UNIVERSITE BRETAGNE LOIRE

Université du Maine

Thèse de Doctorat

Esteban LOISEAU

Mémoire présenté en vue de l'obtention du
grade de Docteur de l'Université du Maine
sous le sceau de l'Université Bretagne Loire

École doctorale : *Sciences et Technologies de l'Information et Mathématiques (STIM)*

Discipline : *Informatique (Section CNU 27)*
Unité de recherche : *Laboratoire d'Informatique de l'Université du Maine (LIUM)*

Soutenue le *28 novembre 2016*

Composition et transformation de modèles pour la spécification de langages de scénarisation pédagogique graphiques centrés sur la plate-forme de formation Moodle

JURY

Rapporteurs :	**Thierry NODENOT**, Professeur, Université de Pau et des pays de l'Adour **Xavier LEPALLEC**, Maître de conférences HDR, Université de Lille 1
Examinateurs :	**Jean-Marc LABAT,** Professeur, UPMC (Paris 6) **Christophe CHOQUET**, Professeur, Université du Maine
Directeur de Thèse :	**Sébastien IKSAL,** Maître de conférences HDR, Université du Maine
Co-encadrant :	**Pierre LAFORCADE**, Maître de conférences, Université du Maine

REMERCIEMENTS

Je tiens à remercier **Xavier Le Pallec** et **Thierry Nodenot**, d'avoir donné de leur temps et d'avoir accepté le rôle de rapporteur sur cette thèse.

Je remercie également **Jean-Marc Labat** et **Christophe Choquet**, d'avoir bien voulu être examinateurs et participer à ce jury.

Mes remerciements à **Sébastien Iksal**, pour la direction de cette thèse, pour ses conseils, sa patience et son soutien.

Merci à **Pierre Laforcade**, pour son soutien autant professionnel que personnel, sans lui je n'aurais pas fait cette thèse et je ne saurais faire la somme de ses contributions à ce travail.

Merci à mes collègues doctorants : **Quentin**, **Inès**, **Guillaume**, **Aous**, **Vincent**, **Zeyneb** et tous les autres, anciens et nouveaux, pour leur soutien et leur amitié, pour avoir fait de ces journées au bureau un vrai plaisir.

Merci à tous les membres de **l'équipe IEIAH** et plus largement du **LIUM** pour leur accueil.

Je remercie également tous mes collègues de **l'IUT de Laval**, en particulier des **départements Informatique** et **MMI**.

Merci à **Yann Walkowiak** pour son soutien et son amitié, pour m'avoir confié son bureau ces quelques années.

Merci à tous mes ami(e)s : **Julie**, **Axel**, **Nathan**, **Fabien** et les autres.

Je remercie ma **famille** pour ses encouragements, en particulier **mes parents**, **mon frère** et **ma sœur** qui m'ont soutenu dans ma vie d'éternel étudiant.

Je dédie cette thèse à la mémoire de mon grand-père, Xixi, qui m'a toujours encouragé à apprendre.

TABLE DES MATIÈRES

1

INTRODUCTION

Dans ce chapitre, nous introduisons le contexte de recherche de cette thèse qui croise le domaine des EIAH et de l'Ingénierie Dirigée par les Modèles. Nous précisons en particulier le périmètre de nos travaux dans le contexte du projet de recherche GraphiT, et établissons une première problématisation.

1.1 CONTEXTE DE LA THÈSE

1.1.1 *EIAH et scénarisation pédagogique*

Cette thèse s'inscrit dans le domaine de recherche des Environnements Informatique pour l'Apprentissage Humain. Le terme EIAH englobe tout dispositif informatique visant à favoriser l'apprentissage, quelles que soient les modalités : à distance, en présentiel ou en situation mixte [Tcho2]. Ce domaine de recherche implique de nombreuses disciplines (sciences de l'éducation, sciences de l'information et de la communication, informatique, didactique etc.) sur des problématiques telles que la conception de dispositifs, l'analyse des usages et des traces d'utilisation, le tutorat ou encore la modélisation de l'apprenant. Les plate-formes de formation en ligne, telles que les LMS (*Learning Management System*) sont un exemple d'EIAH employés couramment par de nombreux organismes de formation.

L'emploi d'un EIAH en tant qu'outil dans une situation d'apprentissage implique une conception à plus grande échelle. Cette conception pédagogique peut être vue comme un processus comprenant plusieurs phases : expression des besoins, analyse et conception, implémentation, déploiement et tests [Vano3]. Dans le cadre de cette thèse nous nous intéressons en particulier au scénario pédagogique, qui est un produit de la phase d'analyse et de conception. Un scénario pédagogique consiste en une description, formelle ou non, du déroulement d'une séquence d'enseignement [Tcho9]. Il contient des informations sur les objectifs pédagogiques, le séquencement des activités, les ressources, les rôles attribués etc. Le scénario pédagogique peut être construit à différents niveaux de détails et peut intégrer tout ou partie des éléments précédents [LP04]. Il peut décrire une situation à gros grains, en terme de structure uniquement (séquencement des sessions d'apprentissages par exemple) ou jusqu'à la plus petite unité de modélisation en intégrant tous les détails liés à l'opérationnalisation du scénario (ressources matériels, humaines, temporalité...). Un scénario pédagogique n'est pas nécessairement formel, mais il est décrit le plus souvent à l'aide d'un langage de scénarisation pédagogique. Certains langages visent uniquement à une exploitation informatique des scénarios, d'autres proposent une notation visuelle facilitant l'activité de scénarisation par les enseignants.

