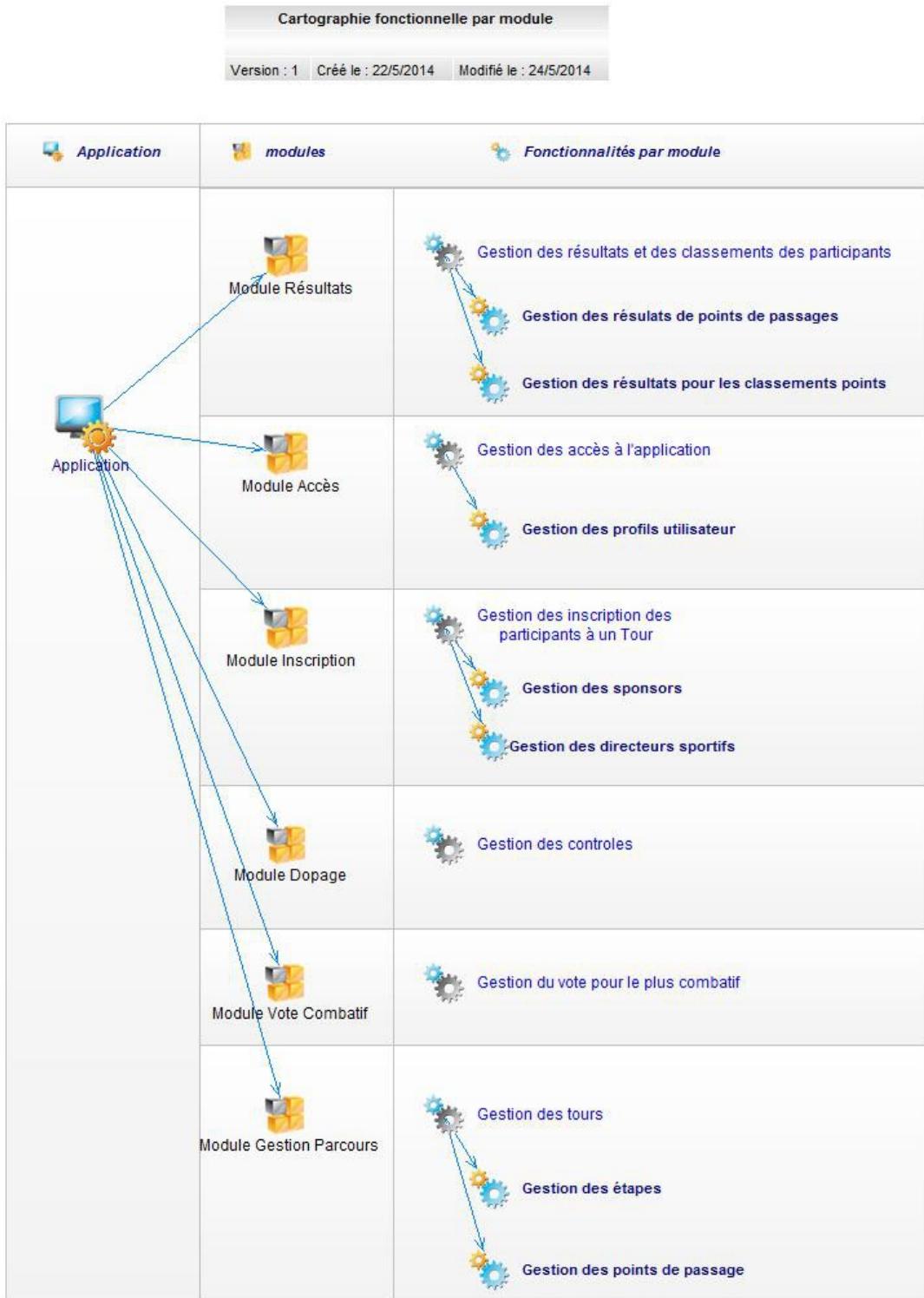


Illustration 38: Cartographie fonctionnelle par module

Partie 3 Développement

Dans cette partie, nous aborderons le développement de la solution proposée et son déploiement pour disposer d'un système fonctionnel. Étant donné le nombre de travaux conséquent réalisé dans cette partie, il est impossible d'intégrer tout ce qui a été fait dans ce rapport. C'est pourquoi une vue d'ensemble a été préférée, qui sera elle-même illustrée par quelques exemples jugés pertinents

I. Organisation des scripts SQL

II. Automatisation

Dans un but de simplifier la vie des utilisateurs de l'application, et plus particulièrement du côté administration, un grand nombre de tâches ont été automatisées. Si le développement de ces fonctionnalités a été particulièrement complexe et fastidieux, celles-ci apportent une vraie valeur ajoutée à l'application. Plutôt que rentrer en profondeur dans le fonctionnement en détail de chacune des fonctionnalités développées, l'architecture globale et la philosophie générale vont être présentées.

Concrètement, on peut dire que deux types d'automatisation ont été effectuées:

- Lorsque les données sont déjà présentes à un autre endroit de la base, elles sont automatiquement rapatriées. Ainsi l'utilisateur n'a pas besoin de ressaisir des données qui seraient déjà présentes. Cela évite du même coup certaines incohérences si l'utilisateur venait à saisir des informations différentes. Ainsi, les données redondées sont gérées de façon transparente pour l'utilisateur qui n'a besoin que de saisir un nombre minimal d'informations.
- Lorsque les données sont calculables à partir d'autres informations, elles sont automatiquement calculées par le système. Cela allège la partie administrative des utilisateurs et évite les erreurs humaines. Ainsi, toutes les informations relatives aux temps, aux points ou aux classements sont gérés par le système en fonction des résultats intermédiaires de chacun des coureurs.

On présente ci-dessous un descriptif des différents développements qui aboutissent à cette automatisation. Le contenu de tous les scripts est disponible à travers les fichiers fournis avec ce document. Il est importante de préciser que certains de ces développements sont parfois attachés aux deux types d'automatisation énumérés ci-dessus.

Nom	Type	Fonction
TI_EQUIPE	TRIGGER (à l'insertion d'une nouvelle équipe)	Récupère les informations relatives aux pays et donne un numéro d'équipe automatiquement en fonction du nombre d'équipes déjà inscrites.
TI_PARTICIPANT	TRIGGER (à l'insertion d'un nouveau participant)	Récupère les informations redondées par rapport à la table CYCLISTE et attribue un numéro de participant automatiquement en fonction de l'équipe et du nombre de participants déjà inscrits pour cette même équipe.
TI_ETAPE	TRIGGER (à l'insertion d'une nouvelle étape)	Récupère les informations relatives aux villes et attribue automatiquement le numéro d'étape. Cela met également à

Nom	Type	Fonction
		jour l'entité <i>TOUR</i> en modifiant sa date de début et sa date de fin en fonction des dates des étapes.
TI_POINT_PASSAGE	TRIGGER (à l'insertion d'un point de passage)	Récupère les informations relative à la ville si besoin et attribue un numéro de point de passage automatiquement. Met également à jour la distance de l'étape et la distance à parcourir jusqu'à l'arrivée pour chacun des points de passage associées à l'étape.
TI_TERMINER_ETAPE	TRIGGER (à l'insertion d'un résultat d'étape pour un participant)	Récupère simplement les données redondées de la table <i>CYCLISTE</i> .
TI_TERMINER_ETAPE_EQUIPE	TRIGGER (à l'insertion d'un résultat d'étape pour une équipe)	Récupère simplement les données redondées de la table <i>EQUIPE</i> .
TI_PASSER_BEFORE TI_PASSER_AFTER	TRIGGER (à l'insertion d'un résultat d'un participant pour un point de passage dans <i>PASSER</i>)	Constitue l'élément central pour la mise à jour automatique des résultats de la course. Ces deux scripts mettent à jour les informations dans <i>TERMINER_ETAPE</i> et <i>TERMINER_ETAPE_EQUIPE</i> tout au long de l'étape. Il calcule les points reçus pour chaque participant en fonction de la catégorie du point de passage et du barème en vigueur. Si le participant est arrivé au bout de l'étape, on met à jour son classement et son temps. Si l'étape est terminée (chaque participants est arrivé ou a abandonné), on met à jour les tables <i>PARTICIPANT</i> et <i>EQUIPE</i> , ainsi que <i>PORTER</i> . C'est dans ces scripts que seront appellés la procédure de mise à jour des classements.

