Diag-K: Ontologie pour l'élaboration du modèle de connaissances de l'apprenant dans les EIAH

Amina BAGHLI

27 juin 2011

Résumé

Dans ce rapport, nous proposons une ontologie (vocabulaire RDF) générale et extensible nommée "Diag-K", pour l'élaboration du modèle des connaissances de l'apprenant. Cette ontologie regroupe les connaissances nécessaires à inscrire dans les EIAH (Environnements Informatiques pour l'Apprentissage Humain) pour diagnostiquer le niveau des connaissances de l'apprenant par rapport au domaine (s) enseigné (s). Le concepteur de l'EIAH pourra utiliser le vocabulaire que nous avons défini pour exprimer dans son système les connaissances qu'il juge pertinentes (les concepts du domaine enseigné, les solutions de l'expert du domaine, etc.) pour l'accomplissement du diagnostic des connaissances de l'apprenant interagissant avec cet EIAH. Pour la mise en place de notre ontologie, nous nous sommes fondés sur l'étude et l'analyse de différents travaux existants dans l'axe de recherche de la modélisation de l'apprenant dans les EIAH.

Mots clès : Environnement Informatique pour l'Apprentissage Humain (EIAH), modèle de l'apprenant, modélisation de l'apprenant, modèle du domaine, modèle de l'expert, modèle basé sur des contraintes (CBM).

Remerciment

Je tiens à remercier vivement mon encadrent Mme SETTOUTI Lemya de m'avoir confié ce travail, et d'avoir bien voulu me faire profiter de sa compétence, de sa rigueur scientifique, de ses remarques pertinentes et de ses conseils judicieux. Qu'elle trouve ici l'expression de ma profonde reconnaissance.

Un très grand remerciement à mon Co-encadrent Mme HALFAOUI Amel pour son soutien, son aide ainsi que ses précieux conseils qu'elle m'a prodigué pour réaliser ce travail.

Je rends hommage à tous mes professeurs d'université à qui je dois ma formation scientifique spécialement Mr MIDOUNI Djallel.

Mes vifs remerciements s'adressent à tous ceux qui m'ont aidé à réaliser ce modeste travail spécialement à ma chère amie Tiha et ma tendre cousine Zyneb.

Je remercie enfin mes chers parents, mes deux adorables sœurs et mon petit frère pour leur soutien et encouragements.

Dédicaces

Je dédie ce travail à mes chers parents sans qui je n'aurais jamais réussi, je les remercie pour leur soutien, encouragement, patience, compréhension et surtout leur grand amour.

A mes chères sœurs et mon petit frère.

A ma tendre tente et son mari MAHI.

A mon oncle Abdelatif, sa femme Wassila ainsi que toute la famille BAGHLI.

A mes deux adorables cousines Lemya et Amel avec ces deux fils Walid et Mehdi.

A mes chers ami(e)s et cousin(e)s : Tiha, Fadela, Lamia, Esma, Naziha, Imene, Meryam, Jija, wassila, Fayza, Zyneb, Lilya, Meryam, Rym, Selman, Hadi, Halim, Ahmed, Salim, Mohamed, les deux Djallel, Zaki...

A mon professeur d'anglais et tout mon groupe. De même, a mes professeurs : Amel, Djallel, Zyneb, Nedjet, Mr Benachenhou et Mr Mansouri.

A tous ma famille et tous ceux qui m'aiment et qui ont cru en moi de près comme de loin durant toute ma carrière universitaire.

Table des matières

Table des figures

Liste des tableaux

Chapitre 1

Introduction

La modélisation de l'apprenant appelée aussi diagnostic (Dillenbourg et Self [DS92]) est un champ de recherche très actif dans la communauté des Environnements Informatiques pour l'Apprentissage Humain (EIAH). Le diagnostic est le processus de modélisation de l'apprenant dans les EIAH, et consiste en l'ensemble des traitements permettant d'élaborer et de mettre à jour des informations pertinentes sur l'apprenant (l'état de ses connaissances, de ses compétences, ses stratégies de résolution de problèmes, ses performances, ses styles d'apprentissage, etc.) à partir de l'analyse de son comportement. L'analyse du comportement de l'apprenant consiste le plus souvent à observer et à tracer les informations sur les interactions de l'apprenant avec le système durant une session d'apprentissage, en suite à les analyser et les interpréter.

1.1 contexte du travail

Notre travail de recherche s'inscrit dans le cadre du projet "Personnalisation des Environnements Informatiques pour l'Apprentissage Humain" (EIAH) du cluster ISLE de la région Rhône-Alpes. L'objectif global de ce projet est de mettre en place les outils théoriques et informatiques pour comprendre le comportement de l'apprenant utilisant un EIAH et adapter son activité. Les méthodes et les formes de cette adaptation pouvant être nombreuses et variées, ce projet se décompose en cinq tâches étudiant chacune une certaine modalité de personnalisation. Interdépendantes et complémentaires, ces tâches s'intéressent à des objets théoriques et pratiques allant des traces d'interaction et scénario pédagogique à l'analyse des activités d'apprentissage et l'élaboration des modèles de connaissances de l'apprenant.

Un travail de recherche qui est en cours de réalisation dans le cadre d'une thèse effectuée au laboratoire LIRIS (Laboratoire d'InforRmatique en Image et Systèmes d'information), se place à l'intersection des objectifs de deux de ces tâches : la tâche 1 étudiant les traces d'interaction et la tâche 3 cherchant à élaborer le modèle de l'apprenant à partir de l'analyse de ces traces. Ce travail de thèse a aboutit à la mise en place d'un cadre conceptuel permettant d'assister et d'aider le concepteur de l'EIAH pour la réalisation du diagnostic. Par la suite, le cadre conceptuel devra déboucher sur la réalisation d'une plateforme utilisable par le concepteur d'un EIAH afin de lui faciliter la mise en œuvre de l'analyse automatique des traces d'activité pour l'élaboration de profils d'apprenants, en explicitant les connaissances nécessaires à inscrire dans le système pour son accomplissement. Ainsi, le concepteur d'un EIAH pourra expliciter le modelé des traces issues de son EIAH, le profil de l'apprenant qu'il souhaite constituer, et expliciter également les connaissances devant intervenir pour interpréter les traces d'interactions de l'apprenant avec l'EIAH afin de calculer et/ou inférer des éléments de son profil.

Ce cadre inclut donc des modèles de connaissances permettant de représenter explicitement le profil de l'apprenant, les observations sur les interactions de l'apprenant avec un EIAH, et d'exprimer les connaissances nécessaires pour l'interprétation des traces en vue de l'élaboration du profil de l'apprenant. Notre travail se situe à ce niveau, et consiste d'une part à définir une liste exhaustive des connaissances d'interprétation des traces pour la modélisation des connaissances de l'apprenant et d'autre part de proposer un vocabulaire RDF (une ontologie RDF/RDFS) permettant au concepteur d'un EIAH de décrire les connaissances qu'il juge pertinentes pour inférer l'état des connaissances de l'apprenant par rapport au domaine enseigné.

1.2 Motivation et objectif

En général, le modèle de l'apprenant est constitué d'une part d'éléments indépendants du domaine d'application tels que ses préférences ou ses performances, et d'autre part d'éléments dépendants du domaine et du contexte d'application, tels que ses connaissances et ses compétences [MFdCC08, dK01].

Le travail de SETTOUTI [SGML10] consiste à élaborer les éléments "indépendants du

domaine" pour la modélisation du profil de l'apprenant (informations personnelles, performances, préférences,etc.). Les connaissances permettant de diagnostiquer ces éléments sont regroupées au sein d'un méta-modèle (vocabulaire RDF) qu'elle a défini et est constitué entre autres de patterns de requêtes (requêtes SPARQL) génériques pour inférer une performance (par exemple : le temps mis par l'apprenant pour résoudre un exercice) ou inférer une stratégie de résolution de problème (par exemple : pour répondre à une question, l'apprenant consulte d'abord l'aide ou un exemple illustrant la résolution de l'exercice, ensuite il donne la réponse ; ou bien l'apprenant donne une réponse ensuite il demande au système de l'évaluer jusqu'à ce qu'il trouve la bonne réponse), et de seuils de comparaison prédéfinis par l'enseignant. Cette approche consiste à fournir au concepteur de l'EIAH les moyens nécessaires pour faire un diagnostic indépendamment des connaissances du domaine d'application.

Notre travail a pour objectif de compléter cette approche, en tenant compte des éléments dépendants du domaine d'application et donc des connaissances de diagnostic permettant leur élaboration. Nous avons orienté notre étude de l'état de l'art et des travaux existants dans ce sens afin d'étendre ce méta-modèle.

1.3 Plan du mémoire

Notre mémoire est organisé en six sections. Dans la première section nous avons présenté le contexte de notre travail ainsi que notre motivation et objectif. Dans la seconde section, nous présenterons l'état de l'art en ce qui concerne les Environnements Informatiques pour l'Apprentissage Humain (EIAH), les approches et les techniques de modélisation des connaissances de l'apprenant dans les EIAH. nous décrirons aussi dans cette section, les cadres théoriques pour la modélisation de l'apprenant en termes de terminologies, modèles et langages, puis nous finirons par l'étude et l'analyse de quelques travaux connexes sur lesquels nous nous sommes basés pour la mise en place de notre proposition.

Dans la troisième section, nous discuterons de notre problématique par rapport à l'état de l'art et l'étude des travaux existants. Dans la quatrième section, nous présenterons notre approche de modélisation et de formalisation de notre ontologie "Diag-K". La cinquième section concernera la validation de notre approche La dernière section sera l'occasion de discuter nos propositions ainsi que nos principales perspectives de recherche.

Chapitre 2

État de l'art

Notre travail se trouve à la confluence de plusieurs domaines : les EIAH, la modélisation des connaissances, le diagnostic des connaissances, etc. Dans ce chapitre, nous nous intéressons plus particulièrement aux différents aspects de la recherche en modélisation de l'apprenant. Après une introduction sur les EIAHs, nous présenterons leurs définitions puis nous aborderons les composantes principales de ces derniers, avant d'en exposer les différentes approches et techniques de modélisation de l'apprenant et celle de l'expert. Par la suite nous présenterons les cadres théoriques pour l'élaboration de modèles de l'apprenant. Enfin, nous étudierons et analyserons quelques travaux sur lesquels nous nous sommes basé pour l'élaboration de notre ontologie.

2.1 Introduction

L'utilisation de l'informatique dans l'éducation remonte aux années 1970 et a donné lieu à de nombreuses recherches relatives à la création du système éducatif. Ces systèmes ont d'abord été appelés " Enseignement Programmé " puis " Enseignement Assisté par Ordinateur " (EAO), ils sont devenus par la suite, les " Enseignements Intelligemment assistés par Ordinateur " (EIAO) avec l'introduction des techniques d'Intelligence Artificielle, pour aboutir à la fin aux " Environnements Informatiques pour l'Apprentissage Humain " (EIAH) et qui est traduit, par rapport aux EIAO, dont la notion de partenariat est entre l'homme et sa machine. Ainsi Tchounikine (2002) [TCH02] définit les EIAH comme des environnements informatiques qui mettent en interaction des agents humains et artificiels, donnent accès à des ressources formatives, et ont pour objectif de suivre et d'accompagner un apprentissage. Plus généralement, il est courant d'employer ce terme pour n'importe quel système combi-

nant l'utilisation de l'ordinateur à une intention didactique.

La recherche en EIAH ne se limite pas à la réalisation logicielle, mais aussi à plusieurs problématiques, notamment celle de la modélisation des connaissances, aussi bien celle du domaine enseigné (où intervient le travail de formalisation) que celles de l'apprenant (notamment grâce à l'analyse de productions d'élèves). C'est un domaine de recherche fortement pluridisciplinaire, où la didactique, la psychologie cognitive, la pédagogie, les sciences de l'éducation et l'informatique sont concernés, soulevant des interrogations et posant des problèmes loin de se limiter à des considérations d'ingénierie.

2.2 Les Environnements Informatiques pour l'Apprentissage Humain (EIAH)

Les Environnements Informatiques pour l'Apprentissage Humain (EIAH) sont des environnements informatiques qui ont pour objectifs de favoriser ou susciter des apprentissages, de les accompagner et de les valider. Un EIAH intègre des agents humains (e.g. élève ou enseignant) et artificiels (i.e., informatiques) et leur offre des conditions d'interactions, localement ou à travers les réseaux informatiques, ainsi que des conditions d'accès à des ressources formatives (humaines et/ou médiatisées) " locales " ou " distribuées " (Tchounikine 2002) [TCH02]. La recherche sur les EIAH est fondamentalement pluridisciplinaire, en appelant à la coopération de différents secteurs de l'informatique (génie logiciel, réseau, la modélisation des connaissances et des interactions, etc.), et des sciences de l'homme et de la société (psychologie, didactique, ergonomie, sciences des langages, sciences de la communication, etc.). De tels environnements ont été mis en place pour modéliser les connaissances de l'apprenant afin d'aider l'enseignant à le suivre, à connaitre ses lacunes, et enfin pour l'évaluer. [wik]

2.2.1 Les types d'EIAH

Dans la littérature, nous pouvons distinguer plusieurs familles d'EIAH, notamment les micromondes, les tuteurs intelligents et les hypermédia adaptatifs. Ces EIAH, qui se révèlent complémentaires pour l'enseignement, peuvent être décrits comme suite :

– Les micromondes caractérisent un environnement permettant l'exploration d'un domaine par l'apprenant. leur rôle est de permettre à l'utilisateur de découvrir par lui même un domaine par la manipulation et l'expérimentation. Ceux sont des systèmes réactifs mais qui ne prennent pas d'initiative, et dans lesquels l'utilisateur peut communiquer avec le logiciel sans autre contrainte que celle de respecter la syntaxe de l'environnement : toute validation du travail de l'apprenant est donc externe (i.e. l'apprenant n'est pas évaluer au sein du micromonde). [Ren05].

– Les Systèmes Tutoriels Intelligents (STI) sont des systèmes actifs, qui peuvent orienter l'interaction, porter un jugement sur les actions et les réponses de l'apprenant et lui proposer des explications afin que les erreurs détectées ne se reproduisent plus. Ceux sont des connaissances sur le domaine, mais à la différence des micromondes, ils disposent d'une représentation interne du problème à traiter et sont donc capable d'évaluer si l'objectif de l'exercice est atteint ou pas [Ren05]. Selon Murray [Mur99], les STI sont des systèmes d'enseignement informatiques qui possèdent un contenu sous forme de base de connaissance (qui spécifie ce qui doit être enseigné), des stratégies d'enseignement (qui spécifient la manière d'enseigner ce contenu) ainsi qu'une connaissance sur le niveau de l'apprenant dans le contenu, afin d'adapter dynamiquement leur enseignement.

– Les systèmes Hypermédias Adaptatives (HA) sont apparus dès la fin des années 1980, et ont pris une importance toute particulière avec l'arrivée d'Internet, et le développement de technologies standards particulièrement adaptées à la conception d'hypermédias (HTML, XML, RDF, etc.). Ceux sont donc des systèmes contenant des documents liés entre eux par des hyperliens permettant de passer automatiquement du document consulté à un autre document lié. Lorsque les documents ne sont pas uniquement textuels, mais aussi audiovisuels, on peut parler de système et de documents hypermédias. Ces systèmes reflètent aussi certains aspects de l'utilisateur dans le modèle de l'apprenant, et utilise ce modèle pour adapter à l'utilisateur différents aspects visibles du système. Les hypermédias éducatifs ont plusieurs types. Les HA classiques où un cours statique est créé par l'enseignant sous forme de pages hypermédias, même cours pour tous les étudiants. Les HA modélisent l'apprenant et adaptent le cours selon son niveau. Les HA dynamiques exploitent des documents multimédias réutilisables et créent dynamiquement le cours selon le niveau de l'apprenant [Beg05].

2.2.2 Les composants d'un EIAH

Une étude exhaustive a été effectuée par Wenger [Wen87], qui définit plus formellement quatre types de modèles à spécifier lors de la conception d'un EIAH : le *modèle du domaine*, le *modèle de l'apprenant*, le *modèle pédagogique* (tuteur), le *modèle de l'expert* et le modèle de l'interaction. Le modèle du domaine représente les connaissances nécessaires pour effectuer les tâches proposées aux apprenants. Le modèle de l'interaction caractérise les modes de communication système-apprenant. Le modèle de l'apprenant ou modèle de l'élève est une représentation des connaissances de l'apprenant, qu'elles soient qualitatives ou quantitatives, formelles ou informelles, et qui rend compte complètement ou partiellement d'aspects spécifiques du comportement de l'élève (Sison Shimura, 1998) [SS98]. Le modèle du tuteur (ou modèle pédagogique) spécifie les dialogues tutoriels et les méthodes de remédiation. Ajoutant à ces quatre types de modèle, le modèle de l'expert qui représente les procédures de l'expert permettant la résolution de problèmes.

Dans ce qui suit, nous allons présenter et détailler les modèles sur lesquels nous nous sommes basé dans notre travail, à savoir le *modèle du domaine*, le *modèle de l'expert* et le *modèle de l'apprenant*.