1.1.2 *Contexte local*

Ces travaux s'inscrivent dans l'axe de recherche « Conception, opérationnalisation et adaptation de situations pédagogiques » de l'équipe IEIAH (Ingénierie des EIAH) du Laboratoire d'Informatique de l'Université du Maine. Ils s'appuient sur l'expérience de l'équipe sur des thématiques liées à la scénarisation pédagogique :

- conception de scénarios en Pédagogie par Projet Collectif [Abd09]
- conception collective de scénarios pédagogiques par les enseignants [ElK08]
- conception continue de scénarios pédagogiques ouverts, adaptables par les enseignants [Our12]
- modélisation et opérationnalisation de scénarios pédagogiques sur des plate-formes de formation à distance [Abe13a]

Cette dernière thèse a permis d'initier un travail de recherche sur la scénarisation pédagogique pour les plate-formes de formation en ligne, de type LMS (*Learning Management System*). Ces travaux ont montré qu'il était possible d'identifier et de formaliser le métier de conception pédagogique d'une plate-forme de formation, et ont proposé un processus menant à cette identification. Sur cette base ont été proposés un langage de scénarisation pédagogique graphique outillé ainsi qu'un module d'import de scénario sur Moodle. Ce langage de scénarisation et son éditeur permettent aux enseignants-concepteurs de spécifier des scénarios graphiquement, la limite est qu'il s'appuie directement et uniquement sur le métier de conception de la plate-forme. Autrement dit, son expressivité est limitée par la sémantique que propose le LMS. Ce travail s'est prolongé au travers du projet GraphiT, présenté ci-après.

1.1.3 *Le projet GraphiT*

Le projet GraphiT [1] est un projet de recherche Jeune Chercheur financé par l'ANR et porté par Pierre Laforcade. Il a été initié en février 2012 pour une durée de 44 mois [Laf16].

Les plate-formes de formation telles que Moodle sont aujourd'hui largement déployées dans les institutions académiques pour des usages couvrant plus que la formation à distance : utilisations en présentiel, *blended learning* [TDP11]... Malgré tout, les enseignants concepteurs conservent une vision fonctionnelle orientée « outils » de ces plate-formes. Bien souvent, ils découvrent les usages potentiels des fonctionnalités de la plate-forme par eux-même en détournant les utilisations prévues, en étudiant les différents paramétrages techniques proposés, etc. Le manque d'outils-auteurs spécifiques à leur plate-forme ou bien d'outils de conception génériques leur garantissant que les scénarios conçus seront opérationnalisables automatiquement sur

1. *Graphical VIDLS 4 Teachers*

leur plateforme, les amène à concevoir en même temps qu'ils configurent les activités pédagogiques.

« S'ils souhaitent mettre en place des activités pédagogiques complexes, les enseignants doivent développer des compétences de haut-niveau quant à l'utilisation du LMS : comment et quand utiliser et combiner les différentes fonctionnalités de la plate-forme afin de supporter l'objectif pédagogique fixé ? Ces compétences peuvent être acquises au cours de formations professionnelles, qui se concentrent davantage sur les aspects techniques liés aux fonctionnalités de la plate-forme plutôt qu'à la conception de scénarios pédagogiques cohérents et adaptés à cette plate-forme. »[Loi+15]

Ces constats sont issus d'une enquête menée par le projet, comprenant un questionnaire ainsi que des entretiens individuels réalisés par une ingénieure pédagogique auprès d'enseignants-concepteurs [Pod14]. Sur cette base, le projet GraphiT propose une approche centrée plate-forme, à l'opposée des solutions génériques traditionnelles, pour fournir aux enseignants des outils dédiés à la conception de scénarios pédagogiques opérationnalisables spécifiques à la plate-forme qu'ils utilisent. Au delà de l'objectif de compatibilité avec la plate-forme, un des enjeux du projet est de proposer une solution en accord avec les pratiques et les besoins des enseignants-concepteurs et donc, tout en maintenant cette compatibilité, de s'éloigner de la sémantique et du fonctionnement propre à la plate-forme (et potentiellement différent des pratiques des enseignants).

Un objectif secondaire du projet consiste à explorer en quoi le cadre théorique et outillé de l'Ingénierie Dirigée par les Modèles (IDM) et du *Domain Specific Modeling* (DSM) peut être utile et pertinent dans notre approche. Le projet est de type *design recherche* : il cherche, à travers la conception et le développement d'une solution informatique spécifique, à confirmer/infirmer l'approche originale centrée sur le métier de conception des plate-formes mais également à étudier si le cadre théorique et technique de l'IDM/DSM se prête à soutenir de tels travaux en conception pédagogique [Laf15].

Notre thèse s'inscrit directement dans le projet GraphiT et partage ses constats et objectifs. Nous nous appuyons également sur d'autres résultats du projet : un outil (API) d'import de scénario sur la plate-forme Moodle ainsi qu'un méta-modèle capturant le métier de conception pédagogique de cette même plate-forme. Ces deux contributions ont fait l'objet de publications scientifiques [EOL15] [Abe13b] et ne seront que partiellement décrites dans ce manuscrit, car elles n'entrent pas dans le périmètre de cette thèse.

1.1.4 *Ingénierie Dirigée par les Modèles et Domaine Specific Modeling*

L'Ingénierie Dirigée par les Modèles (ou *Model Driven Engineering*) [JCV12] est une approche d'ingénierie logicielle, qui met l'emphase

sur la manipulation de modèles. Ces modèles sont des abstractions d'objets liés à un domaine métier spécifique. L'IDM favorise la production de modèles qui soient « exécutables », et pas uniquement descriptifs, et met en avant les pratiques d'ingénierie générative qui visent à générer tout ou partie de l'application cible. Ceci, afin de simplifier le développement d'applications complexes. Bien que les modèles considérés puissent être abstraits à des niveaux variés, et donc potentiellement éloignés du système modélisé, l'IDM vise à manipuler des modèles formels. Cette caractéristique repose sur la conformité des modèles à un méta-modèle, qui définit les concepts métiers et leurs relations de façon à restreindre (et formaliser) le contenu des modèles. Il existe également un concept de méta-méta-modèle qui régit la structure des méta-modèles mais qui est lui récursif, au sens où il est conforme à lui-même (il s'auto-décrit). L'IDM est également un champ de recherche en informatique qui aborde des thèmes tels que :

— la transformation de modèles (modèle à modèle ou modèle à texte) ;
— la composition de modèles ;
— le tissage de modèles.

Des travaux s'intéressent également à la modélisation des aspects dynamiques [MFJ05] du domaine, le méta-modèle étant statique, ainsi qu'aux moyens d'exploiter les modèles produits : animation/débogage [Bou+15], simulation etc.