Si la majorité des éléments cités dans ce tableaux sont assez simples et attachés à une fonction précise de l'automatisation, les deux scripts *TI_PASSER_BEFORE* et *TI_PASSER_AFTER* sont plus complexes car ils effectuent un très grand nombre d'actions. Pour clarifier son fonctionnement au lecteur, le schéma ci-dessous décrit sa dynamique.

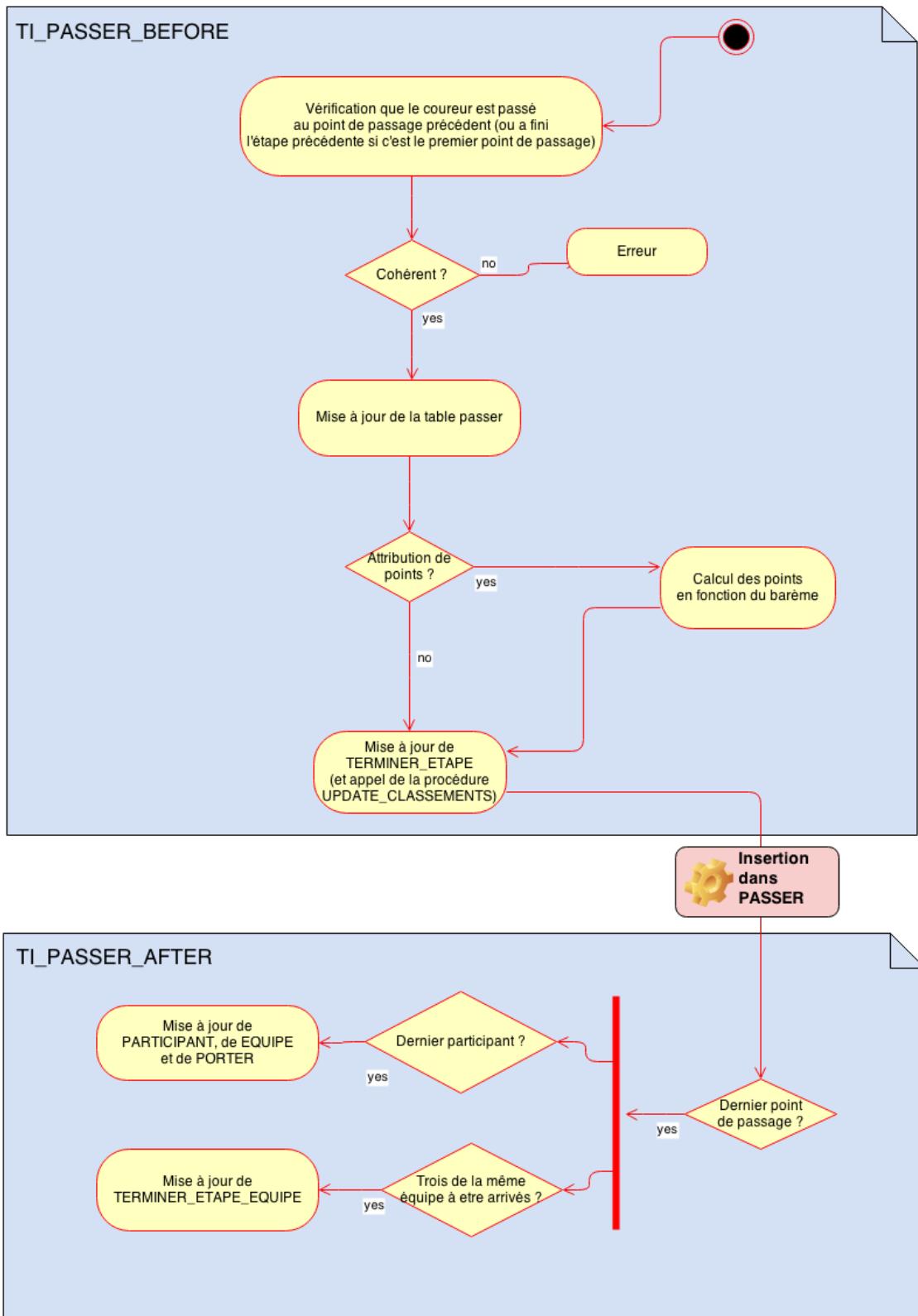


Illustration 39: Fonctionnement de TI_PASSER_BEFORE et TI_PASSER_AFTER

Ci-dessous le déclencheur *TI_PARTICIPANT* pour l'inscription d'un participant à un tour.

```

CREATE OR REPLACE TRIGGER ti_participant
BEFORE INSERT ON participant
FOR EACH ROW
DECLARE
    r_cycliste cycliste%ROWTYPE;
BEGIN
    --On récupère de la table CYCLISTE les données redondées dans la table
PARTICIPANT
    SELECT * INTO r_cycliste FROM cycliste WHERE cycliste_num = :new.cycliste_num;
    :new.cycliste_nom := r_cycliste.cycliste_nom;
    :new.cycliste_prenom := r_cycliste.cycliste_prenom;
    :new.cycliste_daten := r_cycliste.cycliste_daten;
    :new.part_tps_gene := 0;
    :new.part_pts_mont := 0;
    :new.part_pts_sprint := 0;

    --On récupère le nom de l'équipe associée
    SELECT equipe_nom INTO :new.equipe_nom FROM equipe WHERE tour_annee =
:new.tour_annee AND equipe_num = :new.equipe_num;
    --On récupère le nom du pays associé
    SELECT pays_nom INTO :new.cycliste_pays FROM pays WHERE pays_num =
r_cycliste.pays_num;

    --On récupère le nombre de participants déjà inscrits dans cette équipe
    SELECT COUNT(*) INTO :new.part_num FROM participant WHERE tour_annee =
:new.tour_annee AND equipe_num = :new.equipe_num;

    --Et on attribue au participant le numéro suivant
    :new.part_num := (:new.equipe_num - 1) * 10 + :new.part_num + 1;

    EXCEPTION
        WHEN no_data_found THEN dbms_output.put_line('Erreur');
END ti_participant;