Modèle du domaine

Cette composante contient les connaissances que l'EIAH souhaite faire acquérir à l'apprenant. Ces connaissances portent sur les connaissance déclaratives constituant le savoir théorique : les notions et concepts du cours, les faits, les règles, les lois, les principes, etc. Elles sont donc constituées de concepts liés entre eux pour former une hiérarchie. Ces connaissances seront acquises, dans un premier temps, en réactivant les connaissances antérieures des élèves. Les notions et concepts importants du cours seront présentés sous forme de cours en ligne, et la résolution de problèmes réalisés par les élèves, dans un deuxième temps sous forme d'exercices ou QCM.

Le modèle de l'apprenant manipule ces connaissances en enregistrant celles qui sont propres à chaque apprenant en particulier et informe sur son niveau d'assimilation. Ces connaissances servent aussi à inférer des solutions de l'apprenant par l'intermédiaire du modèle de l'expert et du modèle d'interaction pour vérifier l'exactitude d'un exercice. Le modèle

du domaine permet donc de générer des explications relatives à la solution de l'expert. En effet, le modèle du domaine doit être en mesure de générer des solutions aux problèmes dans le même contexte que celui où se trouve l'apprenant, afin que les réponses respectives puissent être comparées. Ainsi le système est en mesure de déterminer les différences et correspondances entre les actions de l'apprenant et celles qui sont attendues (i.e. celles de l'expert).[BQCK06]

Ces connaissances ne peuvent servir que dans la mesure où l'on peut les représenter. Les réseaux bayésien et les ontologies sont les mécanismes les plus utilisés pour la représentation de ces connaissances.

Modèle de l'expert

Ce modèle regroupe toute la connaissance nécessaire au processus d'enseignement, on parle de connaissance reliée à l'expertise du domaine ou de procédures de l'expert. En général, un modèle de l'expert doit posséder un savoir-faire, c'est-à-dire une expertise sur la manière de résoudre les problèmes du domaine.

Ce modèle permet donc au système d'acquérir des compétences sur la matière enseignée. Ces compétences sont les procédures, la connaissance du comment de l'action et une suite d'actions associés à des tâches acquises par la pratique. Ainsi, grâce à ce modèle, le système peut : résoudre les problèmes soumis à l'apprenant, suivre le cheminement du raisonnement de l'apprenant et s'adapter aux différentes singularités de raisonnement de chaque apprenant.

Plusieurs formalismes sont utilisés pour la représentation de la connaissance dans un modèle de l'expert tels que les représentations à base de règles (règles de production, logique de premier ordre) ou les représentations en réseaux (réseaux sémantiques, graphes conceptuels, réseaux bayésien, etc.) [Zou05].

Modèle de l'apprenant

Dans [Wen87], Wenger dénit le modèle de l'apprenant comme étant une représentation de tous les aspects liés au comportement et aux connaissances de l'apprenant. Selon Self [J.94], le modèle de l'apprenant doit comporter les informations suivantes : les connaissances de l'apprenant et ses conceptions erronées, ses compétences et son comportement (sa manière d'interagir avec le système). Pour déterminer ce qu'a fait l'apprenant, Self souligne le fait qu'il est nécessaire de posséder un historique des interactions de l'apprenant avec le système. Winkels [Win90] décrit le modèle de l'apprenant comme une structure de données qui reflète l'état des connaissances présumées de l'apprenant concernant un domaine cible (le domaine de l'apprentissage). Greer [Gre94] considère le modèle de l'apprenant comme une représentation abstraite des croyances, des connaissances et des compétences de l'apprenant dans le système, incluant l'historique des actions de l'apprenant pouvant être analysées et interprétées.

2.3 Modélisation des connaissances de l'apprenant dans les EIAH

Dans cette section, nous allons passer en revue les différentes solutions élaborées pour modéliser l'apprenant dans les EIAH. Nous détaillerons dans un premier temps, les approches de modélisation des connaissances de l'apprenant utilisées puis les techniques de diagnostic associées.

2.3.1 Approches de modélisation de l'apprenant

Il existe plusieurs approches de modélisation de l'apprenant dans les EIAH. Sisson et Shimura [SS98] ont classifié ces approches dans deux catégories :

1- Les approches synthétiques qui sont guidées par les données, c'est à dire par les comportements observés de l'apprenant. Ces approches utilisent comme techniques de diagnostic, celles du data mining pour inférer les caractéristiques individuelles de l'apprenant

2- Les approches analytiques dont le modèle de l'apprenant est une structure de données, au sens informatique, qui caractérise pour le système d'enseignement, l'état d'un sous-

ensemble des connaissances de cet apprenant. Ce dernier, va se définir par l'écart entre les propres connaissances de l'apprenant et les connaissances cibles, enjeu de l'apprentissage, telles qu'elles sont représentées dans le système. En ce sens, les approches analytiques, outre les données, sont guidées par les modèles (modèle de l'apprenant, modèle du domaine d'application, modèle de l'expert). Ces approches utilisent en général des systèmes à base de connaissances pour inférer des connaissances de l'apprenant.

Les approches analytiques les plus connues sont l'approche d'expertise partielle (OVERLAY) dans laquelle la connaissance de l'apprenant n'est qu'un sous-ensemble de la connaissance cible (concept du modèle de domaine, connaissance de l'expert) ; et l'approche différentielle (OVERLAY étendu) incorpore des "connaissances fausses", correspondant à des misconceptions (connaissances erronées) de l'apprenant .

L'approche OVERLAY

Dans cette approche, le modèle de l'apprenant consiste en une structure de données comprenant l'ensemble des connaissances et compétences élémentaires à acquérir et précisant pour chacune d'elles, le degré de maîtrise de l'apprenant. Généralement, dans les ITS ces connaissances et compétences sont les connaissances de l'expert, et dans les HA ceux sont le plus souvent les concepts du domaine enseigné.

Cette approche suppose qu'au cours d'un apprentissage, le modèle de l'élève évolue théoriquement d'un modèle initialement « vide » (en supposant que l'élève commence avec une absence totale de connaissance) vers un modèle identique à celui de l'expert, qui correspond à une connaissance parfaite. Entre ces deux extrêmes, chaque élément de connaissance peut être indexer par des valeurs binaires (connu/inconnu) ou par un degré de maîtrise (par exemple sur une échelle allant de 0 à 1). Dès lors, le modèle de l'élève est donc construit en comparant la performance de l'apprenant avec celle que l'expert aurait produite dans les mêmes circonstances. Cette performance de l'apprenant peut être directement observable ou non à partir de son comportement (action, réponse, temps de réponse, taux de réussite, etc.). [DS92]

La mise à jour du modèle de recouvrement repose sur plusieurs sources d'information :

– Les informations implicites découlent de la comparaison entre le comportement de l'élève et les décisions de l'expert. L'acquisition de ces informations dépend de la capacité du concepteur à relier un comportement donné à un ensemble bien défini de réalisations effectives. C'est sur ce point, que la collaboration avec le didacticien peut s'avérer la plus fructueuse (sémantique des actions).

– Les informations explicites peuvent être obtenues par un questionnement direct de l'élève (réponse à un test ou un questionnaire). Dans ce cas là, les réponses de l'apprenant sont comparées aux solutions de l'expert.

– Les pré-requis supposés sont inférés ou stockés lors d'une précédente séance pour initialiser le modèle. Ces informations sont obtenues par des calculs sur les actions et réponses de l'apprenant (temps de réponse à une question, taux de réussite à un exercice,...).

L'approche OVERLAY étendu

L'approche Overlay interprète la connaissance de l'apprenant comme étant une connaissance correcte, mais ne permet pas de prendre en compte des méthodes ou résultats incorrects que l'apprenant peut avoir acquis. Pour élaborer un modèle de connaissances de l'apprenant, il faut donc tenir compte en plus de ses connaissances correctes, les erreurs qu'il puisses commettre et que les chercheurs désignent par le terme "bug". En ce sens, le modèle overlay étendu est donc une extension du modèle d'expertise partielle, et ajoute pour la modélisation de l'apprenant en plus des connaissances de l'expert une librairie de conceptions erronées (bug library).

Le processus utilisé pour la création d'une telle librairie peut être énumératif ou génératif. Dans le premier cas, il s'agit de lister l'ensemble des conceptions erronées possibles suite à une analyse du problème et des erreurs fréquemment commises par les apprenants. Dans le second cas, il s'agit de générer des conceptions erronées à partir d'une théorie cognitive donnée.[Ren05]

L'exemple le plus connu est l'étude des erreurs faites par les élèves dans la soustraction, avec le tuteur BUGGY (Brown et Burton, 78) et l'ensemble des travaux qui en sont issus.

2.3.2 Techniques de diagnostic des connaissances de l'apprenant

Il existe dans la littérature plusieurs techniques de modélisations des connaissances de l'apprenant [Py, Wenger]. Les plus utilisées sont le modèle de traçage et la modélisation à base de contraintes. Ces deux méthodes permettent d'analyser les réponses de l'apprenant dans les systèmes EIAH proposant des problèmes dans le domaine enseigné.

Modèle de traçage (MT)

Certains systèmes implémentent une approche par expertise partielle étendue, mais avec un suivi de l'apprenant plus contraignant : on parle alors de modèle de traçage. Dans cette variante, le système interagit avec l'apprenant à chaque pas de résolution d'un problème, intervenant dès qu'il s'éloigne d'une solution correcte.

Anderson en (1983) a travaillé sur la théorie ACT (Active Control of Trought) et a mis l'accent sur la différence entre la connaissance déclarative (le savoir faire) et la connaissance procédurale (le savoir).

Cette technique de traçage consiste à analyser étape par étape le raisonnement de l'apprenant à partir de règles de production réparties en deux sous-ensembles : règles correctes et règles erronées, et à intervenir immédiatement lorsque l'apprenant effectue ou commet une erreur. Comme exemple pour cette technique, il y a le Système LISP Tutor [Anderson et Reiser 1985]. Il propose un exercice de programmation à l'apprenant qui y répond tout en tapant son programme, à chaque fois que l'étudiant étudie une nouvelle partie de son code, le tuteur tente d'associer cette étape à une des règles de la base. Si la règle associée est incorrecte, alors le tuteur montre et explique l'erreur. Si l'étudiant ne peut pas continuer en suivant une solution correcte alors le tuteur peut lui montrer comment faire avec une démonstration de l'algorithme à programmer ou il lui montre l'étape correcte.[Aux09]

L'avantage de cette technique est de suivre et de guider systématiquement l'apprenant vers l'élaboration d'une solution correcte. Cependant, l'apprenant ne peut pas explorer des chemins corrects mais sous-optimaux, et encore moins des chemins incorrects au bout desquels il se rendra compte de ses erreurs par lui-même.

Autrement dit, en contraignant toutes les possibilités lors de l'utilisation du système, le concepteur limite l'apparition du phénomène du comportement de l'apprenant dus aux enchaînements d'erreurs, mais se prive en même temps des observations de séquences d'actions extrêmement révélatrices des difficultés profondes de l'élève.

De plus, selon Gui (1989), dans un tel contexte d'apprentissage où l'environnement est trop contraignant, il est peu probable que l'apprenant ait l'occasion de développer les aptitudes heuristiques (les « essais-erreurs ») indissociables de toute acquisition opérationnelle de connaissances. [Ren05]

Ces systèmes basés sur ces approches ont l'avantage de pouvoir évaluer l'utilité d'un pas de raisonnement (c'est à dire, pas uniquement les vrai et les faux).

Le modèle de traçage est composé de : règles de l'expert, règles de bogue, un traceur du modèle et une interface utilisateur. Dans ce qui suit, nous allons présenté les règles de l'expert et les règles erronées (BUGGY rule).

Les règles de l'expert

Les règles de l'expert modélisent les étapes qu'un individu compétent peut suivre pour la résolution d'un problème [Aux09]. Ceux -ci inclus :

1- Les règles de planifications : ceux sont des règles pour décomposer un problème sous forme de sous problèmes.

2-Les règles d'exécutions :ceux sont des règles qui attribuent la solution à un sous problème atomique.

La règle de planification exprime la connaissance du domaine procédurale et la règle d'exécution exprime la connaissance du domaine déclarative. [VK06]

Un exemple de règle de planification pour le domaine de géométrie, tiré de Anderson et al. (1995) est fourni dessous :

SI Le but est de prouver que deux triangles sont conformes ALORS Mettre comme un sous-but à prouver que ces parties correspondantes sont conformes.

Les BUGGYs règles

Dans un modèle de traçage (MT), la connaissance du domaine est capturée sous forme de plusieurs règles. Le point crucial d'un MT Tuteur (MTT) est de "tracer" l'entrée de l'étudiant alors que le traçage consiste à trouver une séquence de règle d'exécutions dont le résultat final doit correspondre avec l'entrée de l'étudiant. Si le tuteur est capable de faire le principe de traçage, alors il a compris le processus par lequel l'étudiant est arrivé à une réponse. Pour identifier les erreurs de l'étudiant dans un MTT, un ensemble de BUGGY règles qui reflètent les fausses perceptions communes de l'étudiant. Si la trace de tuteur d'une solution de l'apprenant contient l'application d'une ou plusieurs de ces BUGGY règles alors le professeur fournit la remédiation associé avec les BUGGYs règles .[VK06]

Cette règle est construite sous la forme de : *IF condition THEN action.*

Modèle basé sur les contraintes (CBM)

L'approche basé sur les contraintes (CBM : Constraint Based-Modelling), introduite par Ohlsson [Ohlsson 1994] [S.94], est basée sur la théorie de l'apprentissage à partir des erreurs de performance [Ohlsson 1996] [OS96]. Cette théorie se concentre sur la manière dont les personnes découvrent et corrigent leurs propres erreurs lorsqu'ils exercent leurs compétences. L'hypothèse de base est que les informations nécessaires au diagnostic ne sont pas dans la séquence des actions de l'apprenant, mais dans la situation ou l'état du problème dans lequel l'apprenant arrive. Ohlsson considère qu'une solution correcte ne peut être atteinte en passant par un état du problème qui viole les concepts du domaine. L'analyse des productions et la modélisation de l'apprenant avec les CBM ne se fait donc pas sur les actions de l'apprenant mais sur la solution proposée par l'apprenant.[VK06]

L'approche CBM ne requiert pas l'utilisation d'un module expert exécutable contrairement à de nombreux tuteurs intelligents. C'est un avantage important, car dans beaucoup de domaines, il est très difficile de construire un résolveur de problèmes. Dans l'approche CBM, il n'y a pas non plus de représentation des "connaissances" car seules les contraintes du domaine y sont modélisées : ceux sont des règles locales qu'une solution ou qu'un élément de solution doivent respecter. [Aux09]

Dans l'approche CBM, chaque contrainte a un identifiant et une paire ordonnée (Cr, Cs) où Cr est une condition de pertinence et Cs est une condition de satisfaction :

- La condition de pertinence décrit les états dans lesquels la contrainte peut être appliquée.

- La condition de satisfaction définit les états dans lesquels la condition est satisfaisante.

Une contrainte est ainsi de la forme : *SI Cr est vrai, ALORS Cs devrait être vrai également, dans le cas contraire quelque chose est erroné.*

Les contraintes utilisent des paternes assortis entre la solution idéale et la solution de l'étudiant. Par conséquent l'évaluation d'une solution dans un CBM tuteur devrait être exécutée quand l'étudiant a complété le problème.

2.4 Cadres théoriques de modélisation de l'apprenant dans les EIAH

Nous présentons dans cette section deux cadres de modélisation de l'apprenant dans les EIAH. Le premier cadre définit les notions sous-jacentes aux processus de diagnostic. Le second décrit le processus de diagnostic à partir de l'observation des interactions de l'apprenant avec le système.

2.4.1 Cadre théorique de Dillenbourg et Self

Dillenbourg et Self [DS92] proposent un cadre théorique permettant de lever un certain nombre d'ambiguïtés de langage, en définissant de manière formelle les concepts sous-jacents à la problématique générale de modélisation de l'élève. Le cadre théorique souligne plusieurs objectifs, notamment la définition d'une terminologie consistante et utilisable dans l'axe de recherche de la modélisation de l'apprenant ; l'importance des aspects conceptuels de la modélisation de l'apprenant et la description de plusieurs approches de la modélisation de

l'apprenant en précisant la particularité de chacune d'entre-elles. Nous ne donnerons dans cette partie que les grandes lignes de leur cadre (figure2.1), le lecteur intéressé pourra se reporter à la référence originale pour de plus amples détails.