Dans un processus classique exploitant l'IDM, le méta-modèle est défini en accord avec des experts du domaine et les modèles sont manipulés par les utilisateurs finaux, le plus souvent via des outils dédiés. Dans le cadre du projet GraphiT et de cette thèse, la projection de l'approche IDM sur le domaine cible, la scénarisation pédagogique, est considéré de la manière suivante :

— le scénario pédagogique (SP) est un modèle de la session d'apprentissage telle que prévue par l'enseignant ;
— le langage de scénarisation pédagogique est décrit par un méta-modèle (LSP) ;
— le métier de conception pédagogique de la plate-forme est un méta-modèle (MPL) ;
— un cours sur la plate-forme de formation peut-être abstrait par un modèle (CPL) ;

Ainsi, un modèle de scénario SP est conforme au méta-modèle LSP et un modèle de cours CPL est conforme au méta-modèle MPL. La figure 1 illustre le positionnement et les relations entre ces différents éléments. Comme indiqué en section précédente, le méta-modèle MPL ainsi que le module logiciel l'exploitant (API d'import) sont des contributions du projet GraphiT.

L'objectif du méta-modèle LSP est de capturer les pratiques pédagogiques des enseignants-concepteurs, il apparaît donc que LSP et MPL sont hétérogènes. Le méta-modèle MPL est issue d'un proces-

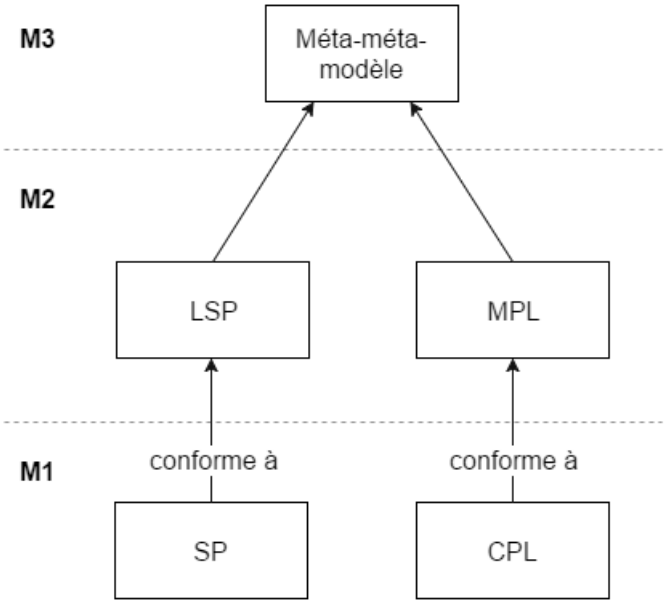

FIGURE 1 – Positionnement des différents modèles et méta-modèles

sus d'identification se basant sur une analyse technique et fonctionnelle de la plate-forme qui elle même a été conçue avec une intention d'usage pédagogique : il est donc possible que LSP et MPL partagent des similitudes. Les pratiques considérées pour la conception de LSP sont celles d'enseignants-concepteurs utilisateurs de plate-formes, qui n'en sont pas les concepteurs et qui souvent sont tributaires d'un choix de celle-ci opéré au niveau de l'institution, ce qui limite la similarité entre LSP et MPL. L'approche DSM que nous adoptons ne se limite pas à la définition de méta-modèle, il existe de nombreux outillages DSM qui permettent la spécification des aspects graphiques et interactifs des langages de modélisation et la génération de code pour le développement d'un éditeur de modèle adapté. Ces outillages permettent directement de produire des éditeurs à destination des utilisateurs finaux, dans notre cas les enseignants-concepteurs. Les modèles produits par ces éditeurs seront conformes aux méta-modèles définis par construction.

Le double objectif visé par le projet GraphiT, compatibilité avec la plate-forme et réponse aux besoins et pratiques des enseignants, trouve ici une problématique au sens IDM. Comment faire le lien entre un modèle de scénario pédagogique, potentiellement indépendant de la plate-forme, et un modèle de cours conforme au méta-modèle du métier de conception pédagogique de la plate-forme ?

1.2 PROBLÉMATIQUE ET PÉRIMÈTRE DE LA THÈSE

1.2.1 *Périmètre de la thèse*

Bien qu'ayant participé à ces travaux dans le cadre du projet GraphiT, cette thèse n'intègre pas directement les problématiques liées à l'opérationnalisation des scénarios pédagogiques. De même, à l'échelle du projet plusieurs plate-formes de formation sont considérées mais dans le cadre de cette thèse, les contributions seront spécifiques au LMS Moodle. L'approche générale et théorique doit rester en revanche généralisable, et sera discutée en conclusion de ce manuscrit.

Les problématiques de capture et de formalisation des pratiques des enseignants sont traitées également à l'échelle du projet [PCC14], mais n'entrent pas dans le périmètre de cette thèse. L'explicitation et la formalisation du métier de conception de la plate-forme de formation sont également des problématiques étudiées par le projet GraphiT, et bien qu'utiles aux travaux présentés ici, ils n'entrent pas directement dans le périmètre de cette thèse. La figure 2 présente les travaux de recherches initiés par le projet GraphiT et situe en particulier nos travaux.