```

Ci-dessous le déclencheur *TI_POINT_PASSAGE* pour l'ajout d'un point de passage.

```

CREATE OR REPLACE TRIGGER ti_point_passage
BEFORE INSERT ON point_passage
FOR EACH ROW
DECLARE
    inconsistency_km EXCEPTION;
    v_pt_pass_km_dep_previous point_passage.pt_pass_km_dep%TYPE;
BEGIN
    --Mise à jour automatique du numéro de point de passage
    SELECT COUNT(*) INTO :new.pt_pass_num FROM point_passage WHERE tour_annee =
:new.tour_annee and etape_num = :new.etape_num;
    :new.pt_pass_num := :new.pt_pass_num + 1;

    --Si ce n'est pas le premier point de passage, on vérifie que le kilométrage est
    --cohérent
    IF :new.pt_pass_num > 1 THEN
        --On vérifie que le kilométrage est cohérent
        SELECT pt_pass_km_dep INTO v_pt_pass_km_dep_previous FROM point_passage
        WHERE tour_annee = :new.tour_annee AND etape_num = :new.etape_num AND
        pt_pass_num = :new.pt_pass_num - 1;

        IF v_pt_pass_km_dep_previous > :new.pt_pass_km_dep THEN
            RAISE inconsistency_km;
        END IF;
    ELSIF :new.pt_pass_km_dep > 0 THEN
        RAISE inconsistency_km;
    END IF;

```

```

--On récupère les informations sur la ville
IF :new.ville_num IS NOT NULL THEN
    SELECT ville_nom INTO :new.pt_pass_ville_nom FROM ville WHERE ville_num
= :new.ville_num;
    IF :new.pt_pass_num = 1 THEN
        UPDATE etape SET
            ville_num_debuter = :new.ville_num
            ,ville_nom_debuter = :new.pt_pass_ville_nom
        WHERE
            tour_annee = :new.tour_annee
            AND etape_num = :new.etape_num;
    END IF;
    UPDATE etape SET
        ville_num_finir = :new.ville_num
        ,ville_nom_finir = :new.pt_pass_ville_nom
    WHERE
        tour_annee = :new.tour_annee
        AND etape_num = :new.etape_num;
    END IF;

--Mettre à jour l'étape (distance)
UPDATE etape SET
    etape_distance = :new.pt_pass_km_dep
WHERE
    tour_annee = :new.tour_annee
    AND etape_num = :new.etape_num;

--Mise à jour tous les points de passage de l'étape (km arrivée)
UPDATE point_passage SET
    pt_pass_km_arr = :new.pt_pass_km_dep - pt_pass_km_dep
WHERE
    tour_annee = :new.tour_annee
    AND etape_num = :new.etape_num;

:new.pt_pass_km_arr := 0;

EXCEPTION
    WHEN no_data_found THEN dbms_output.put_line('Fatal erreur');
    WHEN inconsistency_km THEN
        dbms_output.put_line('Erreur de cohérence: Vérifier que le kilométrage est
supérieur au point de passage précédent');
        RAISE inconsistency_km;
END ti_point_passage;

```

III. Insertion d'un jeu de données

Afin de bénéficier d'une application fonctionnelle, il faut disposer de données plus ou moins importantes. Afin d'obtenir des résultats crédibles, plusieurs scripts écrits en Python ont été élaborés dans le but de récupérer automatiquement les données relatives à la course depuis le site officiel du tour de France. Ainsi, note application est fournie avec les données réelles du tour de France 2013 telles qu'elles sont affichées depuis le site.

Bien entendu, étant donné la complexité du site web, les scripts ont été plus ou moins fastidieux à développer. De plus il aura fallu retravailler les données récupérées en sortie de script, et ajouter les données manquantes dans beaucoup de tables à l'aide de procédures un peu différent des TRIGGER développés pour l'automatisation, cette dernière n'étant fonctionnelle que lorsque les résultats sont saisis dans l'ordre et au fur et à mesure. La manipulation de fichiers avec des expressions régulières aura aussi été utilisée pour faciliter cette démarche.

IV. Développement Oracle Web Toolkit

Afin de simplifier l'organisation et la structuration de notre code source, nous avons organisé et structuré le code source en différents « packages ». En plus d'améliorer la lisibilité et la modularité, les performances sont accrues étant donné que le moteur charge en mémoire le package

entier lorsqu'une de ses procédures est appelée. Ainsi, le moteur n'a plus à faire d'opérations d'Entrée/Sortie sur le disque.

NAME

DB_COMMUN
DB_COURSE
DB_INSCRIPTION
DB_PARAM_COMMUN
DB_RESULTAT
UI_ADMINISTRATION
UI_AUTHENTIFICATION
UI_COMMUN
UI_COURSE
UI_INSCRIPTION
UI_PARAM_COMMUN
UI_RESULTAT
UI_UTILS

13 ligne(s) sélectionnée(s).

Illustration 40: Liste des packages

Nous avons deux grandes catégories de package :

- Les « packages » d'accès à la base de donnée sont préfixés par « **DB** »
- Les « packages » nécessaire à la création de l'interface Homme-Machine c'est à dire ceux qui génèrent du code HTML, sont préfixés par « **UI** »

Pour ces deux grands types de « packages », nous avons mis en place une décomposition en modules qui respecte notre architecture applicative. Les packages *XX_INSCRIPTION* correspondent au module « Gestion des inscriptions » c'est à dire la gestion des participants et des équipes inscrites à un tour.

DB_INSCRIPTION permet l'accès aux tables *PARTICIPANT*, *EQUIPE* et *CYCLISTE*. Par souci de lisibilité, nous présenterons qu'un aperçu de l'interface de ce package.

```

CREATE OR REPLACE PACKAGE "G11_FLIGHT"."DB_INSCRIPTION" AS
    --Retourne un participant en fonction de son numéro
    FUNCTION getPart(n_part varchar2)
        RETURN participant%rowtype;

    [...]

    --Retourne un participant en fonction de certains critères
    FUNCTION getPartCrit
        (crit_nom varchar2 default '%',
         crit_pnom varchar2 default '%')
        RETURN db_param_commun.ref_cur;

    [...]

END DB_INSCRIPTION;
/

```

Illustration 41: Extrait interface package DB_INSCRIPTION

Nous allons détailler la fonction *getPartCrit* qui permet de retourner la liste les participants selon les critères de recherches saisis par l'utilisateur :

```
CREATE OR REPLACE PACKAGE BODY "G11_FLIGHT"."DB_INSCRIPTION" AS  
[...]  
  
FUNCTION getPartCrit(  
    crit_nom varchar2 default '%',  
    crit_pnom varchar2 default '%')  
return db_param_commun.ref_cur IS  
cur_part db_param_commun.ref_cur;  
BEGIN  
OPEN cur_part FOR  
    select *  
    from participant  
    where tour_annee = ui_utils.getSelectedTour  
    AND UPPER(cycliste_nom) like UPPER('%'||crit_nom||'%')  
    AND UPPER(cycliste_prenom) like UPPER('%'||crit_pnom||'%')  
    order by part_num;  
return cur_part;  
exception when others then  
return null;  
END getPartCrit;  
[...]  
  
END DB_INSCRIPTION;  
/
```

Illustration 42: Détail fonction *getPartCrit*

Pour ce type de fonction qui retourne plusieurs lignes, nous utilisons des référence de curseur. Son utilisation nous évite de déclarer un nouveau curseur dans chaque procédure nécessitant un résultat contenant plusieurs lignes. Nous avons donc déclaré un type « *REF_CURSOR* » nommé « *ref_cur* » dans un package contenant tous les paramètres communs pour l'accès aux données (*DB_PARAM_COMMUN*).