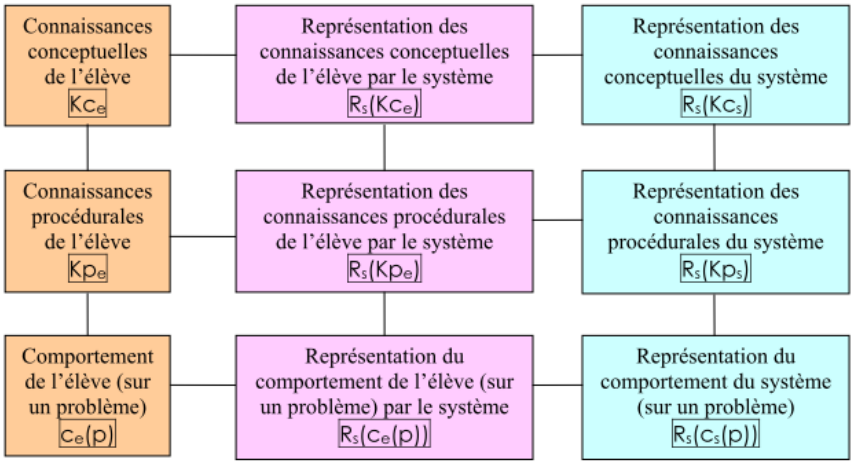

FIGURE 2.1 – Schema récapitulatif du cadre théorique de Dillenbourg

Basé sur l'introduction d'une dimension verticale (relative aux différents degrés de connaissance) et d'une dimension horizontale (afin de différencier le système de l'apprenant), ce cadre général permet d'expliciter tout système ou méthode de modélisation de l'élève. Les deux dimensions constituent le cadre de Dillenbourg et Self où chaque entité de la dimension horizontale possède les trois entités de la dimension verticale.

Dimension verticale

Les auteurs introduisent une dimension verticale afin de distinguer les différents degrés de généralité des connaissances. Cette hiérarchie contient trois niveaux : le comportement définit comme une séquence d'actions effectuées par un agent (humain ou artificiel) pouvant être potentiellement une solution à un problème posé (P). La connaissance procédurale et la connaissance conceptuelle qui rendent compte du comportement observé. Ceux sont les connaissances théoriques (description des concepts du domaine) et stratégiques du domaine (les connaissances de résolution de problèmes).

dimension horizontale

Le cadre théorique prévoit aussi une dimension horizontale, modélisant et mettant en évidence les différences conceptuelles entre le système informatique (s), l'élève (e) et la représentation de l'élève par le système R s (e). Dans le cas simple où le problème à résoudre par l'apprenant n'admet pas plusieurs résolutions correctes, tout écart entre ce que propose l'élève et ce que ferait le système dans la même situation pourrait être considéré comme une erreur.

2.4.2 Cadre théorique pour l'élaboration du modèle de l'apprenant à partir des traces d'interactions modélisées

Ce cadre théorique décrit le processus d'élaboration du profil d'apprenant à partir des traces d'interactions issues d'EIAH supportant une situation d'apprentissage individuel (figure 2.2). Il inclut des modèles de connaissances permettant de représenter explicitement les observations sur les interactions de l'apprenant avec un EIAH, et d'exprimer les connaissances nécessaires pour interpréter les traces d'interaction en vue de calculer et/ou de déduire les éléments du profil de l'apprenant qui sont indépendants du domaine.

L'approche d'élaboration de profils d'apprenant à partir des traces d'activités décrites dans la figure2, s'effectue en trois étapes. Pour chaque étape le concepteur doit définir les modèles nécessaires en utilisant les vocabulaires RDF proposés au sein de cette architecture.

Processus de création d'une trace modélisée

La première étape est le processus de création d'une Trace Modélisée pour un EIAH donné qui est effectuée à partir des éléments observés issus de cet EIAH et du modèle de traces d'activités qui les décrit ($LATM_{TEL}$). La trace modélisée est définie comme une séquence d'éléments observés et temporellement situés, enregistrés à partir des interactions de l'apprenant avec le système durant une session d'apprentissage, et associés explicitement à un modèle de trace (Settouti et al, 2009a) [SPMM09]. Dans cette étape,avant la création de la trace modélisée, le concepteur de l'EIAH doit d'abord définir le modèle de traces issues de cet EIAH ($LATM_{TEL}$). Pour ce faire, le concepteur utilise le vocabulaire LATM proposé dans ce cadre. LATM est une sorte de méta-modèle (une ontologie RDF) décrivant

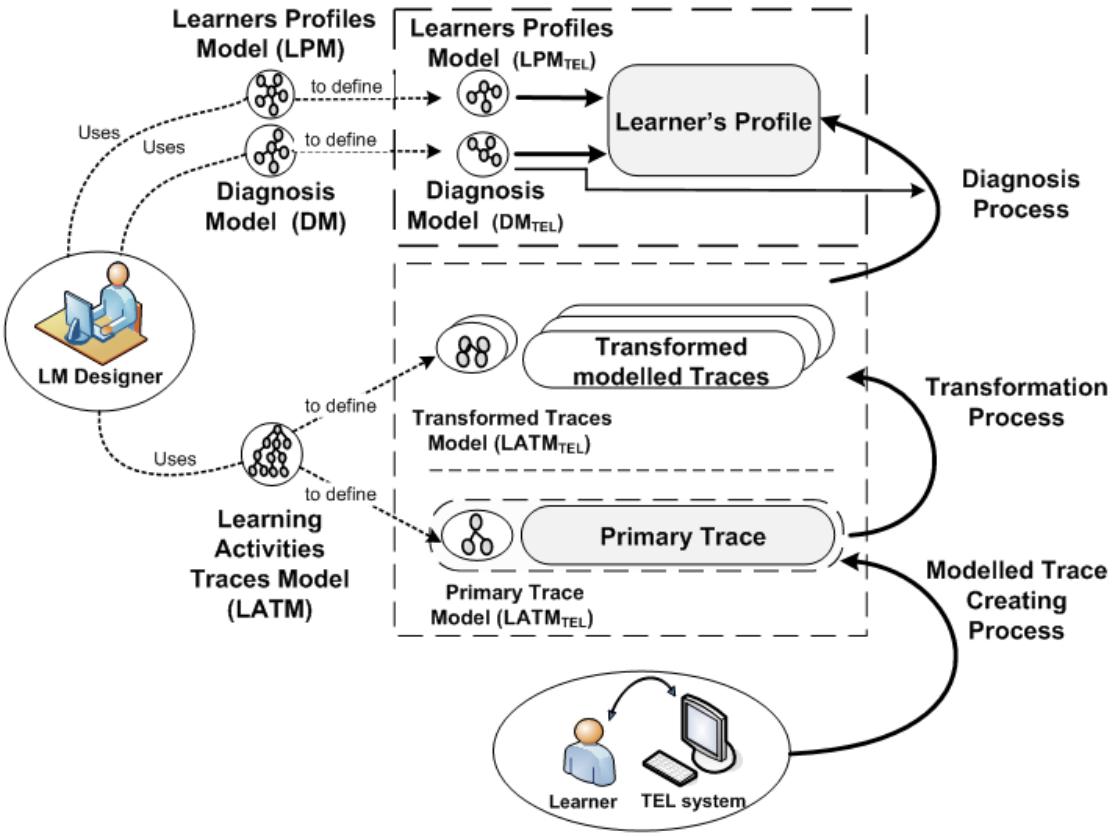

FIGURE 2.2 – Architecture de modélisation de l'apprenant basée sur les traces

et structurant explicitement les traces d'activités issues de tout système informatique, y compris les EIAH. Le vocabulaire de ce méta-modèle permet de décrire les types d'éléments observés, les types de relations entre les éléments observés et les attributs décrivant les éléments observés.

Ce méta-modèle fournit donc un vocabulaire riche qui sert comme outil pour décrire et donner du sens aux traces. Il servira par la suite de support pour les différents traitements (sélection, interrogation, transformation, calcul, inférence) permettant d'exploiter et de manipuler les éléments des traces modélisées en vue d'obtenir des informations sur les performances et les stratégies de l'apprenant.

Processus de transformation

La deuxième étape est le processus de transformation d'une trace modélisée qui est nécessaire lorsque le concepteur de l'EIAH estime que les observés assemblés dans la première trace ne sont pas ou peu significatifs pour effectuer le diagnostic. Le concepteur pourra donc exprimer des règles de transformation en utilisant le langage de transformation du cadre

des SBTm (Settouti et al, 2009a)[SPMM09], pour interpréter et améliorer la trace première, produisant ainsi une nouvelle trace modélisée.

Processus de diagnostic

La troisième et dernière étape représente le processus de diagnostic, qui consiste à instancier le modèle de l'apprenant pour obtenir le profil de l'apprenant interagissant avec l'EIAH, à l'aide des traces modélisées, du modèle de profils d'apprenants et du modèle de connaissances de diagnostic.

Modèle de profils d'apprenants (LPM)

C'est un méta-modèle que le concepteur de l'EIAH utilise pour définir les informations sur l'apprenant qu'il souhaite calculer et / ou inférer à partir des éléments de la trace, tels que des indicateurs de son comportement durant la session d'apprentissage (temps moyen pour résoudre un exercice, taux de réussite à un exercice, etc...) et ses stratégies de résolution de problèmes. Pour le moment, Ce méta-modèle ne permet pas de décrire des estimations qualitatives ou quantitatives du niveau de connaissances de l'apprenant.

Modèle de diagnostic (DM)

Le concepteur utilise ce méta-modèle pour définir le modèle DM_{TEL}. Le méta-modèle DM est une représentation explicite des requêtes et règles SPARQL et des seuils de comparaison d'indicateurs de comportement, et permettra au concepteur du diagnostic de l'EIAH de décrire celles qu'il juge pertinentes pour calculer ou pour inférer une performance et/ou une stratégie de l'apprenant du profil de 'apprenant qu'il souhaite obtenir.

En conclusion, ce cadre contrairement à celui de Self ne définit pas le comportement, et les connaissances de façon générale mais plutôt il permet de les expliciter formellement dans le systèmes moyennant les vocabulaires RDF. Aussi, les méta-modèles proposés permettent de représenter les interactions de l'apprenant avec le système (traces modélisées), comportement observé de l'apprenant (les données calculés à partir des traces (performances) et les stratégies de l'apprenant), les connaissances permettant de les déduire (seuils, patterns de comportements (règles SPARQL)), mais il ne décrit pas les connaissances de diagnostic permettant d'élaborer le modèle de connaissance de l'apprenant.

2.5 Travaux existants sur la modélisation de l'apprenant

Dans cette section, nous présentons quelques travaux de recherche que nous avons étudiés lors de notre revue de l'existant en termes de modèle d'apprenants, approches de modélisation de l'apprenant et techniques de diagnostic. Nous en proposons une synthèse à la fin de cette section.

(1) Les travaux de Mitrovic et Al [Mit03] tournent autour de deux EIAH : SQL-Tutor [Mit03] et E-KERMIT [Har02].

- Le système SQL-Tutor vise l'apprentissage du SQL pour des étudiants d'université. Les auteurs l'ont utilisé dans divers études liées à l'autoévaluation des apprenants. SQL-Tutor est un système à base de contraintes et la représentation du modèle de connaissance de l'apprenant sous-jacent est un modèle Overlay. SQL-Tutor est composé de plus de 700 contraintes. Dans son projet de "Tuteur SQL Intelligent dans le WEB", Mitrovic a présenté le SQLT-WEB comme étant une version supporté par le web pour les débutants, un standard du Système Tutoriel Intelligent (STI) et a montré que le système est efficace et valide. Elle a décrit comment les composants du système standard sont réutilisés pour les systèmes supportés par le WEB. Aussi, elle a présenté et décrit l'architecture de son système en comparant son architecture avec d'autres existantes.

- Le système E-KERMIT (Knowledge-based Entity Relationship Modelling Intelligent Tutor) [Hartley et al. 2002] est un système informatique qui vise l'apprentissage de la conception de base de données utilisant le modèle entité-association. Le système est basé sur la modélisation à base de contraintes (Constraint-Based Modelling). Le modèle de l'apprenant dans E-KERMIT est généré par le module à base de contraintes. E-KERMIT contient un ensemble de problèmes et de solutions idéales prédéfinies. Afin d'évaluer la solution de l'élève, le système compare la solution de ce dernier avec sa propre solution, et la juge correcte ou pas. Les connaissances du modèle d'expertise sont représentées sous la forme de 90 contraintes.

(2) Dans son travail de thèse, Renaudie [Ren05] a présenté un travail ayant pour objectif "la modélisation cognitive de l'élève en Algèbre", en se basant sur le Micromonde APLUSIX pour aboutir à la construction d'un système intelligent. Pour se faire, il a suivit l'approche Overlay etendu pour la représentation du modèle de connaissance de l'apprenant et a construit une librairie de bug en se basant sur l'analyse manuelle des informations relatives aux exercices proposés à l'élève par le systèm APLUSIX. Pour le diagnostic des connaissances, il s'est basé sur le modèle de traçage sous forme de règles erronées (buggy règle).

(3) Dans les systèmes Hypermedias Adaptatifs, nous avons vu les travaux de Arij, Brusilovsky et celui de Iclass.

- Selon peter Brusilovsky [BM07], en général, la représentation du modèle de l'utilisateur dans les Hypermédias Adaptatives (HA) se base sur l'approche par recouvrement (OVERLAY) et le modèle de l'utilisateur est constitué de ses connaissances, ses intérêts, les buts, background et ses caractéristiques individuelles.

  Dans ce type de système, la représentation du modèle de domaine est faite selon : le domaine, le champ d'application et le choix du designer, ainsi que les concepts du domaine enseigné qui peuvent représenter les plus grandes ou les plus petites parties du domaine de connaissance.

- Le projet Iclass [Mue09] [1] a pour but d'adapter la présentation de l'objet d'apprentissage (LO) aux besoins de l'utilisateur, le système Iclass suit les traces des connaissances de l'apprenant. la modélisation de l'apprenant est faite en tenant compte des traces de ce dernier. Le système de modélisation de l'apprenant comprend trois composants : Tracker, Profiler et Moniteur. ils sont définis respectivement comme suit : celui qui reçois, stock, et distribue les informations, celui qui identifie les performances de l'apprenant et sa culture et celui qui crée et met à jour la surveillance de l'interaction.

1. Le projet Iclass est un projet européen intégrant plusieurs universités de différents pays. Ce projet a été développé en prenant en compte l'aspect multiculturel et multilingue de ces pays.

Le système Iclass dans la modélisation de l'apprenant a besoin de stocker le suivit des traces du domaine de connaissance de l'apprenant et d'avoir un accès sur ces connaissances pendant une longue durée. Le modèle de l'apprenant est constitué de ses compétences, connaissances, habilités, et historique de ses intervention avec les systèmes.

L'architecture du modèle de l'apprenant que le système Iclass a proposé, utilise la notion de Publisher/Listener. Il définit l'événement de l'apprenant pour le différencier des autres événements du système. L'événement de l'apprenant est composé de : l'apprenant, le TimesTamp [2] et le contexte.

Le modèle de connaissance de l'apprenant est basé sur un niveau de connaissance pour les concepts pertinents. Il suppose que la plupart des suppositions valides sur la connaissance d'un apprenant peuvent être obtenues en dirigeant la performance de l'apprenant dans les exercices interactifs.

Le système incorpore un modèle de l'apprenant très compliqué qui entrepose de l'information explicite concernant : les matières pédagogique que l'apprenant a étudié et sa connaissance approfondie sur ces matières . La connaissance approfondie est représentée comme un triplet de : connaissance, compréhension, et application dont les valeurs représentent un sous-ensemble de l'objectif pédagogique.

L'évaluation de niveau de connaissance est quantitative (valeur entre 0 et 100) d'un concept et est faite selon un triplet (K,C,A) où "K" est la connaissance, " C " est la compréhension et " A " est l'application. La mise à jour du niveau de connaissance d'un concept est faite en tenant compte des paramètres suivants : la durée de résolution, le nombre d'itération et la difficulté d'exercice.

– Les travaux de Aarij Mahmood HUSSAAN [HUS08] vise à concevoir un EIAH adaptatif capable de mettre à jour les connaissances du domaine et le profil de l'apprenant, et de proposer des scénarios pédagogiques adaptés au profil et aux besoins de chaque apprenant.

2. TimesTamp mesure quand est ce que l'événement se produit

Ils ont proposé une architecture générale afin d'apporter une souplesse et une évolutivité dans l'accompagnement individualisé, dont l'EIAH doit permettre le suivit de l'évolution de chaque apprenant pour lui proposer des scénarios adaptés à ses besoins et ses compétences. Pour cela, l'architecture inclut des boucles de contrôle, aussi bien en temps réel pour l'adaptation du scénario en fonction du comportement de l'apprenant qu'en off-ligne pour la mise à jour des connaissances du domaine et du profil de l'apprenant à partir de l'analyse des traces d'interaction.

Cette architecture inclut six modules : domaine de connaissance, le profil de l'apprenant, le raisonnement, la gestion interactive, l'interaction des traces et le module de gestion de trace.

l'expert du domaine alimente le système par des connaissances du domaine et caractérise le profil de l'utilisateur. Durant l'interaction entre l'apprenant et le système, toutes les actions de l'apprenant sont récupérées,puis représentées sous formes de traces modélisées. Ces traces sont représentées sous forme d'observables, comme suit : Oi(O1,O2,..,On) et chaque observé Oi est caractérisé par 5-uplet : O<Q,R,T,S,E>, où : Q : Exercice/Problème, R : Réponse de l'apprenant, T : Temps de réponse, S : Réponse juste de la question (réponse de l'expert) et E : Evaluation de la réponse. Cette derniére est une fonction f qui calcul l'écart entre la réponse de l'apprenant et la réponse juste de l'expert, et prend en compte la durée T.