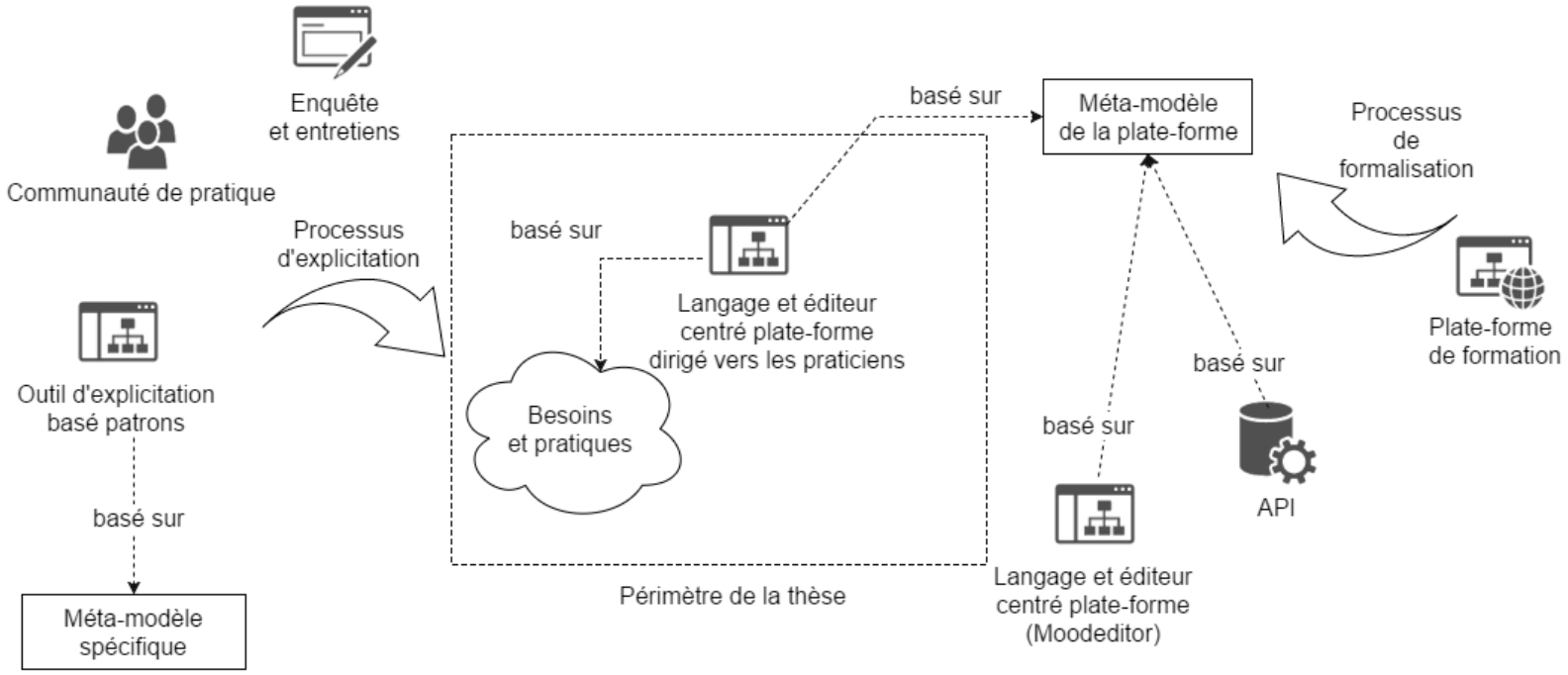

FIGURE 2 – Synthèse des travaux initiés dans le projet GraphiT

1.2.2 *Problématique et objectifs*

Dans le cadre de l'approche centrée plate-forme préconisée par le projet GraphiT, il est nécessaire de prendre la plate-forme comme point de départ. Dès lors, si l'objectif est de permettre aux enseignants de concevoir des scénario proches de leurs pratiques, une première problématisation est de savoir : *comment s'abstraire du métier de conception de la plate-forme ?* Quels concepts métiers issus des pratiques de conception doivent être proposés aux enseignants et comment les associer aux fonctionnalités de la plate-forme nécessaires à leurs implémentations en vue de l'opérationnalisation ?

On peut diviser cette problématique en sous-problèmes :

— Comment passer d'un modèle de scénario à un modèle de cours sans pertes sémantiques ?
— Comment étendre le métier de conception de la plate-forme avec des concepts issus des pratiques ?
— Comment identifier les concepts à ajouter et leurs relations avec ceux de la plate-forme ?

D'un point de vue Ingénierie Dirigée par les Modèles : quelles techniques issues de l'IDM utiliser pour

— transformer les modèles de scénario ;
— étendre le méta-modèle de la plate-forme ;
— relier les concepts des deux méta-modèles ;

Notre proposition ne devra pas seulement être théorique mais devra proposer des outils permettant d'éditer des modèles de scénario et d'exécuter les opérations nécessaires à l'obtention d'un modèle exécutable (opérationnalisable). En accord avec les principes de l'IDM nous devrons limiter les coûts de développement de ces outils en favorisant l'ingénierie générative.

1.3 ORGANISATION DU MANUSCRIT

Le chapitre suivant est dédié à l'état de l'art, à la fois sur les langages de scénarisation pédagogique graphiques (ou VIDL) et sur l'Ingénierie Dirigée par les Modèles et le *Domain Specific Modeling*. Ce chapitre permet de présenter et d'analyser l'existant ainsi que de positionner notre approche.

Le chapitre 3 amène un synthèse de cet état de l'art et présente la méthodologie de recherche adoptée pour ces travaux. Il présente également un aperçu des contributions décrites dans ce manuscrit.

Le quatrième chapitre traite du recueil des pratiques, besoins et exigences des enseignants-concepteurs. Il présente l'enquête et les entretiens que nous avons menés ainsi que leurs principaux résultats.

Le chapitre 5 décrit notre proposition quant à la formalisation et l'exécution des correspondances entre éléments pédagogiques abstraits et outils/fonctionnalités de la plate-forme.

Dans un sixième chapitre, sur la base des besoins et exigences recueillis, nous établissons des spécifications concernant notre proposition de langage de scénarisation pédagogique graphique. Ces spécifications traitent à la fois de l'aspect sémantique (méta-modèle) et de l'aspect graphique (notation visuelle à base de diagrammes).

Notre septième chapitre se concentre sur la conception d'un prototype d'éditeur, outillant les spécifications précédents. Nous y décrivons entre autres les fonctionnalités de celui-ci.

Dans un huitième et dernier chapitre, nous concluons l'ensemble de ce manuscrit en présentant un ensemble de perspectives, à court, moyen et long termes.

2

ETAT DE L'ART

Dans ce chapitre, nous présentons un état de l'art portant à la fois sur les langages de scénarisation pédagogique graphiques et sur l'Ingénierie Dirigée par les Modèles. L'objectif est de définir ces deux domaines et de nous positionner quant aux travaux existants.

Dans une première section, nous définissons le concept de *Visual Instructional Design Language* et étudions comment il est possible de classifier de tels langages. Nous présentons, dans cette même section, plusieurs solutions issues de la littérature qui partagent notre objectif d'opérationnalisation sur un LMS. Une seconde section permet de faire le lien entre VIDL et IDM en étudiant comment un langage de scénarisation pédagogique peut être considéré comme un langage de modélisation, d'après le *Domain Specific Modeling*. La section suivante présente les conclusions d'un travail antérieur à cette thèse qui nous a conduit à étudier l'Ingénierie Dirigée par les Modèles et à adopter l'approche par extension et composition que nous proposons. Nous présentons ensuite les principes de composition et de transformation de modèle et leurs applications dans le cadre de notre proposition.