DB_COURSE est utilisé pour les accès aux tables *ETAPE* et *POINT_PASSAGE* qui sont les constituants du parcours. Les accès à la table *TOUR* pouvant être transversaux, il a été choisi de mettre les procédures d'accès dans un autre « package » nommé *DB_COMMUN*. Ce package contient également des fonctions permettant de récupérer le dernier Tour saisi ou la dernière étape parcourue, ce qui peut être utile dans de nombreuses requêtes.

```

CREATE OR REPLACE PACKAGE "G11_FLIGHT"."DB_COMMUN" AS

FUNCTION getAllTour RETURN db_param_commun.ref_cur;

FUNCTION getLastTour RETURN tour.tour_annee%type;

FUNCTION getLastEtape RETURN etape.etape_num%type;

FUNCTION getTour(a_tour varchar2) RETURN tour%rowtype;

FUNCTION getAcroPays(n_part participant.part_num%TYPE )
RETURN pays.pays_acro%TYPE;

END DB_COMMUN;
/

```

Illustration 43: Extrait interface package DB_COMMUN

Le package DB_RESULTAT est très important pour notre système. En effet, l'une des fonctionnalités essentielles de l'application est la consultation des différents classements d'un Tour. L'interface permet de voir qu'il est possible de récupérer le porteur d'un maillot à une étape donnée ou de récupérer le classement d'une étape.

```

CREATE OR REPLACE PACKAGE "G11_FLIGHT"."DB_RESULTAT"
IS
[...]
-- Retourne le porteur d'un maillot pour une étape donnée
FUNCTION getPorteur(
    v_maillot porter.maillot_couleur%type,
    n_etape porter_etape_etape_num%type)
RETURN participant%rowtype;

--Retourne le classement d'une étape donnée
FUNCTION getEtapeRanking(
    nb_ligne number default 999,
    n_etape etape_etape_num%TYPE default 1)
RETURN db_param_commun.ref_cur;

[...]

-- Procédure de mise à jour des classements appelée depuis un trigger
PROCEDURE update_classements (
    v_tour_annee tour.tour_annee%TYPE ,
    v_etape_num etape_etape_num%TYPE);

END DB_RESULTAT;
/

```

Illustration 44: Extrait interface package DB_RESULTAT

La fonction *getPorteur* utilise la table *PORTER* pour connaître le porteur d'un maillot distinctif.

```

FUNCTION getPorteur(v_maillot porter.maillot_couleur%type,n_etape
porter.etape_num%type)
  RETURN participant%rowtype
IS
  n_part porter.part_num%type;
BEGIN
  SELECT
    part_num INTO n_part
  FROM
    porter
  WHERE
    etape_num = n_etape
  AND tour_annee = ui_utils.getselectedtour
  AND maillot_couleur = v_maillot ;

  return db_inscription.getPart(n_part);

EXCEPTION WHEN OTHERS THEN
  return null;
END getPorteur;

```

Illustration 45: Détail fonction getPorteur

De plus la fonction *getEtapeRanking* utilise la table *TERMINER_ETAPE* pour connaître le classement d'une étape spécifique.

```

FUNCTION getEtapeRanking(nb_ligne number default 999,n_etape
etape.etape_num%TYPE default 1)
  return db_param_commun.ref_cur IS
  cur_part db_param_commun.ref_cur;
BEGIN
  OPEN cur_part FOR
  SELECT
    *
  FROM
    terminer_etape
  WHERE
    tour_annee=ui_utils.getselectedTour
  AND etape_class <= nb_ligne
  AND etape_num=n_etape
  AND etape_class != 0
  ORDER BY etape_class;
  return cur_part;
EXCEPTION WHEN OTHERS THEN
  null;
END getEtapeRanking;

```

Illustration 46: Détail fonction getEtapeRanking

Le détail de cette fonction permet de voir l'utilisation de la fonction *getSelectedTour*. Cette fonction se trouve dans les packages dédiés à la génération de l'IHM. Cette fonction permet de

récupérer le Tour sélectionné par l'utilisateur. Cette fonction est utilisée dans toutes les requêtes et permet ainsi un affichage cohérent des informations tout au long de la navigation de l'utilisateur. Le Tour sélectionné par l'utilisateur est stocké dans un cookie. `getSelectedTour` se trouve dans un package contenant toutes fonctions et procédures « utiles » et « transversales » pour la gestion de l'IHM (`UI_UTILS`). Ce package regroupe les fonctions liées à la manipulation des cookies, au formatage des temps ou à la génération de code HTML utile.

```
--Retourne le Tour sélectionné par l'utilisateur
FUNCTION getSelectedTour RETURN varchar2 IS
a_tour varchar2(4);
BEGIN
  if(ui_utils.existCookie('Tour')) then
    a_tour := getCookieValue('Tour');
  else
    a_tour := db_commun.getLastTour;
  end if;
  RETURN a_tour;
END getSelectedTour;
```

Illustration 47: Détail fonction `getSelectedTour`

Nous avons créé également un package `UI_ADMINISTRATION` qui permet de gérer toute la partie « Administration » du Tour de France. Pour le moment, il n'est possible que de saisir les résultats à un point de passage. Il contient l'interface de gestion ainsi que les procédures pour la saisie de résultat à un point de passage.

Le package `UI_AUTHENTICATION` est dédié à la gestion de l'authentification de l'utilisateur. Il contient le formulaire d'authentification, le formulaire de déconnexion, une procédure pour vérifier si l'utilisateur existe dans la base, une procédure pour afficher un message d'erreur lors de la saisie d'un utilisateur/mot de passe et un formulaire pour afficher l'utilisateur authentifié avec un lien vers la page d'administration.

Nous avons développé en priorité la partie administration et gestion utilisateurs avec Apex donc les fonctionnalités de ces deux derniers packages sont minimales mais permettent d'avoir un aperçu concret de la suite de l'implémentation de ces modules.

Enfin, un package `UI_COMMUN` permet de regrouper les parties communes à toutes les pages de l'application comme le menu, un entête de page, un pied de page ou la liste de sélection du Tour. De cette manière le squelette de toutes nos pages est identique.

```
PROCEDURE [NOM_PROCEDURE] IS
BEGIN
  UI_COMMUN.UI_HEAD;          -- Entête HTML
  UI_COMMUN.UI_HEADER;        -- Entête de la page
  UI_COMMUN.UI_MAIN_OPEN;     -- Cadre principal
  -- Corps de la page
  UI_COMMUN.UI_MAIN_CLOSE;
  UI_COMMUN.UI_FOOTER;        -- Pied de page
END [NOM_PROCEDURE];
```