L'idée principale de leur approche est d'organiser les connaissances du domaine sous formes d'un réseau Bayésien. Le profil de l'apprenant à été représenté sous forme d'un vecteur de composant <attribut, valeur> où les attributs sont des variables du réseau (concepts des connaissances du domaine) et les valeurs correspondent aux probabilités des variables du réseau (l'estimation du niveau de connaissance de l'apprenant pour le concept en question).

– Ulrich [CU09] présente un générateur de cours adaptatif. Le but de son travail est de produire ou générer des cours adaptés pour la compétence d'un utilisateur individuel et faire l'étude des buts qu'utilisent l'organisme. Ils utilisent le modèle de l'apprenant et le modèle du domaine. Le modèle du domaine est structuré en utilisant différentes

relations pédagogiques. Ils sélectionnent la ressource d'apprentissage qui est adaptée le mieux à l'utilisateur en utilisant les valeurs de propriétés différentes dans le profil de l'utilisateur.

En conclusion, nous constatons que chaque système a élaboré son propre modèle de connaissance qui est spécifique au domaine enseigné. Par la suite, nous nous baserons sur l'ensemble de ces travaux pour définir le méta-modelé des connaissance de diagnostic. Le mèta-modèle, une fois construit, doit prendre en considération les cas de figures du modélisation de l'apprenant dans différents EIAH, à savoir :

– Modélisation à base de règles : c'est une combinaison de l'approche OVERLAY avec la technique de diagnostic "modèle de traçage".
– modélisation à base de règles erronés (BUGGY) : c'est une combinaison entre l'approche OVERLAY étendu et la technique du modèle de traçage.
– Modélisation à base de contraintes : c'est une combinaison entre l'approche overlay avec la technique basé sur les contraintes (CBM).

Pour résumer, ci-dessous un tableau récapitulatif de l'ensemble des travaux cités précédemment. Le tableau dresse une liste exhaustive des connaissances nécessaires d'interprétation des traces pour la modélisation de l'apprenant avec l'approche et la technique utilisées pour chaque système.

TABLE 2.1 – Tableau des connaissances nécessaires à inscrire dans le système pour déduire le modèle des connaissances de l'apprenant

EIAH	Approche de modèlisation	Technique de diagnostic	Connaissance à inscrire dans le système
E-Kermit	Approche par recouverement (OVERLAY)	Modèle basé sur les contraintes (CBM)	Modèle de l'expert basé sous forme de 90 contraintes avec des solutions prédéfinies
SQL-Tutor	Approche par recouverement (OVERLAY)	Modèle basé sur les contraintes (CBM)	Modèle de l'expert sous forme de 700 contraintes
GUIDON	Approche par recouvrement (OVERLAY)	Modèle basé sur des règles (tracing model)	Modèle de l'expert sous forme de règles
ANDES	Approche par recouvrement étendu (OVERLAY etendu)	Modèle basé sur des régles correcte et erronés (tracing model)	Modèle de l'expert sous forme de 540 règles avec des solutions prédéfinies
APLUSIX	Approche par recouvrement étendu (OVERLAY etendu)	Modèle basé sur les Buggys règles	Modèle de l'expert sous forme de règle avec une librairie de bug
C-Polmile	Approche par recouverement étendu (OVERLAY etendu)	Modèle basé sur les BUGGYs règles	Modèle de l'expert sous forme de règle avec une librairie de bug
Iclass	Approche par recouvrement (Overlay)	Modèle basé sur des règles	Modèle de l'expert sous forme de règles

2.6 Conclusion

Dans ce chapitre nous avons présenté les Environements informatique d'Apprentissage Humain (EIAH) avec leurs approches de modèlisation de l'apprenant et les techniques de diagnostic. Ensuite, les cadres thèoriques de Dellenbourg et Self, et celui de SETTOUTI. De même, les travaux existants sur la modèlisation de l'apprenant et en finissant avec un tableau comprenant les connaissances nécessaires à inscrire dans le système pour déduire le modèle de connaissances de l'apprenant. Dans ce qui suit, nous allons présenté en détail notre ontologie.

2.7 Formulation de notre problématique

Le cadre définit pour la modélisation de l'apprenant à partir des traces dans le cadre d'un méta modèle de diagnostic tel qu'il est définit ne tient pas compte de toutes les interprétations des traces. En effet, les connaissances d'interprétations telles qu'elles sont définies permettent juste d'interpréter les éléments de traces afin d'inférer des éléments d'un profil de l'apprenant indépendant d'un domaine (par exemple : ses performances, ses stratégies,...), et les connaissances permettant d'interpréter ces éléments du profil sont l'ensemble de patterns de requête définit dans ce cadre et les seuils définis par l'enseignant. Mais la déduction des éléments du profile dépendant du domaine tel que l'état de connaissance de l'apprenant n'a pas été élaborée.

Pour répondre à cette problématique, notre travail consiste à définir une liste exhaustive des connaissances d'interprétations des éléments dépendants du domaine et de les formaliser (modéliser). En ce sens, nous avons proposé "Diag-K", une ontologie générale (méta-modèle) permettant de structurer la représentation d'un modèle de domaine et d'un modèle de l'expert. Le concepteur de l'EIAH pourra utiliser notre ontologie pour spécifier le modèle du domaine enseigné via cet EIAH et/ou l'expertise se résolution de problèmes. Pour ce faire, nous avons utilisé la méthode « méthontology » pour décrire notre ontologie. Pour valider le travail, nous avons pris à titre d'exemple le « CISCO ». Dans ce qui suit, nous allons définir et détailler l'ontologie puis l'exemplifier.

Chapitre 3

Construction de l'ontologie Diag-K

3.1 Introduction

Notre approche consiste à définir une ontologie générale (vocabulaire RDF) permettant de décrire les connaissances nécessaires à l'interpretation des interactions de l'apprenant avec l'EIAH. Cette ontologie permettera de calculer et d'inférer des informations sur l'état de connaissances de l'apprenant sur n'improte quel domaine d'enseignement spécifique (Math, physique, programmation, etc). Cette ontologie est composée de deux parties :

– Le modèle du domaine qui regroupe les connaissances permettant de déduire l'état des connaissances déclaratives de l'apprenant

– Le modèle de l'expert permet d'inférer des informations sur les procédures (connaissances procédurales) adoptées par l'apprenant pour la résolution de problèmes.

Dans la section suivante, avant d'entamer la description de cette ontologie, nous décrivons brièvement la méthode sur laquelle nous nous sommes basé pour représenter et construire l'ontologie "Diag-K".

3.2 Méthodes d'ingénieurie ontologique

Selon Mendes et *al* [MC06], le processus de construction d'une ontologie est assez complexe, impliquant plusieurs intervenants dans les différentes phases du processus. La gestion de cette complexité exige la mise en place de processus de gestion, afin de contrôler

les coûts et le risque, et d'assurer la qualité tout au long du processus de construction. Pour ce faire, il est fort utile d'employer des méthodes de développement pour assister le processus de construction des ontologies. Néanmoins, le degré de maturité des processus de développement d'ontologies n'a pas encore atteint celui de l'ingénierie des connaissances et encore moins le niveau atteint par le génie logiciel.

À l'heure actuelle, il n'existe pas encore de consensus à propos des meilleures pratiques à adopter lors du processus de construction ou même des normes techniques régissant le processus de développement des ontologies, bien que certaines contributions dans cette direction soient déjà disponibles dans la littérature [Psy04].

Les méthodologies recensées permettant de différentes façon de construire une ontologie sont :

- à partir du début.
- par intégration ou fusion avec d'autres ontologies.
- par re-ingénierie.
- par construction collaborative.
- par l'évaluation des ontologies construites.

Ces méthodes peuvent porter sur l'ensemble du processus et guider l'ontologiste dans toutes les étapes de la construction. ENTERPRISE (Uschold King, 9) [M.G96] , TOVE (Grüninger et Fox, 95) [Gru95] et METHONTOLOGY (Fernandez-Lopez et al., 99) [MFL99] sont les méthodes les plus représentatives pour construire des ontologies, mais la plus recommandé et utilisé c'est la méthode METHONTOLOGY vue son recouvrement pour tout le cycle de vie d'une ontologie et sa complétude , et cela nous emmène à la suivre et l'utiliser pour la construction de notre ontologie.

3.2.1 La méthode METHONTOLOGY

Cette méthode a été proposée par Gómez-Pérez et *al* 1999 [A.99]. C'est une méthode de construction d'ontologies itérative et manuelle développée au laboratoire d'intelligence artificielle de l'université Polytechnique de Madrid. Elle a été utilisée pour la construction d'ontologie à partir de zéro aussi bien que pour la réutilisation d'ontologies existantes.

La structure de cette méthodologie repose sur la norme 1074-1995 recommandée par l'IEEE dans le développement des produits logiciels et sur les méthodologies de l'ingénierie des connaissances.

Cette méthode est complète et couvre tout le cycle de vie d'une ontologie, et vise la construction d'ontologie au niveau de connaissance à l'aide d'un modèle conceptuel intermédiaire, sans nécessiter une connaissance a priori de concepts. Cette méthode comprend l'identification du processus de développement de l'ontologie avec : activités de gestion de projets, activités orientées-développement et activités d'aide, et un cycle de vie de l'ontologie basé sur des prototypes évolutifs.

L'approche METHONTOLOGY distingue les étapes suivantes :

1- Cadrage : Cette étape, consiste à cerner l'étendue de l'ontologie et le domaine à prendre en compte.

2- Conceptualisation : Le but de cette étape est d'identifier et de structurer les connaissances du domaine en utilisant un ensemble de représentations intermédiaires semi-formelles (des tables et des graphes), faciles à comprendre par les experts du domaine et qui sont indépendants du formalisme à utiliser pour représenter l'ontologie.

3- Implémentation : Cette étape consiste à formaliser le modèle conceptuel obtenu dans l'étape précédente par un formalisme de représentation d'ontologie. Puis, coder l'ontologie dans un langage d'ontologie formel.

3.3 Construction de l'ontologie des connaissances de diagnostic "Diag-K"

3.3.1 Cadrage

Domaine et contexte d'utilisation de l'ontologie

Nous souhaitons construire une ontologie générale tenant compte des différentes notions sous-jacentes à la modélisation de l'apprenant dans les EIAH. Nous nous intéressons tout particulièrement à formaliser les différentes connaissances à inscrire dans le système pour élaborer le modèle de connaissance de l'apprenant.

Les connaissances à formaliser portent entre autres sur la structure du modèle d'un domaine enseigné et la structure des procédures de résolution de problèmes de l'expert de ce domaine, selon les différentes approches et techniques de modélisation des connaissances de l'apprenant dans les EIAH.

L'ontologie que nous proposons est donc une sorte de méta-modèle que le concepteur d'un EIAH pourra utiliser pour définir le modèle de l'expert et ou le modèle du domaine enseigné via cet EIAH.

3.3.2 Conceptualisation

Construction du glossaire des Concepts (Termes)

La définition des concepts de notre ontologie a été faite sur la base de plusieurs travaux qui ont définit des ontologies qui représente un domaine spécifique (chimie, physique, mathématique,..) de connaissance ([CU09], [HUS08], [Mue09], [BM07], [Mit03],[HDN04]) . Nous nous somme inspiré beaucoup plus sur l'ontologie de ULRICH [Ull05], qui décrit la ressource instructionnelle lié au concept.
la signification de chaque terme est donnée dans les tableaux suivants :

1- Modèle de domaine

Termes	Descriptions
Concept	Un concept est un nom général qui peut dénoter un fragment de connaissance [BM07]. La classe " Concept " est la classe racine de l'ontologie du modèle de domaine. Elle fait parti des objets instructifs qui décrivent les pièces centrales de la connaissance. Bien que les concepts ne sont pas nécessairement des instructions spécifiques parce qu'ils couvrent des types de connaissance en général, ils sont inclus dans l'ontologie parce qu'ils sont nécessaires pour l'instruction [Ull05].Le plus souvent, l'assimilation d'un concept dépend de l'assimilation d'un ou de plusieurs autres concepts. En ce sens, nous définissons la relation "is perequied for" entre les concepts. Et nous avons définit deux propriétés " Based Notion" et " Advanced Notion " pour déterminer la notion du concept.
Fact	Une classe représentant un objet instructif qui fournit de l'information basée sur les vrais événements, il décrit un événement ou quelque chose qui influence sans être une règle générale.[Ull05]
Definition	Une classe représentant un objet instructif qui déclare la signification d'un mot, d'une expression, ou d'un symbole. Souvent, elle décrit un ensemble de conditions ou circonstances qu'une entité doit accomplir. [Ull05]
Law	Une classe représentant un objet instructif qui décrit un principe général entre des phénomènes ou des expressions qui ont été prouvées pour se tenir, ou sont basées sur l'expérience cohérente. [Ull05]
Demonstration	Une classe représentant un objet instructif qui consiste en une situation dans laquelle montre qu'une loi spécifique se tient [Ull05]. c'est un raisonnement qui se prouve avec évidence.
Learning_Object	La classe " Objet d'apprentissage " est comme toute entité numérique matérialisée par l'EIAH et utilisée par l'apprenant lors de son activité d'apprentissage. La description de cette classe peut être enrichie par des méta-données (Lom, dubin Core).

Suite du tableau

Technical_Object	Une classe qui décrit l'objet d'apprentissage et le genre de type associé à la ressource instructionnelle pour sa présentation lors de la formation, l'apprentissage ou l'évaluation. Il peut être représenté par une image fixe ou animée, un son, un multimédia, etc.
Instructional_Object	La classe " Ressource instructionnel " c'est toute entité (unité de contenue significative) numérique , utilisée dans un processus d'enseignement, de formation ou d'apprentissage. Cette unité de contenu peut être produite, acquise, assemblée, modifiée et réutilisée, grâce à un ensemble de spécifications communes, afin de construire des unités d'apprentissage plus ou moins complexes comme un module, une leçon, une évaluation, un cours. Elle est considéré comme un objet d'apprentissage quand des méta-données y sont associées.
Course	Une classe représentant une ressource pédagogique numérique permettant de définir la structure d'une leçon (par exemple introduction, conclusion, rappel, remarque, etc.)
Report	Une classe représentant une ressource pédagogique numérique relative au compte rendu d'un sujet donné ou d'une activité pédagogique.
Reading_Note	Une classe représentant une ressource pédagogique numérique permettant de servir à annoter n'importe quel document (i.e. un texte , une leçon ,etc).
Examples	Une classe représentant une ressource numérique permettant d'illustrer un concept.
Interactive_Exercice	Une classe représentant un élément interactif qui exige le retour d'information de l'apprenant (exercice, QCM, etc). Le retour d'information peut être évalué (automatiquement ou manuellement) et une proportion de succès peut y être assignée.[Ull05]

TABLE 3.1 – Tableau des termes de l'ontologie

2- Modèle de l'expert

Termes	Descriptions
Expert Procedure	Une classe qui est constituée de règles et de contraintes par lesquelles on suit et on évalue l'apprenant. Elles sont utilisées pour déduire les connaissances procédurales et déclaratives de l'apprenant. Elle analyse les réponses de l'apprenant afin de mettre à jour le modèle de l'apprenant.
Tracing model	Une classe permettant de comparer les étapes effectuées par l'apprenant et les étapes existantes dans les règles procédurales définies dans le modèle du domaine. Cette approche est un processus centré sur chaque étape qu'effectue l'apprenant.
Expert rule	Une classe qui contient les étapes qu'un apprenant compétent pourrait prendre ou suivre pour résoudre le problème en question. Elle est composée de deux règles : les règles de planification et les règles exécutives.
Planification rule	Une classe qui permet d'inclure des règles pour décomposer le problème en un sous problème ce qui mène à avoir des règles composées de " but " et " sous-but". Cette règle exprime le domaine de connaissance procédurale.
Execution rule	Une classe qui permet d'adresser la solution de sous problème atomique. Cette règle exprime le domaine de connaissance déclaratives.
Buggy rule	Une classe qui permet de définir la liste des erreurs observables, et reflète les causes de comportement de l'étudiant inexact.
Condition	Une classe qui est la partie primitive de la clause IF, elle contient des étapes à suivre pour la réalisation de la condition.
Action	Une classe qui est la partie décisive de la clause THEN, elle contient les buts et les objectifs à atteindre.

Suite du tableau

Constraint based model	Une classe qui est basée seulement sur l'état courant de la solution auquel l'étudiant en est parvenu, sans tenir compte des étapes que l'étudiant à mis pour arriver à la solution.
Constraint	Une classe qui spécifie certaines conditions qui devront être satisfaisantes selon toutes les solutions correctes. S'il y a violation de cette contrainte , une erreur se déclenche.
Relevant_condition	Une classe qui spécifie quand la contrainte est pertinente.
Satisfaction_Condition	Une classe spécifiant une condition qui devrait être satisfaisante pour toute solution correcte, la condition de pertinence.
Expert_Solution	Une classe qui contient un ensemble de solutions pour n'importe quel problème.