2.1 VISUAL INSTRUCTIONAL DESIGN LANGUAGE

Notre problématique portant sur les langages de scénarisation pédagogique graphiques (ou VIDL) il est tout d'abord nécessaire de les définir.

2.1.1 *Définition*

Les langages de modélisation pédagogique visuels, ou *Visual Instructional Design Language*, sont une évolution des langages de modélisation pédagogiques traditionnels (*Educational Modeling Language*). Ces derniers proposaient un modèle d'information (décrivant les concepts, leurs relations et leurs sémantiques) ainsi qu'une correspondance (*binding*) vers une expression formelle de ce modèle dans un langage compréhensible par l'ordinateur (généralement réalisée en XML). Un EML décrit donc formellement des ressources éducatives et/ou des scénarios pédagogiques. [Raw+02]. Les EMLs visaient principalement la ré-utilisation des scénarios pédagogiques ainsi que l'intéropérabilité en proposant un format indépendant des plate-formes.

On peut donc considérer les VIDLs comme des EMLs particuliers moins enclins à proposer une formalisation strictement formelle ni une exécutabilité des modèles produits, mais au contraire à exploiter les valeurs ajoutées de l'utilisation d'une notation visuelle [Moo09].

2.1.2 *Intérêt*

D'après [FDo6], les VIDL peuvent être utilisés, entre autres, pour :

— amener une meilleure communication entre les experts en usant d'une terminologie commune,
— favoriser la compréhension du scénario, via la représentation visuelle vis-à-vis d'une notation textuelle,
— ré-utiliser un scénario ayant prouvé son efficacité,
— partager des bonnes-pratiques de scénarisation,
— former des concepteurs débutants,
— documenter une approche pédagogique dans un format standard,
— réduire l'écart entre conception et implémentation.

Néanmoins, afin de profiter de tous ces avantages, les enseignants-concepteurs doivent développer une certaine expertise dans l'usage de ce ou ces langages. Le temps d'appropriation du langage dépend directement de l'utilisabilité de celui-ci.

2.1.3 *Classifications*

Devant la multitude de VIDL existants, certains auteurs ont proposé des catégorisations ou taxonomies afin de pouvoir caractériser et comparer ces langages. La classification la plus utilisée est celle proposée par [Bot+06].

2.1.3.1 *Classification de Botturi*

Dans cette publication, Botturi présente une classification s'appuyant sur cinq critères :

STRATIFICATION (valeurs : *flat*, *layered*). Fait référence à la compartimentation des éléments du langage. Si tous les éléments du langage sont regroupés dans une seule représentation, le VIDL est considéré comme *flat* (à plat). S'il est possible de manipuler et visualiser des éléments indépendamment des autres, il est considéré comme *layered* (en couches). Botturi cite UML comme exemple de langage en couches.

FORMALIZATION (valeurs : *formal, semi-formal, informal*). Un langage formel définit strictement les concepts manipulables ainsi que les relations possibles entre eux. Un langage informel, au contraire laisse plus de liberté au concepteur dans le choix des éléments à intégrer au scénario. L'opérationnalisation des scénarios modélisés dépend fortement de cette caractéristique.

ELABORATION (valeurs : *conceptual, specification, implementation*). Cette caractéristique est liée à la granularité des scénarios modélisés. Au niveau conceptuel, le concepteur composera des éléments de haut-niveau, offrant une vue générale du cours modélisé. A l'opposé, au niveau implémentation, un maximum de détails est ajouté au scénario, favorisant ainsi l'opérationnalisation.

	Niveau de stratification	Niveau de formalisation	Niveau d'élaboration	Nombre de perspectives	Notation
E^2ML	plat	semi-formel	conceptuel	plusieurs	visuel
PCeL	plusieurs couches	semi-formel	conceptuel	une	visuel
AUTC	plat	informel	spécification	plusieurs	visuel
IMS-LD	plusieurs couches	formel	spécification	une	textuel
PoEML	plusieurs couches	formel	implémentation	plusieurs	visuel
UML	plusieurs couches	formel	conceptuel / spécification	plusieurs	visuel

FIGURE 3 – Exemples de langages de modélisation positionnés dans la classification de Botturi (traduit de [Bot+06])

PERSPECTIVE (valeurs : *single, multiple*). Un langage multi-perspectives propose plusieurs représentations pour un même élément, selon le point de vue adopté pour la représentation du scénario.

NOTATION SYSTEM (valeurs : *textual, visual*). Spécifie le mode de visualisation principal : textuel ou visuel.

La figure 3 est une version traduite du tableau présenté dans [Bot+06], qui présente le positionnement de plusieurs langages de modélisation pédagogiques selon les critères présentés ci-dessus. Deux langages particuliers sont à noter : UML est un langage de modélisation généraliste, bien que souvent utilisé pour la modélisation d'un système informatique, il est standardisé par l'OMG [Gro16d], IMS-LD est le standard *de facto* pour la modélisation pédagogique [Con16].

En complément, Botturi propose deux dimensions, sur lesquelles positionner les VIDL :

COMMUNICATION (échelle : *Reflective - Communicative*) Qualifie l'usage visé par le langage : réflexif, vise un usage individuel en supportant la réflexion personnelle lors des premières phases de conception par exemple ; communicatif, le langage vise la communication entre les différents participants à l'activité de conception, ou aux autres intervenants n'appartenant pas à la même discipline.

CREATIVITY (échelle : *Finalist - Generative*) Qualifie l'approche de conception visée : un langage génératif vise une conception itérative, par étapes de raffinement du scénario ; un langage proposant de formaliser un scénario dans sa version complète et définitive est considéré comme *finalist*.