Illustration 48: Squelette page IHM

Un extrait de la procédure UI_LEQUIPE qui affiche la liste des équipes, permet d'illustrer la méthode utilisée pour la mise en page ainsi que l'utilisation des curseurs. Ces derniers ont été largement utilisés pour afficher les différentes listes dans notre application.

```
PROCEDURE UI_LEQUIPE (crit_nom varchar2 default '') IS
    equipes db_param_commun.ref_cur;
    rec_equipe equipe%rowtype;
    cpt number(3) :=0;
BEGIN
    UI_COMMUN.UI_HEAD;
    UI_COMMUN.UI_HEADER;
    UI_COMMUN.UI_MAIN_OPEN;

    [...]

    htp.tableOpen(cattributes => 'class="normalTab"');
    htp.tableheader('Numéro',cattributes => 'class="col2"');
    htp.tableheader('Nom');
    htp.tableheader('Web');
    htp.tableheader('Pays');

    equipes := db_inscription.getEquipeCrit(crit_nom);
    fetch equipes into rec_equipe;
    while(equipes%found) loop
        cpt:=cpt+1;
        ui_utils.color_row_p(cpt);

        htp.tableData(rec_equipe.equipe_num);
        htp.tableData(htf.anchor ('ui_inscription.ui_detail_equipe?
n_equipe=' || rec_equipe.equipe_num,rec_equipe.equipe_nom));
        htp.tableData(rec_equipe.equipe_web);
        htp.tableData(rec_equipe.equipe_pays);
        htp.tableRowClose;
        htp.tableRowOpen;
        htp.tableRowClose;
        fetch equipes into rec_equipe;
    END LOOP;
    htp.tableClose;

    [...]

    UI_COMMUN.UI_MAIN_CLOSE;
    UI_COMMUN.UI_FOOTER;
END UI_LEQUIPE;
```

Illustration 49: Extrait procédure UI_LEQUIPE

V. Développement APEX

Pour le Backoffice, nous avons opté pour Oracle APEX (Application Express) qui est un environnement de développement intégré permettant de créer des applications de type web en développement rapide et dont le but est d'exploiter des bases de données Oracle.

Ainsi, on a pu développer un CRUD facilitant l'administration et permettant plus d'interactivité tout étant simple d'utilisation pour l'utilisateur cible qui ne sera pas un spécialiste des bases de données.

La page d'authentification

L'authentification se base sur un Authentication Scheme qu'on a créé (dans "Shared Components") et utilise les comptes APEX comme fonction d'authentification. Le nom d'utilisateur et le mot de passe sont **g11_flight**.

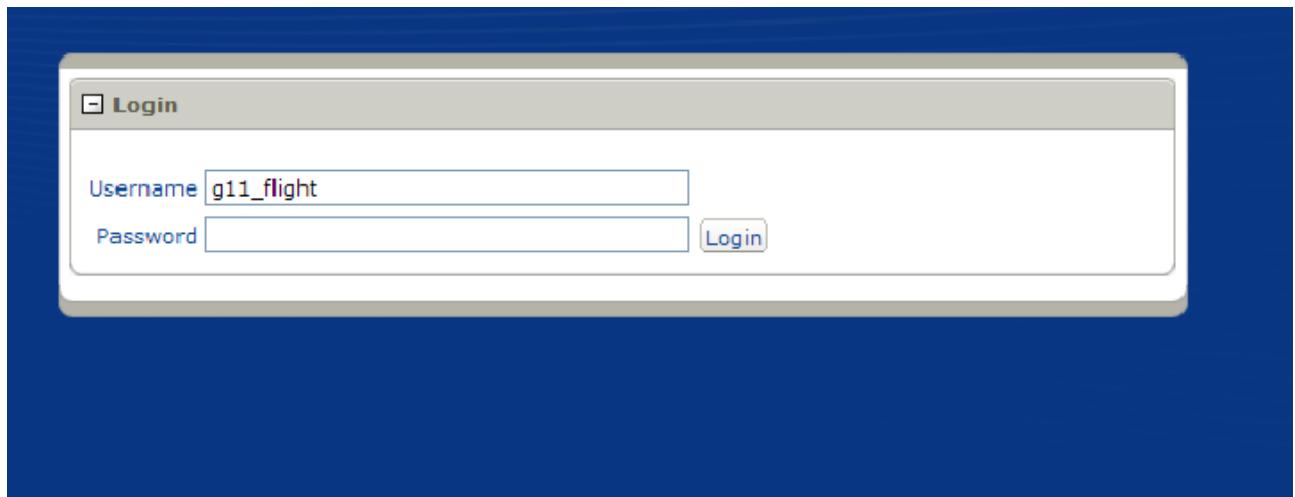


Illustration 50: Page d'authentification

La page d'accueil

La page d'accueil est de type « Master-Detail Report » et se compose de :

- Un menu
- Un tableau avec liste des tours
- Un bouton permettant de créer un Tour

Tours					Accueil	Cyclistes	Autres
Tour							
Tour					Create		
Edit	Edition Tour	Date début	Date fin	Nom Tour			
	104	04/06/2012	26/06/2012	Tour 2012			
	100	29/06/2013	21/07/2013	Tour de France 2013			
	102	12/03/2015	12/06/2015	Tour 2015			

Illustration 51: Page d'accueil

L'utilisateur a la possibilité de voir les détails du tour, de l'éditer (changer le nom, l'édition, ...), gérer ses différents étapes, ses équipes et ses participants.

Page édition tour

Année Tour 2013
Edition Tour 100
Nom Tour Tour de France 2013
2 of 3
Gestion des participants
Gestion des équipes
Gestion des étapes

<input type="checkbox"/> Editer	Nom Etape	Date Etape	Distance Etape	Type Etape	Ville Départ	Ville Arrivée
<input type="checkbox"/>	Porto-Vecchio / Bastia	29/06/2013	213	ligne	PORTO-VECCHIO	BASTIA
<input type="checkbox"/>	Bastia / Ajaccio	30/06/2013	156	ligne	BASTIA	AJACCIO
<input type="checkbox"/>	Ajaccio / Calvi	01/07/2013	146	ligne	AJACCIO	CALVI
<input type="checkbox"/>	Nice / Nice	02/07/2013	25	ligne	NICE	NICE
<input type="checkbox"/>	Cagnes-sur-Mer / Marseille	03/07/2013	229	ligne	CAGNES-SUR-MER	MARSEILLE
<input type="checkbox"/>	Aix-en-Provence / Montpellier	04/07/2013	177	ligne	AIX-EN-PROVENCE	MONTPELLIER
<input type="checkbox"/>	Montpellier / Albi	05/07/2013	206	ligne	MONTPELLIER	ALBI

Illustration 52: Page d'édition du tour

Cette page, qui est la partie «Detail Report » de la page d'accueil est composée de deux parties :

- Partie Tour : pour éditer le tour. Elle inclut aussi les liens permettant de gérer les équipes, les étapes et les participants.
- Parties étapes : permet de voir, d'éditer et de supprimer les étapes.

Pour l'affichage de la liste des étapes on a utilisé la requête suivante dans laquelle on récupère le paramètre passé de la page d'accueil :

```
SELECT '' edit_link,
       "etape_num",
       "tour_annee",
       "etape_nom",
       "etape_date",
       "etape_distance",
       "tetape_lib",
       "ville_nom_debuter",
       "ville_nom_finir",
       "ville_num_finir",
       "ville_num_debuter"
FROM   "#OWNER#"."etape"
WHERE  "tour_annee" = :P42_TOUR_ANNEE
```

Illustration 53: Requête des étapes d'un tour

Page gestion des participants

The screenshot shows a software interface for managing participants in a tour. At the top, it displays the year (Année 2013), edition (Edition 100), start date (Date Début 29/06/2013), end date (Date Fin 21/07/2013), the tour name (Nom Tour de France 2013), and a page indicator (2 of 3). Below this, a table lists nine participants:

	Nom	Poids (Kg)	Taille (cm)	Equipe
	ALBASINI Michael	65	172	ORICA GREENEDGE
	AMADOR Andrey	73	181	MOVISTAR TEAM
	ANTON Igor	58	173	EUSKALTEL - EUSKADI
	ARASHIRO Yukiya	64	170	TEAM EUROPACAR
	ASTARLOZA Mikel	72	185	EUSKALTEL - EUSKADI
	BAGOT Yoann	65	182	COFIDIS, SOLUTIONS CREDITS
	BAK Lars Ytting			LOTTO-BELISOL
	BAKELANTS Jan	7	177	RADIOSHACK LEOPARD

Illustration 54: Page gestion des participants

Cette page contient une partie qui affiche un récapitulatif du tour qu'on édite et d'un tableau (de type Tabular) permettant d'ajouter des participants à notre tour en indiquant leur poids, leur taille et l'équipe avec qui il participe durant le tour. On a aussi la possibilité d'ajouter un participant sans l'affecter à une équipe.