TABLE 3.2 – Tableau des térmes de l'ontologie (suite)

Construction des classes de l'ontologie

– Hiérarchie 1 : Les classes constituant le premier niveau de notre ontologie Diag-K sont respectivement : Learning_Object (LO) , Concept , Expert Procedure et Expert Solution . Ces classes sont reliées avec " Domain Knowledge " qui est une classe abstraite représentant toutes les connaissances du domaine que nous avons pu identifier, par la relation "Composed_Of". (voir figure 3.5) Dans le même niveau :

– La solution de l'expert est constituée d'un ensemble de solutions. Nous définissons la relation "Is_a_set_of" pour lier les deux classes "Expert solution" et "Solution".
– Le concept enseigné est représenté au sein d'un objet pédagogique. Plus précisément, dans l'objet instructif. Nous définissons la relation "is_represented into" pour lier les deux classes "Concept" et "Instructif_Object".
– Le concept a deux relations réflexives : "Has_a_SubClassOf" et "Is_prerequieste For". Aussi, il contient deux propriétés "Is a Base Notion" et "Is a Advanced Notion" de type : datatype.

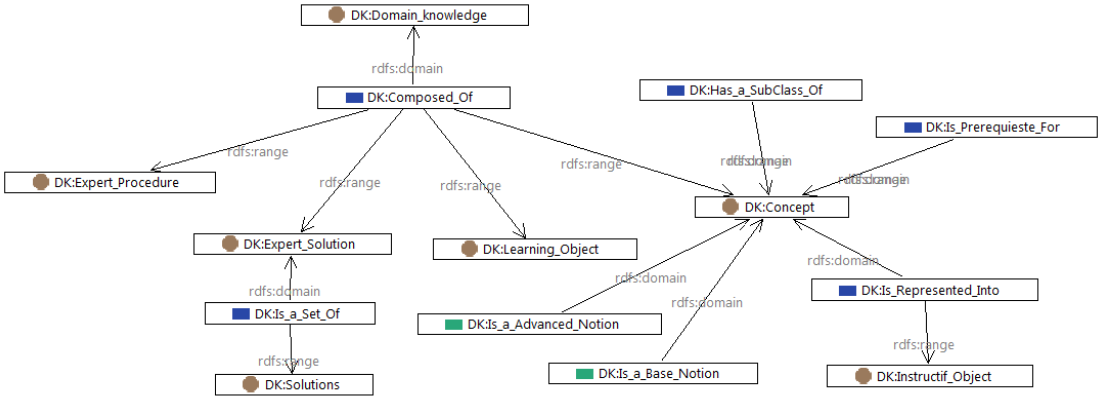

FIGURE 3.1 – La classe "Domain-Knowledge" et ses relations

– Hiérarchie 2 : Le deuxième niveau de notre ontologie est constitué des différentes classes et relations relatives à chaque classe de niveau supérieur (i.e. Concept, Learning_Object (LO), Expert procedure). Nous allons définir dans ce qui suit, ces différentes classes et les relations entre ces classes.

1- Les classes et relations reliées à la classe " Concept " (voir figure 3.2) :

Un concept peut être sous forme (une sorte de) de faits, de définitions, de lois et/ou de démonstrations. Donc, à ce niveau, nous avons une hiérarchie de quatre sous-classes de la classe "Concept", qui est représentée par la relation "rdfs :SubClassOf".

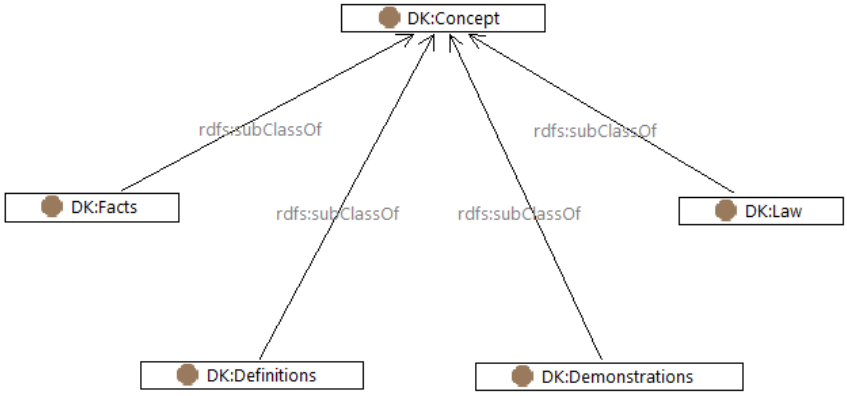

FIGURE 3.2 – Les sous classes de la classe "Concept"

2- Les classes et relations reliées à la classe "Learning_Object" (voir figure 3.3) :

L'objet pédagogique ou d'apprentissage est constitué de deux sous classes : objet instructif et objet technique. La relation qui lie l'objet d'apprentissage à l'objet instructif et l'objet technique est "rdfs :SubClassOf".

L'objet instructif a un support technique. Nous définissons la relation "Has a Support" pour lier entre la classe "Instructif_Object" et "Technical_Object".

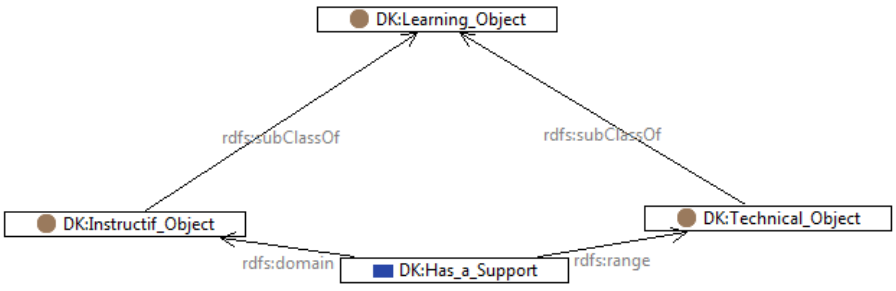

FIGURE 3.3 – Les sous classes de la classe de "Learning_Object"

3- Les classes et relations reliées à la classe " Expert Procedure" (voir figure 3.4) :

La procédure de l'expert peut être soit une règle du modèle de traçage, soit une règle basée sur les contraintes. A ce niveau, nous avons une hiérarchie de deux sous-classes de la classe "Expert Procedure" qui est représentée par la relation "rdfs :SubClassOf". Dans le même niveau, le modèle basé sur les contraintes est constitué de contraintes. Nous définissons la relation "Constitue Of" pour lier les deux classes "Constraint based Model" et "Constraints".

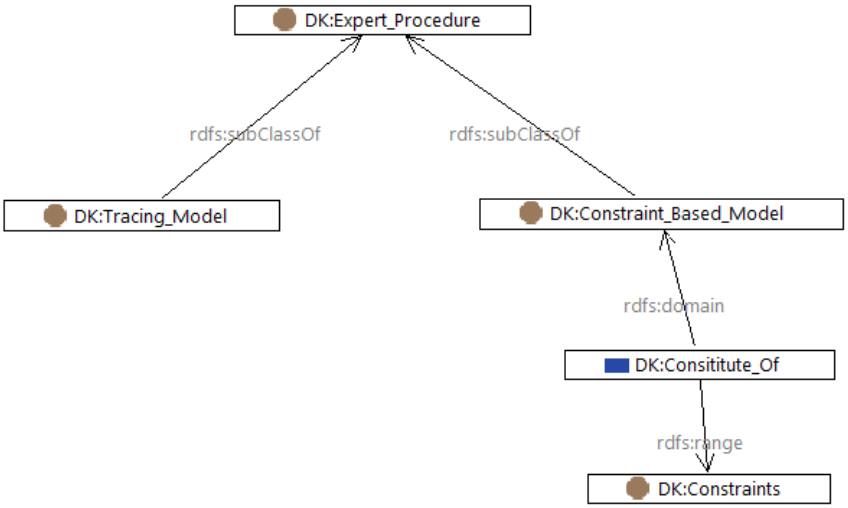

FIGURE 3.4 – Les sous classes de la classe "Expert procedure"

– Hiérarchie 3 : Le troisième niveau de notre ontologie est constitué de différentes classes et relations relatives à chaque classe de niveau supérieur (i.e. Instructional_Object , Technical_Object et Tracing Model). Nous allons définir dans ce qui suit ces différentes classes et les relations entre ces classes.

1- Les classes et les relations reliées à la classe " Instructif_Object " (voir figure 3.5) :

– L'objet instructif peut être constitué de cours, d'exemples , de rapports , de note de lecture et/ou d'exercices interactives. Donc, à ce niveau nous avons une hiérarchie de cinq sous-classes de la classe " Instructional_Object " qui est représenté par la relation "rdfs :SubClassOf ".
– L'objet instructif contient des propriétés "Has a Field", "Has Identifier", "Has a LearningContext" de type data type, qui le caractérise.

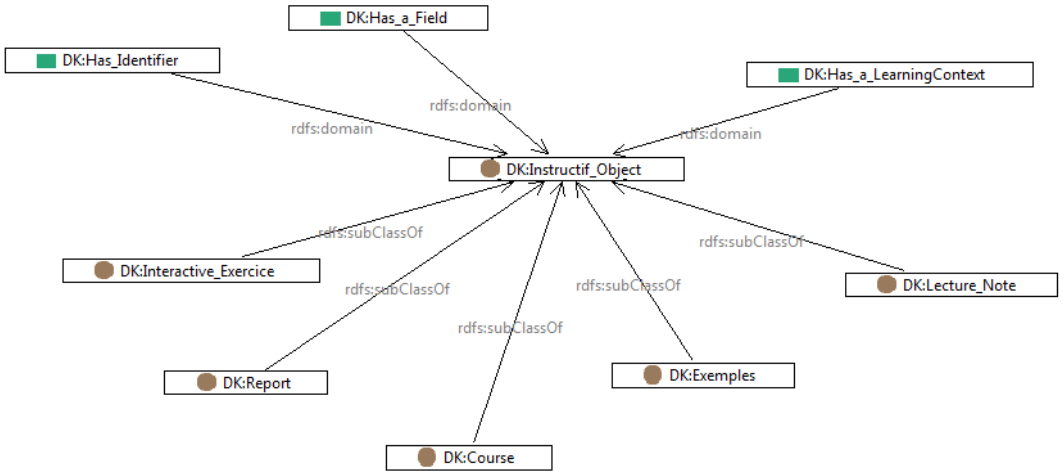

FIGURE 3.5 – Les sous classes de la classe " Instructif_Object"

2- Les classes et les relations reliées à la classe " Technical_Object " (voir figure 3.6) :

Le support technique peut être sous formes de texte, image , son et/ou multimédia. Donc, à ce niveau nous avons une hiérarchie de quatre sous-classes de la classe "Technical_Object" , et qui est représenté par la relation "rdfs :SubClassOf ".

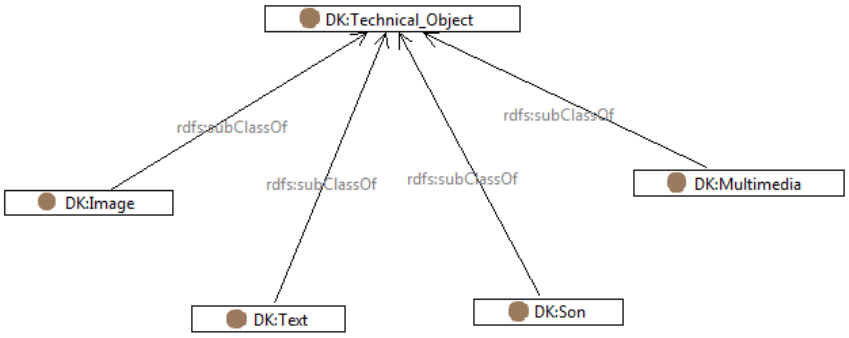

FIGURE 3.6 – Les sous classes de la classe " Technical_Object "

3- Les classes et les relations reliées à la classe " Tracing Model " (voir figure 3.7) :

Le modèle de traçage est composé de deux sous classes :" Expert rules " et " Buggy rules ". Nous les lions avec la relation " rdfs :SubClassOf ".

Dans le même niveau, les Buggys règles sont composés d'action et de condition. Nous définissons la relation "Include" pour lier les deux classes "Action" et "Condition" à la classe "Buggy rules".

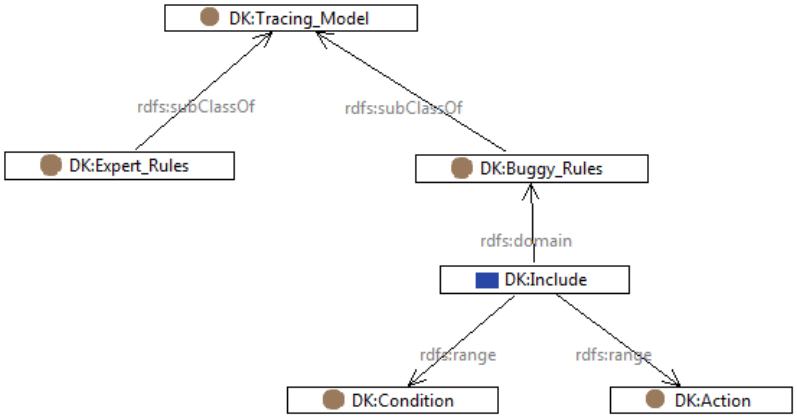

FIGURE 3.7 – Les sous classes de la classe "Tracing Model"

– Hiérarchie 4 : Le quatrième niveau de notre ontologie est constitué de différentes classes et relations relatives à chaque classe de niveau supérieur (i.e. Interactive_Exercice , Expert Rule et Constraints). Nous allons définir dans ce qui suit ces différentes classes et les relations entre ces classes.

1- Les classes et les relations reliées à la classe " Interactive_Exercice " (voir figure 3.8) :

Les exercices interactifs peuvent être sous forme de QCM, de catégorisation (faite glisser), d'ordonnancement (remplissage) et/ou question fermée. A ce niveau, nous avons une hiérarchie de cinq sous-classes de la classe " Interactive_Exercice ", qui est représentée par la relation " rdfs :SubClassOf ".

Dans ce même niveau, la classe QCM est composée d'un ensemble de questions. nous définissons la relation "Contain" pour relier la classe "QCM" avec la classe "Question". Cette dernière est connectée ou reliée à un concept et a une solution pour chaque question. Nous définissons donc, la relation "Is_Connected_To" pour relier la classe "Questions" à la classe "concept" et la relation "Has_a" pour relier la classe "Questions" à la classe "Solutions".

Enfin la classes "Questions" à deux propriétés de type datatype qui sont : "Unique Choice" et "Multiple Choice".

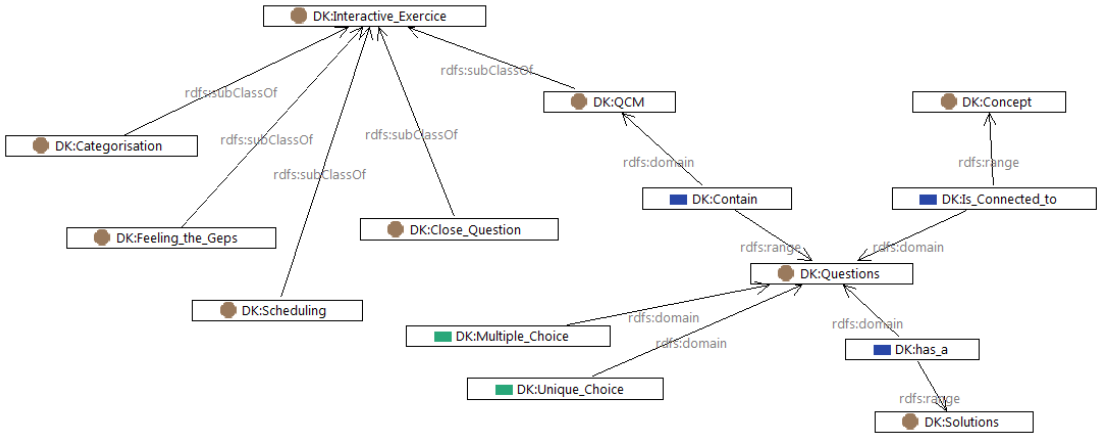

FIGURE 3.8 – Les sous classes de la classe "Interactive_Exercice"

2- Les classes et les relations reliées à la classe " Expert Rule " (voir figure 3.9) :

La règle de l'expert peut être sous forme de règle de planification et/ou la règle d'exécution. A ce niveau nous avons une hiérarchie de deux sous-classes de la classe " Expert Rule ", qui est représentée par la relation " rdfs :SubClassOf ".
La règle de planification est défini en but et sous-but de la règle. Nous définissons la relation "Divide_into" pour relier les deux classes "Goal" et "SubGoal" à la classe "Planification rule".