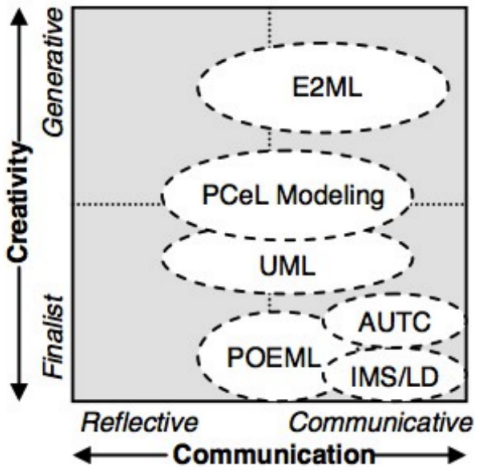

FIGURE 4 – Positionnement de VIDLs sur les deux dimensions [Bot+06]

La figure 4 illustre le positionnement de certains langages de modélisation pédagogique selon ces deux axes. IMS-LD est en particulier positionné comme un langage permettant l'échange de scénarios pédagogiques une fois la conception terminée, ce qui correspond bien à son statut de format standard de persistence des scénarios.

2.1.3.2 *Autres classifications*

Nodenot [Nod07] propose une classification selon les informations représentées dans le scénario. La figure 5 illustre le positionnement de plusieurs langages de scénarisation pédagogique selon les critères suivants :

1. Les rôles et responsabilités que peuvent prendre les acteurs de la session d'apprentissage.
2. Les modalités pour chaque unité d'enseignement : planification, synchronisation selon les acteurs, différenciation, etc.
3. Les connaissances mobilisées dans la réalisation des activités.
4. Les buts d'apprentissages : objectifs, pré-requis etc.
5. La structure du module d'apprentissage : dépendances entre les activités.
6. Les collaborations entre acteurs : responsabilités, synchronisation etc.
7. La diffusion du contenu du scénario : comment sera concrètement implémenté et déployé le scénario.

La figure 5 positionne certains VIDL selon la classification proposée par Nodenot.

Dans [Laf10], Laforcade propose de positionner les VIDL dans trois catégories, selon le métier visé par le langage. Le métier dépend effectivement de la communauté d'utilisateurs et des enjeux pris en

	E²ML	CPM	coUML	MOT+	PoEML	ASK-LDT	COLLAGE
Rôles et responsabilités		+	+		+	+	+
Modalités d'apprentissage	+		+			+	
Domaine, connaissances	+	+	+	+			
Objectifs et buts d'apprentissage	+	+		+			
Structure du cours	+			+	+	+	
Collaborations entre acteurs	+	+	+			+	+
Liens avec l'infrastructure		+	+	+	+	+	

FIGURE 5 – Positionnement de VIDL selon la classification de Nodenot [Nod07]

compte lors de la conception du langage. La figure 6 illustre le positionnement de plusieurs VIDL selon ces trois catégories.

LANGAGE CENTRÉ PRATICIENS : son vocabulaire métier est celui d'une équipe pluridisciplinaire de conception, des communautés d'enseignants-concepteurs. Un langage de cette catégorie doit pouvoir capturer leur métier, leur vocabulaire commun, qu'il soit lié à des approches pédagogiques particulières, à des plate-formes de formation connues, etc. L'objectif d'un tel langage est de faciliter la définition même du scénario pédagogique, c'est-à-dire de guider l'activité de conception.

LANGAGE ABSTRAIT : son vocabulaire cherche à être indépendant de celui de la plate-forme de formation (ou de tout autre dispositif) dans le but de faciliter sa réutilisation ; il est donc possible que le vocabulaire d'un langage abstrait cherche aussi à s'abstraire d'approches pédagogiques trop spécifiques. L'objectif principal est l'interopérabilité des dispositifs et l'abstraction des langages spécifiques à un domaine.

LANGAGE CENTRÉ PLATE-FORME : son vocabulaire est conforme à celui d'une plate-forme bien précise. L'objectif est alors de servir de guide pour la configuration du dispositif (manuellement ou automatiquement selon le format du scénario).

Pour chaque catégorie, deux types de notations sont possibles :

— visuelle, ayant une notation textuelle ou graphique dans le but de rendre le scénario interprétable par les membres de la communauté visée (enseignants, concepteurs, etc.)
— forme informatique standard dans le but de rendre le scénario spécifié interprétable sans ambiguïté par la machine et/ou qu'il soit facilement stocké, recherché et échangé.

Dans le second cas, la notation doit être basée sur un format standard, non propriétaire (généralement XML), permettant d'expliciter les concepts/relations embarqués.

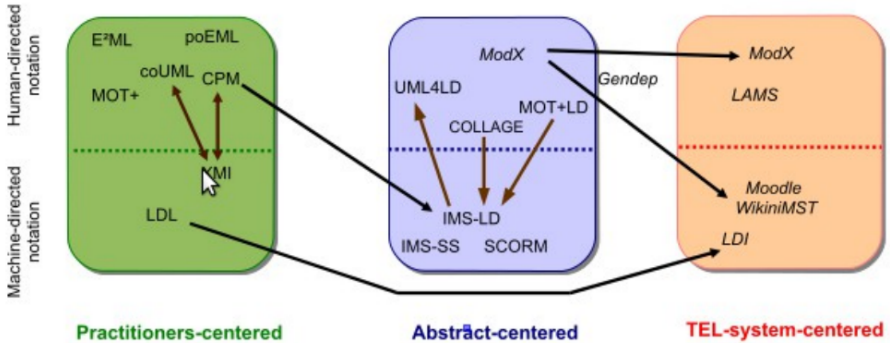

FIGURE 6 – Positionnement de VIDL selon les catégories proposées par Laforcade [Laf10]

Deux types de transformations sont également décrites et illustrées par les flèches sur la figure 6 :

— la transformation de l'un de ces scénarios vers une autre catégorie (transformation extra-domaine) permet à la fois d'atteindre les objectifs de l'autre catégorie mais aussi d'accéder à des communautés différentes (dans un souci de partage et de réutilisation)
— transformation intra-domaine afin de représenter visuellement un scénario décrit dans un format informatique standard tout aussi bien que l'inverse, c'est-à-dire retranscrire le scénario visuel dans un format informatique standard indépendant de toute représentation visuelle.

2.1.4 *Positionnement du VIDL ciblé*

Si l'on se réfère à la classification [Bot+06], les VIDLs qui nous intéressent doivent avoir les caractéristiques suivantes :

STRATIFICATION en couches. Il est important que le langage de conception offre la possibilité aux concepteurs de ne pas devoir spécifier tout leur scénario sur un seul et unique diagramme. L'utilisation de plusieurs couches comme différents diagrammes inter-reliés permet de séparer les points de vue sur la conception (plus pédagogique ou plus fonctionnel/technique).