Page gestion des équipes

The screenshot shows a software interface for managing teams. At the top, it displays the edition (Edition 100), start date (Date Début 29/06/2013), end date (Date Fin 21/07/2013), the tour name (Nom du Tour Tour de France 2013), and a page indicator (2 of 3). Below this, a table lists ten teams:

	Nom	Site Web	Description	Temps Générale	Classement Générale	Sponsor	Acronyme Sponsor	Pays
	AG2R LA MONDIAL	alm.com	AG2R LA MONDIAL	(null)	(null)	AG2R LA MONDIAL	alm	France
	ASTANA PRO TEAM	ast.com	ASTANA PRO TEAM	(null)	(null)	ASTANA PRO TEAM	ast	Kazakhstan
	BELKIN PRO CYCL	bel.com	BELKIN PRO CYCL	(null)	(null)	BELKIN PRO CYCL	bel	Pays-Bas
	BMC RACING TEAM	bmc.com	BMC RACING TEAM	(null)	(null)	BMC RACING TEAM	bmc	États-Unis
	CANNONDALE	can.com	CANNONDALE	(null)	(null)	CANNONDALE	can	Italie
	COFIDIS, SOLUTIC	cof.com	COFIDIS, SOLUTIC	(null)	(null)	COFIDIS, SOLUTIC	cof	France
	EUSKALTEL - EUSK	eus.com	EUSKALTEL - EUSK	(null)	(null)	EUSKALTEL - EUSK	eus	Espagne
	FDJ.FR	fdj.com	FDJ.FR	(null)	(null)	FDJ.FR	fdj	France
	GARMIN - SHARP	grs.com	GARMIN - SHARP	(null)	(null)	GARMIN - SHARP	grs	États-Unis
	KATUSHA TEAM	kat.com	KATUSHA TEAM	(null)	(null)	KATUSHA TEAM	kat	Russie

Illustration 55: Page gestion des équipes

Cette page contient une partie qui affiche un récapitulatif du tour comme dans le page de gestion des participants et d'un tableau (de type Tabular) permettant d'ajouter les équipes au tour en indiquant son nom, son site web, sa description, son pays, son sponsorleur poids, leur taille et l'équipe avec qui il participe durant le tour. On a aussi la possibilité d'ajouter un participant sans l'affecter à une équipe.

Les colonnes Temps Générale et Classement Générale permettent d'afficher le temps et classement générale d'une équipe une fois que le tour a commencé. Sinon, on n'affiche rien.

Page gestion des étapes

Gestion des étapes							
Edit	Numéro Etape	Nom Etape	Date Etape	Distance Etape (Km)	Type Etape	Ville Départ	Ville Arrivée
	1	Etape 1	03/06/2014		ligne	AILLON-LE-JEUNE	ARMENTEULE
	1	Porto-Vecchio / Bastia	29/06/2013	213	ligne	PORTO-VECCHIO	BORGO
	2	Bastia / Ajaccio	30/06/2013	156	ligne	BASTIA	AJACCIO
	3	Ajaccio / Calvi	01/07/2013	146	ligne	AJACCIO	CALVI
	4	Nice / Nice	02/07/2013	25	ligne	NICE	NICE
	5	Cagnes-sur-Mer / Marseille	03/07/2013	229	ligne	CAGNES-SUR-MER	MARSEILLE
	6	Aix-en-Provence / Montpellier	04/07/2013	177	ligne	AIX-EN-PROVENCE	MONTPELLIER
	7	Montpellier / Albi	05/07/2013	206	ligne	MONTPELLIER	ALBI
	8	Castres / Ax 3 Domaines	06/07/2013	195	ligne	CASTRES	AX 3 DOMAINES
	9	Saint-Girons / Bagnères-de-Bigorre	08/07/2013	169	ligne	SAINTE-GIRONS	BAGNÈRES-DE-BIGORRE
	10	Saint-Gildas-des-Bois / Saint-Malo	09/07/2013	197	ligne	SAINT-GILDAS-DES-BOIS (VC-D2)	SAINT-MALO
	11	Avranches / Mont-Saint-Michel	10/07/2013	33	ligne	AVRANCHES	LE MONT-SAINT-MICHEL
	12	Fougères / Tours	11/07/2013	218	ligne	FOUGÈRES	TOURS
	13	Tours / Saint-Amand-Montrond	12/07/2013	173	ligne	TOURS	SAINT-AMAND-MONTROND
	14	Saint-Pourçain-sur-Sioule / Lyon	13/07/2013	191	ligne	SAINT-POURÇAIN-SUR-SIOULE	LYON

Illustration 56: Page gestion des étapes

Cette page , qui est de type « Master », permet d'ajouter, de modifier et de supprimer les étapes d'un tour.

Une fois que l'utilisateur appuie sur le pinceau pour éditer, il est redirigé vers une page de type « Détail » lui permettant de gérer les points de passage du tour sélectionné.

The screenshot shows two windows side-by-side. The top window is titled 'Edit ETAPE' and contains fields for 'Annee Tour' (2013), 'Numéro Etape' (3), 'Nom Etape' (Ajaccio / Calvi), 'Date Etape' (01-JUL-2013), 'Type Etape' (ligne), 'Ville Départ' (AJACCIO), and 'Ville Arrivée' (CALVI). A status bar at the bottom says '3 of 22'. The bottom window is titled 'POINT_PASSAGE Detail' and displays a table of points of passage with columns for Nom Pt Passage, Km Depart, Altitude Pt Passage, Horaire Pt Passage, Ville, and Catégorie. The table data is as follows:

Nom Pt Passage	Km Depart	Altitude Pt Passage	Horaire Pt Passage	Ville	Catégorie
AJACCIO	0	100	12h55	AJACCIO	
A Bracalina (ALATA)	4	100	13h00		
Col de San Bastiar	12	100	13h11		Col de catégories 4
Masorchia (CALCA)	19.5	100	13h21		

Illustration 57: Pages gestion des points de passage d'une étape

L'application permet aussi :

- La gestion des cyclistes
- La gestion des catégories de points de passage
- La gestion des directeurs sportifs
- La gestion des spécialistes pour le vote du meilleur combatif

Conclusion

La réalisation de ce projet aura été pour nous l'occasion de mettre en pratique la plupart des notions vues dans les unités de valeur GL52 et BD50, aussi bien en ce qui concerne les connaissances relatives à la spécification ou à la conception qu'au au développement à proprement parlé.