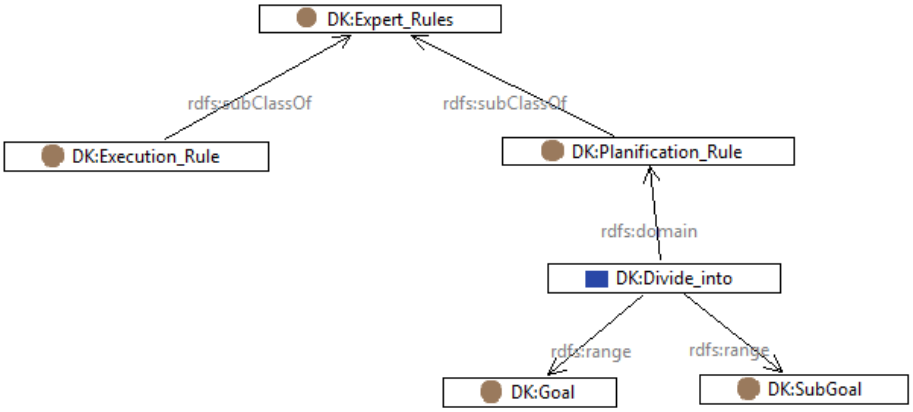

FIGURE 3.9 – Les sous classes de la classe "Expert rule"

3- Les classes et les relations reliées à la classe " Constraints " (voir figure 3.10) :

La contrainte a une condition pertinente et une condition satisfaisante. Nous définissons deux relations qui sont liées à la classe " Constraint". La première relation est "Has a Condition" qui relie la classe "Relevant Condition" à la classe "Constraint". La deuxième relation est "Has a Result" qui relie la classe "Satisfaction Condition" à la classe "Constraint".

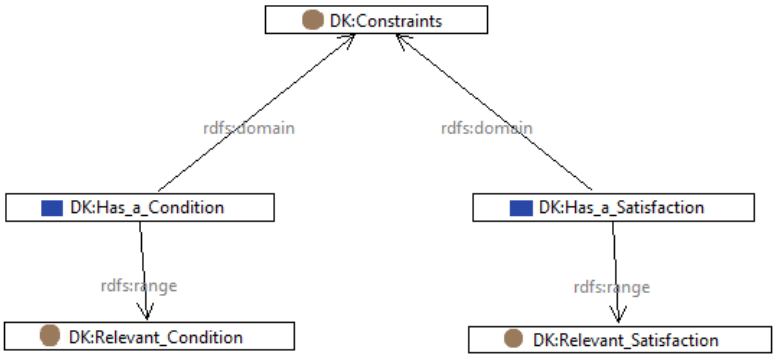

FIGURE 3.10 – Les sous classes de la classe "constraints"

Présentation de l'ontologie "Diag-k"

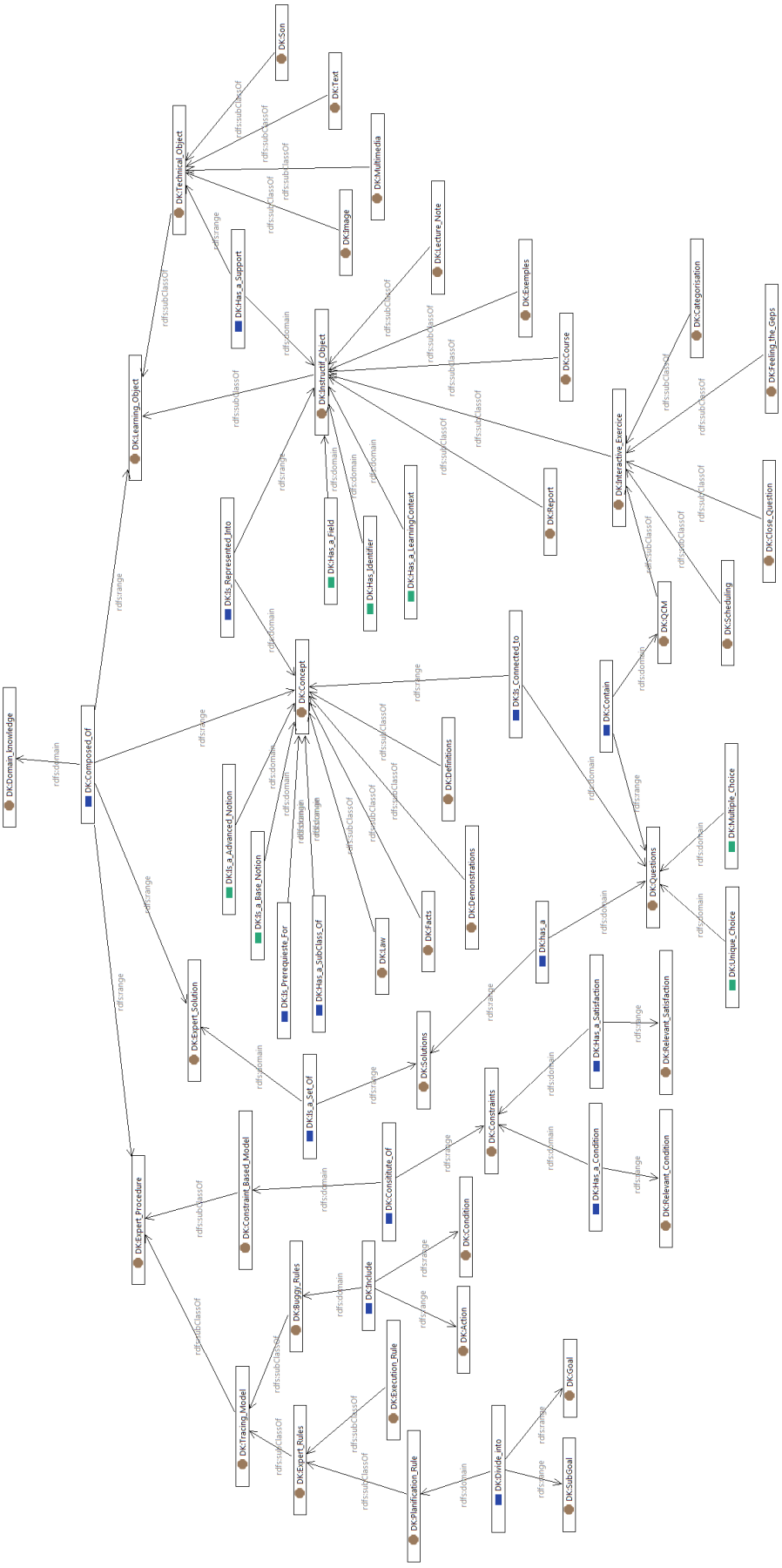

FIGURE 3.11 – La présentation de l'ontologie "Diag-k"

Dans cette section, nous présentons la totalité de notre ontologie présentée dans la figure 3.11. Cette ontologie est constituée de deux parties, la partie qui représente le modèle du domaine et la seconde partie qui représente le modèle de l'expert.

– Le modèle de domaine est constitué d'un ou plusieurs concepts qui peuvent être un fait, une définition, une loi et/ ou une démonstration, qui sont représentés sous forme d'objet instructif. Cet objet est un objet d'apprentissage (Learning_Object) et qui peut être le cours, exemple, rapport, note de lecture et/ ou un exercice interactif. Les exercices interactifs peuvent être sous forme de : QCM, question fermé, ordonnancement, remplissage (felling the geps) et/ou catégorisation. L'objet instructif a besoin d'un support pour le représenter, ce support est appelé Objet Technique. Ce dernier est lui aussi un Objet d'Apprentissage et qui peut être un texte, une image illustrative, un son et/ ou un multimédia.

Ce modèle peut contenir a priori, les connaissances qu'un système souhaite faire acquérir à son apprenant. Ces connaissances sont des connaissances déclaratives constituant le savoir théorique qui peut être des notions de bases ou avancées d'un concept pour un cours donné, des faits, des lois, des principes, etc. Les notions et concepts importants dans un cours accompagné de son évaluation seront présentés dans une plateforme.

– Le modèle de l'expert est composé de procédures et de solutions de l'expert. Cette dernière est un ensemble de solutions apportées par l'expert. La partie procédure de l'expert se décompose en deux parties : la partie définissant le modèle de traçage (MT) et la partie définissant le modèle basé sur les contraintes (CBM).

– Le modèle de traçage suit et guide l'apprenant dans chaque étape effectuée lors de sa résolution d'un problème et intervient dès que ce dernier (l'apprenant) s'éloigne d'une solution correcte. Le modèle de traçage est composé de règles de l'expert et de BUGGYs règles. Les règles de l'expert modélisent les étapes d'un apprenant intelligent et compétent qui peut suivre une résolution d'un problème.

Celles-ci incluent des règles d'exécutions et de planifications. Ces dernières sont décomposées en buts et sous buts d'un problème donné.

La règle de planification exprime la connaissance du domaine procédurale et la règle d'exécution exprime la connaissance du domaine déclaratif. Les BUGGYs règles reflètent les fausses perceptions de l'étudiant. Elles sont formulées sous forme d'actions et de conditions (IF condition THEN action).

– Le modèle basé sur les contraintes (CBM) est constitué de contraintes. Ces dernières sont structurées ou caractérisées avec une pertinence et une satisfaction de condition.

3.3.3 Formalisation

Dans cette phase, l'utilisation d'un outil est nécessaire pour représenter les différents éléments de notre ontologie. Plusieurs outils existent, notre choix a été porté sur l'utilisation de TopBraid pour les avantages que présente ce dernier. TopBraid est un environnement de modélisation visuel pour la création et gestion des modèles de domaine et des ontologies dans les normes (standard) du web sémantique RDF shema et OWL. Il est basé sur la plate-forme de Eclipse et utilise JENA comme API.

Nous avons choisi comme langage de modélisation RDF/RDFS s'appuyant sur la notation de type Turtule/N3 et un modèle de graphe orienté qui offre souplesse et interopérabilité pour décrire notre ontologie.

Le modèle de données RDF est proche des réseaux sémantiques. Un ensemble de déclarations RDF peut être vu comme un multi-graphe orienté, étiqueté, dont les sommets sont des ressources ou des littéraux et dont les arcs sont étiquetés par les propriétés des ressources. (pour plus de détails, voir annexe I)

3.3.4 Instanciation

La validation de notre approche est faite selon une instanciation. Pour montrer la nature générique de notre ontologie "Diag-K", nous avons décidé de prendre comme domaine "CISCO", représentée dans la figure 2.13 par l'instance "cisco :CISCO" qui est un rdf :type de la classe "Domain_Knowledge". Sa représentation en RDF est :
cisco :CISCO rdf :type DK :Domain_Knowledge.

Cette formation CISCO est faite en ligne via une plateforme spécifique nommée :
"www.cisco.netacad.net", représentée par l'instance "cisco :www.cisco.net"(voir figure 2.13) qui est un rdf :type de la classe "Learning_Object".
Sa représentation en RDF est : cisco : www.cisco.net rdf :type DK : Learning_Object.

CISCO contient plusieurs niveaux de formation, nous avons choisi de représenter le premier palier de cette formation nommé CCNA1, cette dernière est l'instance de la classe

"Instructif_Object" représentée par "cisco :CCNA1" qui est un rdf :type de cette classe. Sa représentation en RDF est : cisco :CCNA1 rdf :type DK : Instructif_Object.
Tous les supports techniques de la formation que ça soit images, textes,...vont être regrouper dans l'instance "cisco :Material_CCNA1" qui est un rdf :type de la classe "Technical_Object".
La classe "Instructif_Object" est représentée par l'instance "cisco :CCNA1". L'instance "CCNA1" a comme cours "cisco :Model_OSI" et a comme exercices interactifs "cisco :Exercice_M_OSI".

La classe "Interactive_Exercice" contient des QCM, qui sont représentés dans le graphe par l'instance "cisco :QCM_M_OSI", qui à leur tours contiennent un ensemble de questions représentés par les instances suivants "cisco :Q1" , "cisco :Q2" de la classe "Question".
Ces dernières sont liés à l'instance "cisco :QCM_M_OSI" par la relation "cisco :contain" et ils sont connectés respectivement aux instances de la classe "concepts" qui sont : "cisco :C_reseau", "cisco :C_liaison" par la relation "cisco :is_connected_to".

Chaque question a une solution. En effet, l'instance "cisco :Q1" est liée par la relation "cisco :has_a" à l'instance "cisco :S1". De même, l'instance "cisco :Q2" est liée par la relation "cisco :has_a" à l'instance "cisco :S2". Les instances "cisco :S1" et "cisco :S2" sont des instances de la classe "Solutions" et ils sont liés à l'instance "cisco :E_Solutions_Cisco" de la classe "expert_Soltion" par la relation "cisco :is_a_set_of".

Sans oublier, que l'instance "cisco :CISCO" de la classe "Domain_Knowledge" est composée des instances suivantes :
"cisco :www.cisco.net", "cisco :C_physique", "cisco :C_liaison", "cisco :C_reseau" et "cisco :E_Solutions_Cisco", de leur classes respectives : "Learning_Object", "Concept" et "Expert_solution".

Toute la formalisation de RDF est donnée en Annexe.

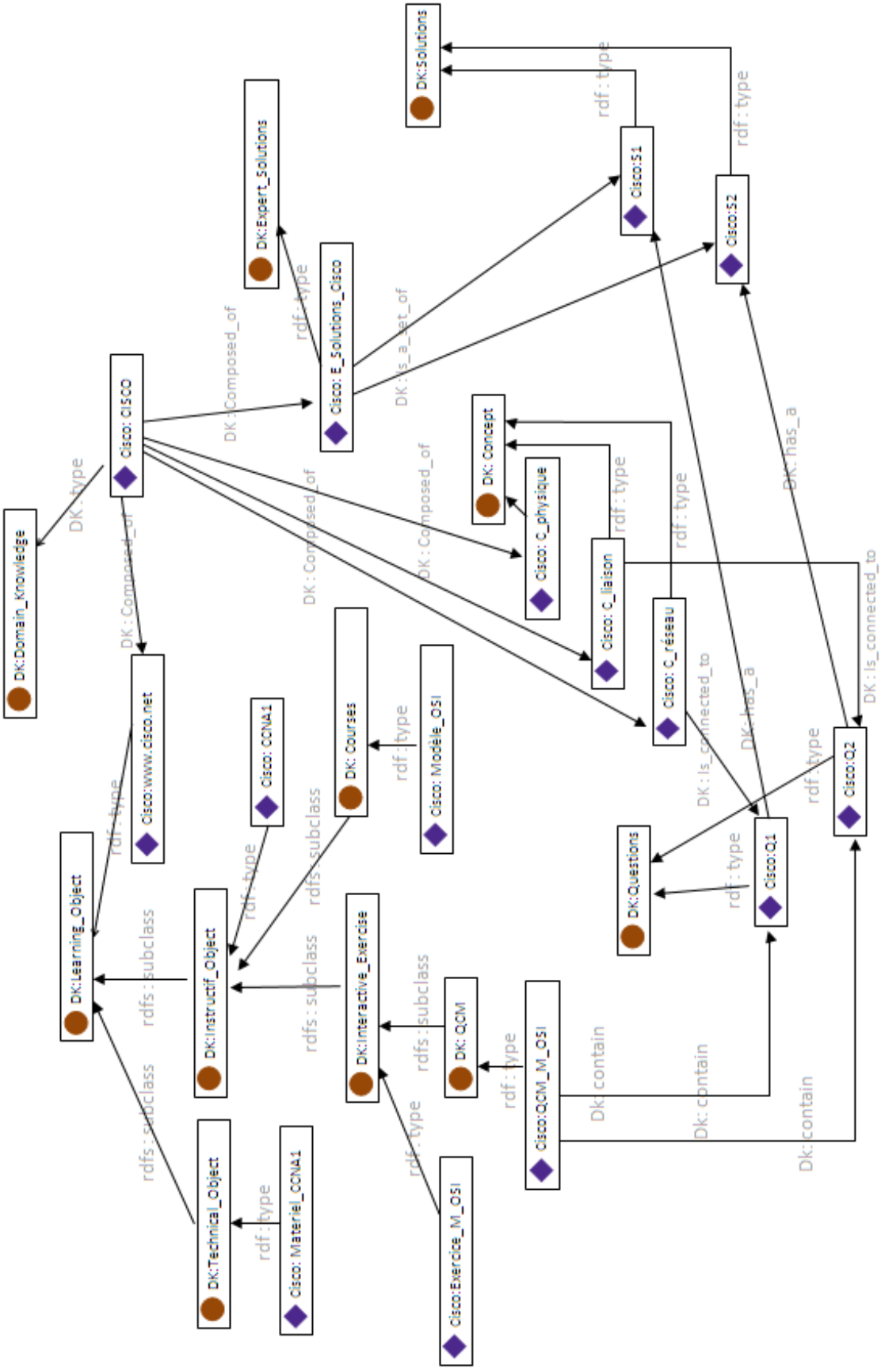

FIGURE 3.12 – La présentation du l'instanciation "CISCO"

3.4 Conclusion

Dans ce chapitre, nous avons vue en détail l'ontologie "Diag-K" avec la définition de ses classes et ses propriétés. L'outil TopBraid et le langage RDF/RDFS avec lequel on a travaillé et la méthode "Méthontologie" sur laquelle on s'est basé pour construire l'ontologie. Enfin un exemple de cours pour expliciter le modèle.