FORMALISATION : nous visons les langages de scénarisation pédagogique opérationnalisables. L'opérationnalisation souhaitée est automatique donc relève du traitement informatique, il est alors important que le langage soit formel.

ÉLABORATION : comme pour le critère précédent, l'opérationnalisation nécessite des informations de niveau implémentation. Néanmoins, du point de vue de l'utilisateur (l'enseignant-concepteur), la modélisation doit pouvoir se faire au niveau spécification, afin de favoriser la réflexion pédagogique aux considéra-

tions techniques. Pour ce critère, la valeur hybride spécification / implémentation est préférée.

PERSPECTIVE : les objectifs fixés n'imposent pas de valeur spécifique pour ce critère. Un langage multi-perspective peut permettre une meilleure compréhension du scénario, mais apporte une complexité supplémentaire au langage.

NOTATION : visuelle, principalement, afin d'être exploitée par les enseignants-concepteurs. L'outillage doit permettre d'exploiter une notation textuelle qui sera utilisée pour la persistance de la sémantique des modèles et leur exploitation informatique.

COMMUNICATION : l'aspect collaboratif n'est pas considéré pour le moment. Il s'agirait plutôt de langages réflexifs.

CRÉATIVITÉ : l'approche IDM proposée par le projet GraphiT, permet d'exploiter des technologies qui sont destinées à formaliser et fixer les modèles plutôt qu'à créer/raffiner des solutions : la visée est donc finaliste.

Selon la classification de [Nodo7], les VIDLs visés devraient aborder les dimensions **rôles et responsabilités**, **modalités d'apprentissage**, **buts d'apprentissage**, **structure d'un module d'apprentissage**, en relation étroite et cohérente avec ce que la plate-forme de formation est capable d'exprimer. En revanche nous écartons la représentation des connaissances car celle-ci n'est généralement pas fonctionnellement couverte par les plate-formes de formation. Enfin, la représentation de la diffusion devrait également être prise en compte en permettant aux concepteurs de spécifier ou de visualiser tout ou partie de leur conception en terme de configuration de la plate-forme.

Selon la classification basée sur la centration du métier de conception, les VIDLs que nous visons sont centrés sur le métier de conception de la plate-forme (*TEL-system-centered*) mais ne sont pas limités à l'expressivité de celui-ci : l'objectif est de s'abstraire de ce métier orienté infrastructure concrète de réalisation pour se diriger vers un métier de conception plus abstrait, mettant davantage en avant des dimensions pédagogiques. Les VIDLs seront donc centrés plate-formes mais dirigés vers les besoins de conception des enseignants les utilisant. Les notations sont visuelles (*Human-directed notation*), mais doivent proposer un formalisme de persistance afin que les scénarios soient traduits dans un format interprétable par la machine (*Machine-directed notation*).

2.1.5 *État de l'art des VIDL centrés plate-forme*

Dans cette section, nous présentons différents travaux de l'état de l'art, tous ces travaux proposent un langage outillé de scénarisation pédagogique supportant l'opérationnalisation, en particulier vers la plate-forme Moodle.

2.1.5.1 *Cadmos*

CADMOS [KRB12] est un éditeur de scénario pédagogique graphique qui s'adresse à des enseignants-concepteurs novices en termes de compétences et connaissances informatiques. Le scénario est décrit en plusieurs phases : d'abord de façon statique en spécifiant les activités des apprenants et des enseignants ainsi que les ressources et services utilisés (voir figure 7), puis en orchestrant les différentes activités à l'aide de *swim lanes* représentant le déroulement du cours du point de vue des différents acteurs (voir figure 8). Cette seconde

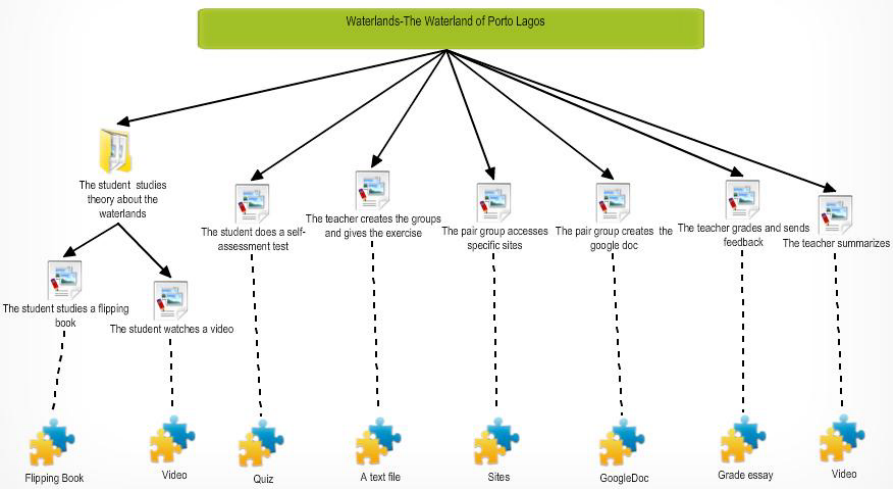

FIGURE 7 – Exemple de diagramme du *conceptual model* d'un scénario dans CADMOS [Bol+12]

phase permet de séquencer les activités et de préciser les règles et pré-conditions associées à celles-ci. Le processus global de conception supporté par l'outil est considéré comme incrémental. CADMOS s'appuie sur son propre méta-modèle [KRB12], les scénarios conçus sont sauvés dans un format propriétaire (.cdm). Un export est possible vers IMS-LD [1] (niveaux A et B) : un fichier manifest peut être généré et exécuté par des éditeurs ou players compatibles IMS-LD. L'outil offre également la possibilité d'importer un fichier manifest IMS-LD existant, et de le traduire dans son propre format. Ainsi, un enseignant peut ré-utiliser un scénario IMS-LD existant. CADMOS propose également un export compatible avec la plate-forme Moodle. Un script dédié gère la traduction des concepts de CADMOS vers Moodle et génère concrètement un fichier backup (.mbz) prêt à être déployé sur la plate-forme. Pour importer le scénario sous forme de cours il faut donc que l'enseignant utilise la fonctionnalité de Moodle

1. IMS-LD[Con16] est le standard *de facto* pour la modélisation de scénarios pédagogiques. Il utilise la métaphore théâtrale et exploite le format XML pour la persistence des modèles. Il ne propose pas de notation visuelle par défaut, mais de nombreux travaux ont porté sur la conception d'outils de modélisation graphique pour IMS-LD [Der15][Gri+09].