Nous avons pu observer la véritable valeur du travail de spécification effectué en amont du développement qui, s'il est rigoureusement réalisé, permet de gagner un temps et une énergie considérables, alors que ce bénéfice ne semblait pas clairement évident au premier abord. La combinaison des deux modules a débouché sur un logiciel complet et fonctionnel, et un dossier de spécification pouvant servir de documentation ou de support à d'éventuelles opérations de maintenance sur la solution.

En ce qui concerne la partie spécification, les tâches ont été menées à terme, non sans difficultés pour certains aspects. En effet, il nous aura été parfois difficile d'appliquer les concepts vus en cours à des cas particuliers, et nous nous sommes parfois heurté à de vrais problèmes qu'il nous aura alors été difficile de surmonter, étant donné notre faible expérience en la matière. En particuliers, les diagrammes de séquence, ou la modélisation des classements dans le diagramme de classe aura été un vrai défi, que nous espérons avoir relevé avec succès.

Bien entendu, le dossier de spécification, s'il est conforme à ce qui avait été demandé dans le cadre de ces unités de valeur, est incomplet puisque seulement un nombre restreint des fonctionnalités produites par le logiciel ont été traitées jusqu'à la profondeur nécessaire pour la production et la mise à disposition d'une telle application. Nous avons pu voir qu'un dossier de spécification complet peut prendre, sinon plus, autant de temps que les tâches relatives au développement et à la mise en œuvre de la solution.

La partie attachée au développement, si elle est beaucoup moins complexe, nécessite un temps considérable. Comme à notre habitude dans ce genre de projet, nous avons sous-estimé volontairement le temps que prendraient les différentes tâches de cette partie pour arriver à une solution la plus aboutie possible. De mauvais choix de conceptions ont parfois entravé ou retardé le bon déroulement du développement, ce qui montre encore une fois l'importance du travail de spécification. Il faut dire que nous n'avons pas eu le temps de prendre le recul nécessaire pour avoir un dossier de spécification complet et parfait au premier jet. Ainsi, nous avons quelquefois été contraints de revenir en arrière pour faire correspondre les spécifications du logiciel avec les nouveaux besoins ou les nouvelles contraintes qui avaient été identifiées.

Au moment de l'écriture du rapport, même si nous avons une solution utilisable, un certain nombre de fonctionnalités restent manquantes. Par exemple, nous ne disposons pas actuellement de module d'authentification, une brique nécessaire pour la gestion des différents accès depuis le site, même si cela est partiellement résolut avec le développement APEX du site.

Au final, nous pouvons affirmer (en toute modestie) que nous sommes plutôt satisfaits du travail accompli. S'il n'a pas la prétention d'être parfait et exhaustif, il nous aura permis d'acquérir de nombreuses aptitudes indispensables dans le domaine du génie logiciel, et à bien intégrer les bonnes pratiques et l'importance des différentes étapes, inhérentes à la production d'un logiciel de qualité.