Chapitre 4

Conclusion et perspectives

Dans ce mémoire, nous avons tenté, après avoir pris en compte les différents systèmes relatifs à la modélisation des connaissances de l'apprenant, d'élaborer un méta-modèle que nous avons intitulé « Diag-K » et qui se veut une extension (ou servira d'extension) au modèle de diagnostic proposé par SETTOUTI. « Diag-K » signifie « Ontologie pour l'élaboration d'un modèle de connaissances de l'apprenant dans les EIAH ». Ce méta-modèle se définit en tant que modèle de connaissances et comporte deux volets : Le modèle du domaine et le modèle de l'expert.

L'originalité de ce travail est d'avoir proposé un meta-modèle qui prend en compte le caractère pluridisciplinaire des EIAH. En effet, le modèle que nous avons proposé réunit, en plus du modèle de domaine, toutes les approches et techniques utilisées au sein des EIAH contrairement aux systèmes existants qui se basent sur une seule approche et technique. De plus, ce modèle de connaissances a, lui-même, pour but de s'adapter aux différentes utilisations dans divers environnements d'apprentissage.

Par faute de temps, nous n'avons pas pu générer le profil de l'apprenant par contre nous avons essayer de penser à une règle de diagnostic qui sera rapidement mise en place. Cette règle comprend trois conditions : la comparaison entre la réponse de l'apprenant et celle de l'expert, le temps moyen adapté au groupe et nécessaire à la réponse, et enfin la vérification de la documentation (y compris l'aide) utilisée par l'apprenant au moment de sa réponse. La particularité de cette règle est de mesurer le taux de maitrise d'un concept pour un apprenant, par exemple : maitrise de concept à 70 pourcents.

La formulation de la règle de diagnostic sera mise en évidence dès que le modèle "Diag-K" sera fusiner avec le modèle de diagnostic proposé par SETTOUTI.

Nous prévoyons faire la réalisation d'une plateforme utilisable par le concepteur d'un EIAH afin de lui faciliter la mise en œuvre de l'analyse automatique des traces d'activité pour l'élaboration de profils d'apprenants. Cette plateforme s'appuira sur une le modèle "diag-k" et le modèle de diagnostic de SETTOUTI.

Bibliographie

[A.99] G-Pz A. ontological engineering : A state of the art . 2 :33–43, 1999.

[A01] Michard A. "xml langage et applications", 2001.

[Aux09] Ludovic Auxepaules. Analyse des diagrammes de lapprenant dansun eiah de la modsation orientbjet. 2009.

[Beg05] Mounir Beggas. Modelisation par un syst multi-agents d'un hypermedia educatif adaptatif dynamique. *Centre Universitaire d'Eloued - Magister en Informatique*, 156 :15–24–37, 2005.

[BM07] Peter Brusilovsky and Eva Millán. The adaptive web. chapter User models for adaptive hypermedia and adaptive educational systems, pages 3–53. 2007.

[BQCK06] C. Buche, R. Querrec, P. Chevaillier, and G. Kermarrec. Apports des systemes tutoriaux intelligents et de la realite virtuelle a lapprentissage de competences. *In Cognito – Cahiers Romans de Sciences Cognitives (CRSC)*, 2(2) :51–83, 2006.

[CHA00] P.-A. CHAMPIN. http ://www710.univ-lyon1.fr/ champin/rdf-tutorial/, 2000.

[CU09] Erica Melis Carsten Ullrich. Pedagogically founded courseware generation based on htn-planning. 2009.

[DBeRG] Eds. D. BRICKLEY et R. GUHA. Rdf vocabulary description language 1.0 : Rdf schema.

[dK01] Nora Parcus de Koch. *Software Engineering for Adaptive Hypermedia Systems. Reference Model, Modeling Techniques and Development Process.* PhD thesis, Fakultr Mathematik und Informatik der Ludwig-Maximilians-Universitnchen, 2001.

[DS92] P. Dillenbourg and J.A. Self. A framework for learner modelling. *Interactive Learning Environments*, 2 :111–137, 1992.

[GKeJC00] Eds. G. KLYNE et J. CARROLL. Resource description framework (rdf) : Concepts and abstract syntax., 2000.

[Gre94] Jim Greer. Granularity-based reasoning and belief revision in student models. student modelling : The key to individualized knowledge-based instruction,. *International Journal of Student Modelling*, 61 :10, 1994.

[Gru95] T.R. Gruber. toward principles for the design of ontologies used for knowledge sharing . *International Journal of Human Computer Studies.*, 1995.

[Har02] A. Hartley, D. & Mitrovic. Supporting learning by opening the student model. *In Cerri, S., Gouarderes, G. and Paraguacu, F. (eds.) Proc. 6 th International. Conference on Intelligent Tutoring Systems ITS 2002, Biarritz, France,*, pages pp. 453–462., 2002.

[HDN04] Nicola Henze, Peter Dolog, and Wolfgang Nejdl. Reasoning and ontologies for personalized e-learning in the semantic web. *Educational Technology & Society*, 7 :82–97, 2004.

[HUS08] Aarij Mahmood HUSSAAN. Using interaction traces to update domain knowledge and user prole in adaptive its. Master's thesis, Universitaude Bernard Lyon1, 2008.

[J.94] SELF J. The role of student models in learning environments. *Transactions of the institute of Electronics, Information and Communication Engineers, E77-D1.*, 182 :..., 1994.

[MC06] Ferraris C. Durand G Martel C., Vignollet L. Ldl : a language to model collaborative learning activities . *World Conference on Educational Multimedia, Hypermedia and Telecommunications, juin 2006.*, 2006.

[MFdCC08] Constantino Martins, Luíz Faria, Carlos Vaz de Carvalho, and Eurico Carrapatoso. User modeling in adaptive hypermedia educational systems. *Educational Technology & Society*, 11(1) :194–207, 2008.

[MFL99] ; A.G-Pz & A.Pazos-Sierra ; M. F. L. building a chemical ontology using methontology and the ontology design environment . *IEEE Intelligent Systems & their applications.* ., 1999.

[M.G96] M.Uschold & M.Grninger. ontologies : Principles, methods and applications . knowledge engineering review. 1996.

[Mit03] Antonija Mitrovic. An intelligent sql tutor on the web. *Int. J. Artif. Intell. Ed.*, 13 :173–197, 2003.

[Mue09] Martin Muehlenbrock. Continuous learner modeling in iclass. 2009.

[Mur99] Tom Murray. Authoring intelligent tutoring systems : An analysis of the state of the art. *International Journal of Artificial Intelligence in Education*, 10 :98–129, 1999.

[OLeRS99] Eds. O. LASSILA et R. SWICK. Resource description framework (rdf) model and syntax specification., 1999.

[OS96] LANGLEY P. OHLSSON S. Psychological evaluation of path hypotheses in cognitive diagnosis. learning issues for intelligent tutoring systems,. *Mandl H. & Lesgold A. Eds, Springer-Verlag,*, pages pp 42–62., 1996.

[Psy04] Val Psych"proposition dune mode dingenierie ontologique pour les eiah : applica- tion aux systs auteurs aspects thiques, mods et expmentations" ,. *proposition de recherche doctorale*, MAI 2004.

[Ren05] D Renaudie. *Méthodes d'apprentissage automatique pour la modsation de l'´l en alge. PhD thesis, Institut National Polytechnique de Grenoble, 2005.*

[S.94] OHLSSON S. Constraint-based student modeling. student modeling : The key to individualized knowlegde-based instruction. Springer-Verlag, pp 167-189., *203 :..., 1994.*

[SGML10] Lemya Settouti, Nathalie Guin, Alain Mille, and Vanda Luengo. A trace-based learner modelling framework for technology-enhanced learning systems. In ICALT*, pages 73–77, 2010.*

[SPMM09] Lotfi S. Settouti, Yannick Prie, Jean-Charles Marty, and Alain Mille. A trace-based system for technology-enhanced learning systems personalisation. In IEEE International Conference on Advanced Learning Technologies*, pages 93–97, 2009.*

[SS98] Raymund Sison and Masamichi Shimura. Student modeling and machine learning. IJAIED International Journal of Artificial Intelligence in Education*, 9 :128–158, 1998.*

[TCH02] Pierre TCHOUNIKINE. Quelques ments sur la conception et l'ingerie des eiah, 2002.

[Ull05] *C. Ullrich. The learning-resource-type is dead, long live the learning-resource-type !* Learning Objects and Learning Designs, *pages 7–15, 2005.*

[VK06] *David Rosenthal. Viswanathan Kodaganallur, Rob R. Weitz. A comparison of model-tracing and constraint-based intelligent tutoring para- digms. 2006.*

[Wen87] *E. Wenger.* Artificial Intelligence and Tutoring Systems, *volume 20. ACM, 1987.*

[wik] *http ://fr.wikipedia.org/wiki/environnements*$_i$*nformatiques*$_p$*our*$_l$

[Win90] R. Winkels. A new framework for describing and designing intelligent tutoring systems. In *Artificial Intelligence in Higher Education*, pages 230–243. Springer Verlag, Berlin, 1990.

[Zou05] Amal Zouaq. Systs tutoriels intelligents pour lorganisation : Gestion de la connaissance et de lintelligence collective. Technical report, Universit Montr Drtement dInformatique et de Recherche Optionnelle, 2005.

Annexe I : Codage de l'ontologie avec la technologie RDF/ RDF Schema

Introduction

Les "informations" jouent un rôle prépondérant, en ce sens qu'elles sont à la base de toute communication entre les êtres vivants. Celles-ci doivent être structurées pour être compréhensibles et réutilisables. L'accès et le partage de ces informations doivent se faire de la manière la plus simple possible.

La source d'inspiration est celle de la représentation de connaissance et notamment les langages de représentation de connaissance que sont les logiques de descriptions et les réseaux sémantiques (que nous considérerons ici sous leur aspect plus avancé du modèle des graphes conceptuels, un modèle graphique de représentation de connaissances de type Entité/Relation). Ces langages permettent d'exprimer la connaissance de nature ontologique (décrire des classes d'entités, les relier par spécialisation, décrire et typer leurs attributs) ou assertionnelle (les assertions affirment l'existence de relations entre des objets).

Dans cette section, on évoquera principalement RDF/S car il nous semble présenter des avantages déterminants pour la manipulation informatique en offrant une souplesse et une interopérabilité pour décrire les connaissances et leurs représentations ontologiques.

Langage de description RDF

– Description

RDF (Ressource Description Framework) est un langage utilisé pour représenter des informations sur des ressources et pour permettre un traitement automatique de celles-ci. Le principe est d'affecter des métadonnées à la description de ces ressources.

RDF [CHA00],[GKeJC00],[OLeRS99] est utilisé pour annoter des documents écrits dans des langages non structurés, ou comme une interface pour des documents écrits dans des langages ayant une sémantique équivalente (des bases de données, par exemple).

– Modèle et syntaxe RDF

La fondation de RDF est un modèle destiné à représenter des propriétés et des valeurs de propriétés données. Le modèle RDF s'appuie sur des principes bien établis provenant des différentes communautés de représentation des données. Vous pouvez imaginer les propriétés RDF comme les attributs de ressources et en ce sens elles correspondent aux paires traditionnelles attribut-valeur. Les propriétés RDF représentent également les relations entre les ressources. Un modèle RDF peut par conséquent ressembler à un diagramme Entité-Relation (ER). (Plus précisément, les schémas RDF — qui sont eux-mêmes des cas de modèles de données — sont des diagrammes ER.).Dans la terminologie de la conception orientée objet, les ressources correspondent aux objets et les propriétés correspondent aux variables.

Un document structuré en RDF est un ensemble de triplets de la forme :
<ressource,propriété,valeur> (<sujet,prédicat,objet>).
Les éléments de ces triplets peuvent être des URIs [1][DBeRG], des littéraux ou des variables.

1. Uniform Ressource Identifier, soit identifiant uniforme de ressource, protocole mis en place pour le web qui normalise la syntaxe de courtes chaînes de caractères désignant un nom ou une ressource, physique ou abstraite

Cet ensemble de triplets peut être représenté de façon naturelle par un graphe (plus précisément un multi-graphe orienté étiqueté), où les éléments apparaissant comme sujet ou objet, sont les sommets, et chaque triplet est représenté par un arc dont l'origine est son sujet et la destination son objet. ce document peut être représenté sous forme RDF/XML , Ntriplet, Turtul/N3, etc.

Le modèle RDF propose certains mots-clés réservés, qui permettent de donner une sémantique particulière à des ressources. Ainsi, on peut représenter des ensembles d'objets (rdf :bag), des listes (rdf :seq), des choix alternatifs (rdf :alt)...

RDF permet de spécifier un tel triplet selon deux syntaxes possibles : sérielle et abrégée (simplifiée). La première rend complètement explicite la structure créée, alors que la seconde est un tout petit peu plus rapide à écrire.

RDF autorise quelques simplifications ou variations syntaxiques. Dans notre cas, la simplification porte sur les déclarations de nouvelles ressources d'une classe définie ailleurs (dans notre schéma local). La syntaxe de base utilise la propriété rdf : type pour de telles déclarations. Le triplet nœud ->type ->classe peut être rendu implicite en utilisant le nom de la nouvelle ressource comme un nom d'élément. [A01]

Triplets :
<dispo1, rdf :type , file :///rdf et rdfs/ident-dispo RDFS.rdfDispositif>
<Dispositif, disp :titre$_{p}rojet$,"$Ingnieriedessystmes$" >

– RDF Schéma

RDF Schema est un langage extensible de représentation des connaissances. il appartient à la famille des langages du Web Sémantique publiés par le W3C. RDFS fournit des éléments de base pour la définition d'ontologies ou vocabulaires destinés à structurer des ressources RDF. C'est un des piliers du web sémantique puisqu'il permet de bâtir des concepts, définis par rapport à d'autres concepts, ayant la particularité d'être partagés à travers le web. [http ://fr.wikipedia.org/wiki/RDF$_{S}chema$]

Les déclarations RDF définissent des relations entre des objets (nœuds d'un graphe) qui appartiennent à un univers sémantique. A chacun de ces univers sémantiques correspond un domaine nominal, identifié par un préfixe particulier. Un tel domaine, pour lesquels sont définies des propriétés spécifiques et des catégories conceptuelles, est appelé un schéma.

Tout utilisateur a la possibilité de créer un nouveau schéma, de modifier, et notamment d'enrichir et d'affiner un schéma existant, de créer ce faisant un nouveau schéma personnalisé, et d'utiliser un schéma pour décrire les propriétés d'objets ayant une existence dans ce schéma.

RDFS (« RDF Schema ») offre les moyens de définir un modèle (ou bien encore un schéma) de méta données qui permet de :

• donner du sens aux propriétés associées à une ressource ;

• formuler des contraintes sur les valeurs associées à une propriété afin de lui assurer aussi une signification.

Les différentes classes et propriétés qui peuvent existés :

• rdfs :Class pour définir une nouvelle classe.

• rdfs :subClassOf, utilisé pour exprimer le fait qu'une classe hérite d'une autre classe (comme en programmation orientée objet), elle en possède donc toutes les caractéristiques (les propriétés).

Ex : On définit la classe Concept descente de la classe rdfs : class :

DK :Concept
a rdfs :Class ;
rdfs :subClassOf rdfs :Class .

On définit les propriétés avec le vocabulaire suivant :

• rdf :Property pour définir une nouvelle propriété.

• rdfs :domain pour associer la propriété à la classe.

• rdfs :range pour donner le type de la valeur que peut prendre la propriété.