Annexes

I. Modèle conceptuel de données

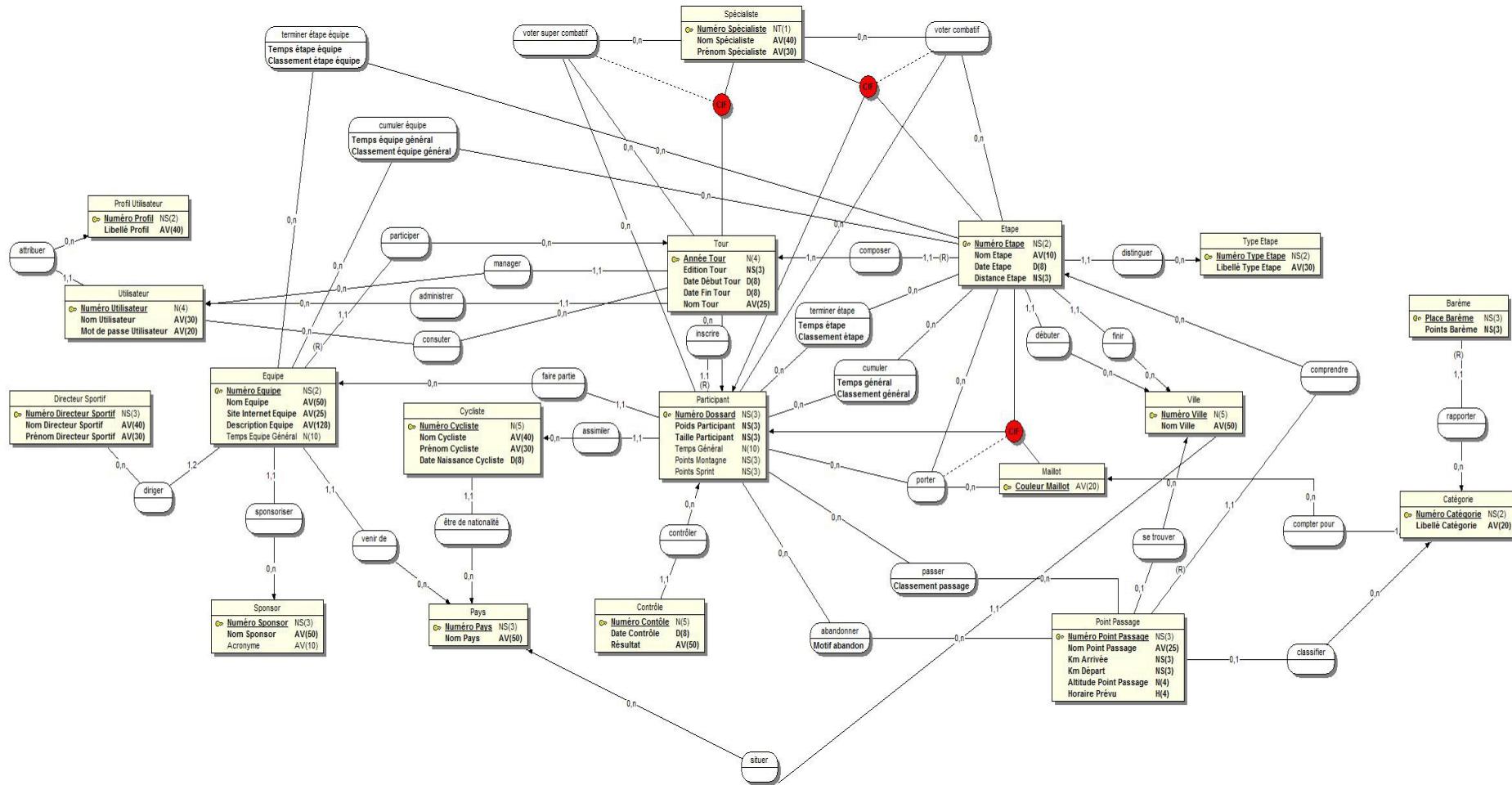


Illustration 58: Modèle complet

II. Modèle Logique Relationnel Normalisé

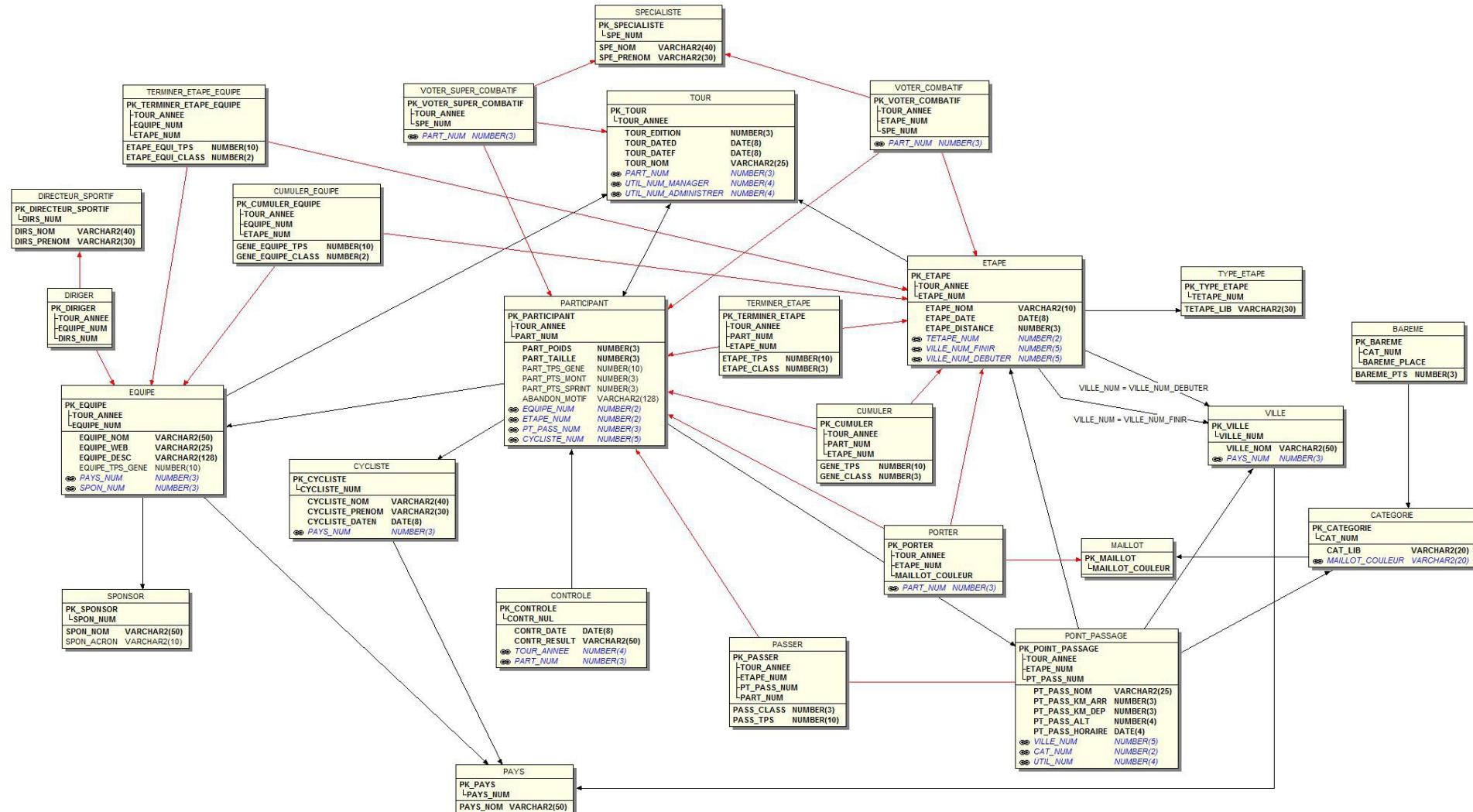


Illustration 59: MLR Normalisé Complet

III. Modèle Logique Relationnel Optimisé

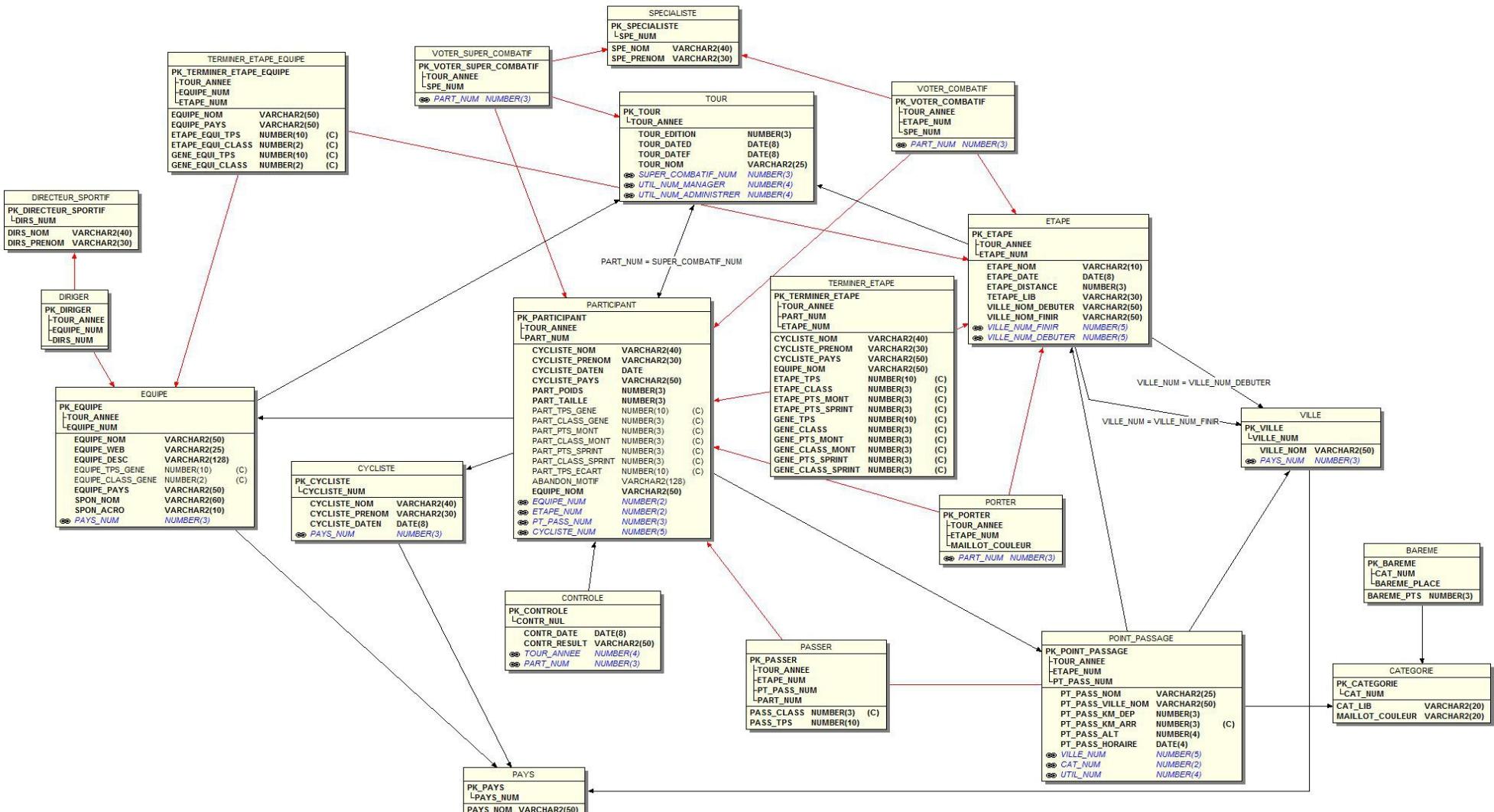


Illustration 60: MLR Optimisé Complet