Exemple : On définit la propriété "IsPrerequiesteFor" comme relation réflexive entre la classe "concept". d'où le "domain" est le départ et "range" est l'arrivée.

```
DK:IsPrerequiesteFor
      a       rdf:Property ;
      rdfs:domain  DK:Concept ;
      rdfs:range DK:Concept .
```

Annexe II

Code source des classes

```
baseURI: http://example.org/Domain\_Knowledge

@prefix DK:      <http://example.org/Domain_Knowledge#> .
@prefix owl:     <http://www.w3.org/2002/07/owl#> .
@prefix rdf:     <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs:    <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd:     <http://www.w3.org/2001/XMLSchema#> .

<http://example.org/Domain_Knowledge>
      rdf:type owl:Ontology ;
      owl:versionInfo "Created with TopBraid Composer"^^xsd:string .

DK:Domain_Knowledge
      rdf:type rdfs:Class ;
      rdfs:subClassOf rdfs:Class .

DK:Expert_Solution
      rdf:type rdfs:Class ;
      rdfs:subClassOf rdfs:Class .

DK:Categorisation
      rdf:type rdfs:Class ;
      rdfs:subClassOf DK:Interactive_Exercice .
```

```
DK:Facts
      rdf:type rdfs:Class ;
      rdfs:label "Facts xsd:string ;
      rdfs:subClassOf DK:Concept .

DK:Close_Question
      rdf:type rdfs:Class ;
      rdfs:subClassOf  DK:Interactive_Exercice .

DK:Concept
      rdf:type rdfs:Class ;
      rdfs:subClassOf rdfs:Class .

DK:Report
      rdf:type rdfs:Class ;
      rdfs:subClassOf DK:Instructif_Object .

DK:Multiple_Choice
      rdf:type owl:DatatypeProperty ;
      rdfs:domain  DK:Questions ;
      rdfs:range xsd:string .

DK:Constraints
      rdf:type rdfs:Class ;
      rdfs:subClassOf rdfs:Class .

DK:Tracing_Model
      rdf:type rdfs:Class ;
      rdfs:subClassOf DK:Expert_Procedure .

DK:Text
      rdf:type rdfs:Class ;
      rdfs:subClassOf DK:Technical_Object .
```

```
DK:Is_a_Set_Of
      rdf:type rdf:Property ;
      rdfs:domain DK:Expert_Solution ;
      rdfs:range DK:Solutions .

DK:Buggy_Rules
      rdf:type rdfs:Class ;
      rdfs:subClassOf DK:Tracing_Model .

DK:Execution_Rule
      rdf:type rdfs:Class ;
      rdfs:subClassOf DK:Expert_Rules .

DK:Law
      rdf:type rdfs:Class ;
      rdfs:label "Law" xsd:string ;
      rdfs:subClassOf  DK:Concept .

DK:Demonstrations
      rdf:type rdfs:Class ;
      rdfs:label "Demonstrations" xsd:string ;
      rdfs:subClassOf DK:Concept .

DK:Multimedia
      rdf:type rdfs:Class ;
      rdfs:subClassOf DK:Technical_Object .

DK:Consititute_Of
      rdf:type rdf:Property ;
      rdfs:domain DK:Constraint_Based_Model ;
      rdfs:range DK:Constraints .
```

```
DK:Suport
      rdf:type DK:Technical_Object ;
      rdfs:label "Suport" xsd:string ;
      rdfs:subClassOf rdfs:Resource .

DK:Planification_Rule
      rdf:type rdfs:Class ;
      rdfs:subClassOf DK:Expert_Rules .

DK:hasConceptName
      rdf:type rdfs:Class ;
      rdfs:range xsd:string ;
      rdfs:subClassOf owl:DatatypeProperty .

DK:Has_a_LearningContext
      rdf:type owl:DatatypeProperty ;
      rdfs:domain DK:Instructif_Object ;
      rdfs:range xsd:string .

DK:Expert_Rules
      rdf:type rdfs:Class ;
      rdfs:subClassOf DK:Tracing_Model .

DK:Is_Represented_Into
      rdf:type rdf:Property ;
      rdfs:domain DK:Concept ;
      rdfs:range DK:Instructif_Object .

DK:Condition
      rdf:type rdfs:Class ;
      rdfs:subClassOf rdfs:Class .
```

```
DK:Expert_Procedure
      rdf:type rdfs:Class ;
      rdfs:subClassOf rdfs:Class .

DK:Solutions
      rdf:type rdfs:Class ;
      rdfs:subClassOf rdfs:Class .

DK:Learning_Object
      rdf:type rdfs:Class ;
      rdfs:subClassOf rdfs:Class .

DK:Has_a_Satisfaction
      rdf:type rdf:Property ;
      rdfs:domain DK:Constraints ;
      rdfs:range DK:Relevant_Satisfaction .

DK:Interactive_Exercice
      rdf:type rdfs:Class ;
      rdfs:subClassOf DK:Instructif_Object .

DK:Technical_Object
      rdf:type rdfs:Class ;
      rdfs:subClassOf DK:Learning_Object .

DK:Divide_into
      rdf:type rdf:Property ;
      rdfs:domain DK:Planification_Rule ;
      rdfs:range DK:SubGoal , DK:Goal .
```

```
DK:Has_a_Field
      rdf:type owl:DatatypeProperty ;
      rdfs:domain DK:Instructif_Object ;
      rdfs:range xsd:string .

DK:Image
      rdf:type rdfs:Class ;
      rdfs:subClassOf DK:Technical_Object .

DK:SubGoal
      rdf:type rdfs:Class ;
      rdfs:subClassOf rdfs:Class .

DK:Goal
      rdf:type rdfs:Class ;
      rdfs:subClassOf rdfs:Class .

DK:Exemples
      rdf:type rdfs:Class ;
      rdfs:subClassOf DK:Instructif_Object .

DK:Include
      rdf:type rdf:Property ;
      rdfs:domain DK:Buggy_Rules ;
      rdfs:range DK:Action , DK:Condition .

DK:Son
      rdf:type rdfs:Class ;
      rdfs:subClassOf DK:Technical_Object .

DK:Video
      rdf:type DK:multimedia ;
      rdfs:label "Video" xsd:string .
```

```
DK:Composed_Of
      rdf:type rdf:Property ;
      rdfs:domain DK:Domain_Knowledge ;
      rdfs:range DK:Concept , DK:Expert_Procedure ,
      DK:Expert_Solution , DK:Learning_Object .

DK:Instructif_Object
      rdf:type rdfs:Class ;
      rdfs:subClassOf DK:Learning_Object .

DK:Has_Identifier
      rdf:type owl:DatatypeProperty ;
      rdfs:domain DK:Instructif_Object ;
      rdfs:range xsd:string .

DK:Has_a_Condition
      rdf:type rdf:Property ;
      rdfs:domain DK:Constraints ;
      rdfs:range DK:Relevant_Condition .

DK:Course
      rdf:type rdfs:Class ;
      rdfs:subClassOf DK:Instructif_Object .

DK:Action
      rdf:type rdfs:Class ;
      rdfs:subClassOf rdfs:Class .

DK:Constraint_Based_Model
      rdf:type rdfs:Class ;
      rdfs:subClassOf DK:Expert_Procedure .
```

```
DK:Questions
      rdf:type rdfs:Class ;
      rdfs:label "Questions" xsd:string ;
      rdfs:subClassOf rdfs:Class .

DK:Relevant_Satisfaction
      rdf:type rdfs:Class ;
      rdfs:subClassOf rdfs:Class .

DK:Feeling_the_Geps
      rdf:type rdfs:Class ;
      rdfs:subClassOf DK:Interactive_Exercice .

DK:Has_a_Support
      rdf:type rdf:Property ;
      rdfs:domain DK:Instructif_Object ;
      rdfs:range DK:Suport , DK:Technical_Object .

DK:Is_a_Advanced_Notion
      rdf:type owl:DatatypeProperty ;
      rdfs:domain DK:Concept ;
      rdfs:range xsd:string .

DK:Relevant_Condition
      rdf:type rdfs:Class ;
      rdfs:subClassOf rdfs:Class .

DK:Has_a_SubClass_Of
      rdf:type rdf:Property ;
      rdfs:domain DK:Concept ;
      rdfs:range DK:Concept .
```

```
DK:Lecture_Note
      rdf:type rdfs:Class ;
      rdfs:subClassOf DK:Instructif_Object .

DK:Scheduling
      rdf:type rdfs:Class ;
      rdfs:subClassOf DK:Interactive_Exercice .

DK:Is_a_Base_Notion
      rdf:type owl:DatatypeProperty ;
      rdfs:domain DK:Concept ;
      rdfs:range xsd:string .

DK:Unique_Choice
      rdf:type owl:DatatypeProperty ;
      rdfs:domain DK:Questions ;
      rdfs:range xsd:string .

DK:Definitions
      rdf:type rdfs:Class ;
      rdfs:label "Definitions" xsd:string ;
      rdfs:subClassOf DK:Concept .
```

Code source des instances

```
baseURI: http://example.org/Domain_Knowledge
@prefix cisco:      <http://example.org/Domain_Knowledge#>
@prefix DK:      <http://example.org/Domain_Knowledge#> .
@prefix owl:     <http://www.w3.org/2002/07/owl#> .
@prefix rdf:     <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs:    <http://www.w3.org/2000/01/rdf-schema#> .
```

```
@prefix xsd:     <http://www.w3.org/2001/XMLSchema#> .

Cisco:CCNA1
      rdf:type DK:Concept ;
      rdfs:comment "CCNA1 est une instance de
      la classe \"concept\""^^xsd:string ;
      rdfs:label "CCNA1"^^xsd:string ;
      rdfs:subClassOf rdfs:Resource .

Cisco:Is_Prerequieste_For
      rdf:type rdf:Property ;
      rdfs:domain DK:Concept , DK:CCNA1 ;
      rdfs:range DK:Concept , DK:CCNA1 .

Cisco:Is_Connected_to
      rdf:type rdf:Property ;
      rdfs:domain <http://example.org/Domain_Knowledge#Question.M-OSI>
       , DK:CCNA1 , DK:Questions ;
      rdfs:label "Is Connected Of"^^xsd:string ;
      rdfs:range DK:CCNA1 , DK:Concept .

Cisco:Definitions_des_7couches
      rdf:type DK:Definitions ;
      rdfs:comment "instance de la classe \"Définitions\""^^xsd:string ;
      rdfs:label "Definitions des 7couches"^^xsd:string ;
      rdfs:subClassOf DK:CCNA1 .

<http://example.org/Domain_Knowledge#www.cisco.netacad.net>
      rdf:type DK:Instructif_Object ;
      rdfs:comment"instance de la classe\"Instructif_Object\""^^xsd:string ;
      rdfs:label "www.cisco.netacad.net"^^xsd:string ;
      rdfs:subClassOf rdfs:Resource .
```

```
Cisco:Is_Represented_Into
      rdf:type rdf:Property ;
      rdfs:domain DK:CCNA1 , DK:Concept ;
      rdfs:range <http://example.org/Domain_Knowledge#www.cisco.netacad.net>,
      DK:Instructif_Object .

Cisco:Has_a_Support
      rdf:type rdf:Property ;
      rdfs:domain <http://example.org/Domain_Knowledge#www.cisco.netacad.net>,
           DK:Instructif_Object ;
      rdfs:range DK:Support , DK:Technical_Object .

Cisco:Cours_1
      rdf:type DK:Course ;
      rdfs:comment "instance de la classe \"Course\" qui est une sous classe de
      \"instructif object\" et qui a comme instance \"Modèle_OSI\""^^xsd:string ;
      rdfs:label "Modèle OSI"^^xsd:string ;
      rdfs:subClassOf <http://example.org/Domain_Knowledge#www.cisco.netacad.net> .

Cisco:Modèle_OSI
      rdf:type DK:Cours_1 ;
      rdfs:comment "instance de la ressource Cours_1"^^xsd:string ;
      rdfs:label "Modèle OSI"^^xsd:string .


Cisco:Evaluation_modeleOSI
      rdf:type DK:Interactive_Exercice ;
      rdfs:comment "instance de la classe \"Interactive_Exercice\" plus exactement
     les QCM permettant de faire l'evaluation"^^xsd:string ;
      rdfs:label "evaluation1"^^xsd:string ;
      rdfs:subClassOf <http://example.org/Domain_Knowledge#www.cisco.netacad.net> .
```

```
Cisco:QCM_Modele-OSI
      rdf:type DK:QCM ;
      rdfs:label "QCM Modedle-OSI"^^xsd:string ;
      rdfs:subClassOf DK:Evaluation_modeleOSI .


Cisco:Contain
      rdf:type rdf:Property ;
      rdfs:domain DK:QCM ,
     <http://example.org/Domain_Knowledge#Question.M-OSI> ;
      rdfs:label "Contain"^^xsd:string ;
      rdfs:range DK:Questions , DK:QCM_Modele-OSI .

<http://example.org/Domain_Knowledge#Question.M-OSI>
      rdf:type DK:Questions ;
      rdfs:comment "instance de la classe \"Questions\""^^xsd:string ;
      rdfs:label "Question.M-OSI"^^xsd:string ;
      rdfs:subClassOf rdfs:Resource .

Cisco:has_a
      rdf:type rdf:Property ;
      rdfs:domain DK:Questions ,
             <http://example.org/Domain_Knowledge#Question.M-OSI> ;
      rdfs:range DK:Solutions ,
              <http://example.org/Domain_Knowledge#Solutions.M-OSI> .

<http://example.org/Domain_Knowledge#Solutions.M-OSI>
      rdf:type DK:Solutions ;
      rdfs:comment "instance de la classe \"Solutions\""^^xsd:string ;
      rdfs:label "Solutions.M-OSI"^^xsd:string ;
      rdfs:subClassOf rdfs:Resource .
```

```
<http://example.org/Domain_Knowledge#C.Physique>
      rdf:type DK:CCNA1 ;
      rdfs:comment "instance du concpet de CCNA1"^^xsd:string ;
      rdfs:label "Couche Physique"^^xsd:string .

Cisco:Def_CPhysique
      rdf:type DK:Definitions_des_7couches ;
      rdfs:comment "instance de Definition_des_7_couches qui elle même
      est une instance de la classe definition"^^xsd:string ;
      rdfs:label "Def Couche physique"^^xsd:string .

<http://example.org/Domain_Knowledge#Q2.CPhysique>
      rdf:type <http://example.org/Domain_Knowledge#Question.M-OSI> ;
      rdfs:comment "instance de la classe Question.M-OSI qui appartient
      à la classe OCM.Model-OSI"^^xsd:string ;
      rdfs:label "Q2.Couche physique"^^xsd:string .

<http://example.org/Domain_Knowledge#S.CPhysique>
      rdf:type <http://example.org/Domain_Knowledge#Solutions.M-OSI> ;
      rdfs:comment "instance de la classe Solutions.M-OSI"^^xsd:string ;
      rdfs:label "S.CPhysique"^^xsd:string .

<http://example.org/Domain_Knowledge#C.Liaison>
      rdf:type DK:CCNA1 ;
      rdfs:comment "instance du concept de CCNA1"^^xsd:string ;
      rdfs:label "Couche liaison donnée"^^xsd:string .

Cisco:Def_CLiaison
      rdf:type DK:Definitions_des_7couches ;
      rdfs:comment "instance de Definition_des_7_couches qui elle même
      est une instance de la classe definition"^^xsd:string ;
      rdfs:label "Def Couche liaison"^^xsd:string .
```

```
<http://example.org/Domain_Knowledge#Q3.CLiaison>
      rdf:type <http://example.org/Domain_Knowledge#Question.M-OSI> ;
      rdfs:comment "instance de la classe Question.M-OSI qui appartient
      à la classe OCM.Model-OSI"^^xsd:string ;
      rdfs:label "Q3.Couche liaison"^^xsd:string .

<http://example.org/Domain_Knowledge#S.CLiaison>
      rdf:type <http://example.org/Domain_Knowledge#Solutions.M-OSI> ;
      rdfs:comment "instance de la classe Solutions.M-OSI"^^xsd:string ;
      rdfs:label "S.CLiaison"^^xsd:string .

<http://example.org/Domain_Knowledge#C.Reseau>
      rdf:type DK:CCNA1 ;
      rdfs:comment "instance du concpet de CCNA1"^^xsd:string ;
      rdfs:label "Couche reseau"^^xsd:string .

 Cisco:Def_CReseau
      rdf:type DK:Definitions_des_7couches ;
      rdfs:comment "instance de Definition_des_7_couches qui elle même
      est une instance de la classe definition"^^xsd:string ;
      rdfs:label "Def Couche reseau"^^xsd:string .

<http://example.org/Domain_Knowledge#Q1.CReseau>
      rdf:type <http://example.org/Domain_Knowledge#Question.M-OSI> ;
      rdfs:comment "instance de la classe Question.M-OSI qui appartient
      à la classe QCM.Model-OSI" ^^xsd:string ;
      rdfs:label "Q1.Couche reseau"^^xsd:string .

<http://example.org/Domain_Knowledge#S.CReseau>
      rdf:type <http://example.org/Domain_Knowledge#Solutions.M-OSI> ;
      rdfs:comment "instance de la classe Solutions.M-OSI"^^xsd:string ;
      rdfs:label "S.CReseau"^^xsd:string .
```

```
Cisco:Support
      rdf:type DK:Technical_Object ;
      rdfs:comment "instance de la classe\"Technical_Object\""^^xsd:string ;
      rdfs:label "Suport"^^xsd:string ;
      rdfs:subClassOf rdfs:Resource .

Cisco:Plateforme
      rdf:type DK:Support ;
      rdfs:comment "instance du support"^^xsd:string ;
      rdfs:label "Plateforme"^^xsd:string .

Cisco:texte
      rdf:type DK:Text ;
      rdfs:comment "instance de la classe \"Text\""^^xsd:string ;
      rdfs:label "texte"^^xsd:string ;
      rdfs:subClassOf DK:Support .

Cisco:textuel
      rdf:type DK:texte ;
      rdfs:comment "instance du texte"^^xsd:string ;
      rdfs:label "textuel"^^xsd:string .

Cisco:multimedia
      rdf:type DK:Multimedia ;
      rdfs:comment "instance de la classe \"Multimedia\""^^xsd:string ;
      rdfs:label "multimedia"^^xsd:string ;
      rdfs:subClassOf DK:Support .

Cisco:Video
      rdf:type DK:multimedia ;
      rdfs:comment "instance du multimedia"^^xsd:string ;
      rdfs:label "Video"^^xsd:string .
```

```
Cisco:image
      rdf:type DK:Image ;
      rdfs:comment "instance de la classe \"Image\""^^xsd:string ;
      rdfs:label "image"^^xsd:string ;
      rdfs:subClassOf DK:Support .
```