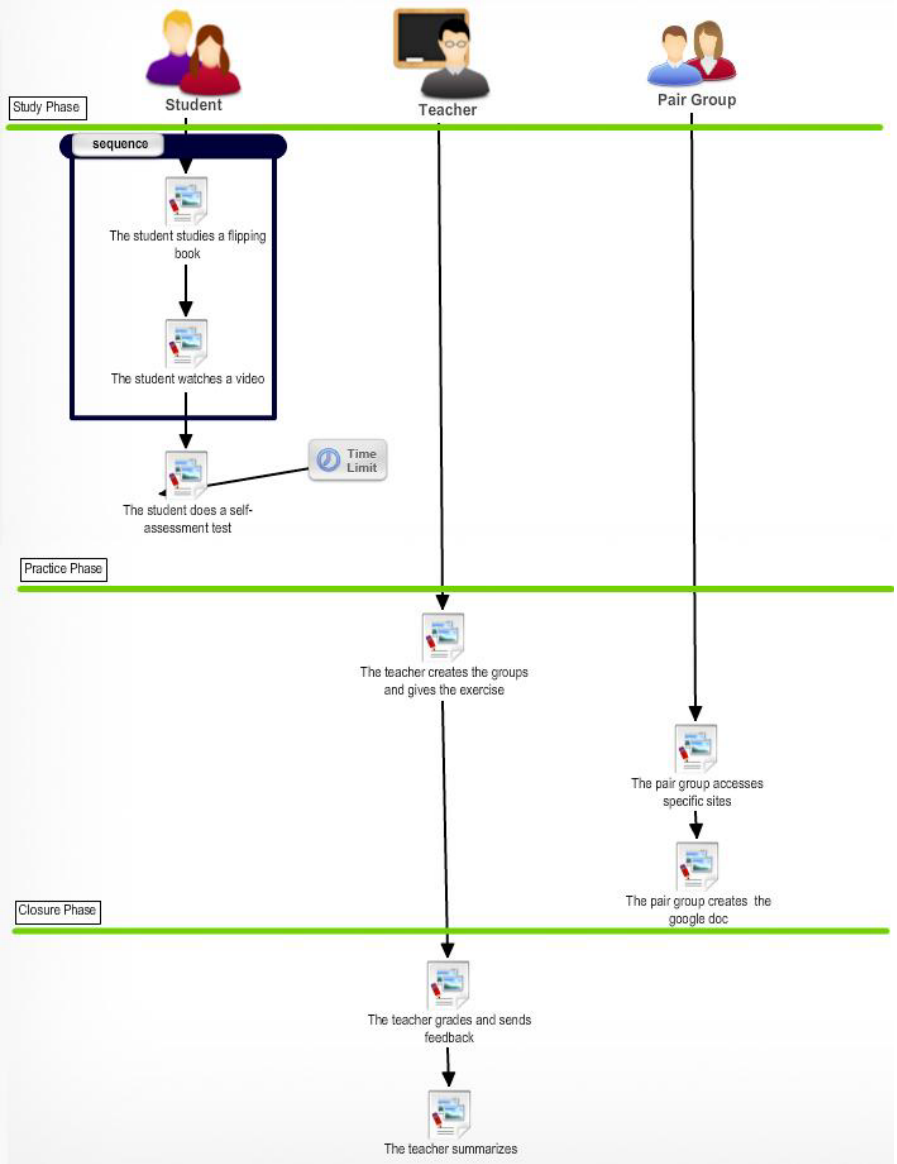

FIGURE 8 – Exemple de diagramme du *flow model* d'un scénario dans CADMOS [Bol+12]

de restauration de cours. D'après les auteurs de CADMOS, le cours déployé peut être ajusté directement dans Moodle ou dans CADMOS.

2.1.5.2 *Flexo*

Le projet Flexo [DVT10] propose un langage de scénarisation pédagogique intégrant des éléments sémantiques liés à la description des activités ainsi qu'à leur séquencement. Il propose des extensions traitant la représentation des rôles, l'intégration de web-services et d'activités spécifiques à la plate-forme cible. A l'instar de CADMOS, Flexo propose une compatibilité avec IMS-LD et Moodle. La particularité de Flexo est de proposer un langage pivot au format textuel (tel

```
course courseID {
  assessment_activity assessmentActivityID {
    @moodle:chat
    @moodle:text "This is a moodle activity"
    @moodle:visible yes
    @moodle:view_past_sessions yes
    simple_activity simpleActivityID {
      title "Activity title"
      description "Activity description"
      resource "http://..."
    }
    assessment {
      //Evaluation params
      num grade [0,10]
      boolean finished
      collection scale ["good", "regular", "bad"]
    }
    flow:
      if(grade > 5 && finished) {
        then: otherActivity
      } else {
        flow: anotherActivity
      }
  }
}
```

FIGURE 9 – Exemple d'un scénario au format texte dans Flexo [DVT10]

que dans la figure 9) qui peut être soit manipulé directement, soit via un éditeur graphique à base de diagramme (figure 10).

Le modèle conceptuel de Flexo, tel que visible dans la figure 11, est centré sur le concept d'activité qui peut être décliné en activité simple ou composite. Il est possible d'associer des ressources aux activités (qui sont donc l'implémentation de celles-ci). La dynamique du modèle est liée au concept d'activité d'évaluation (*AssessmentActivity*) auquel il est possible de greffer des conditions (*Flow et Expression*) afin de proposer des branchements dans le déroulement du scénario.

Le langage de modélisation graphique reprend ces concepts : activité simple/composite, évaluation, activité d'évaluation, conditions etc. La structure principale est l'activité d'évaluation représentée par un bloc contenant une activité (simple ou composite) et une évaluation. Ce bloc peut alors être chaîné à un autre de façon linéaire ou exploiter le branchement conditionnel (voir exemple de la figure 10). La version textuelle du modèle de scénario suit un format spécifique (illustré dans la figure 9) et peut être annotée pour spécifier l'implé-

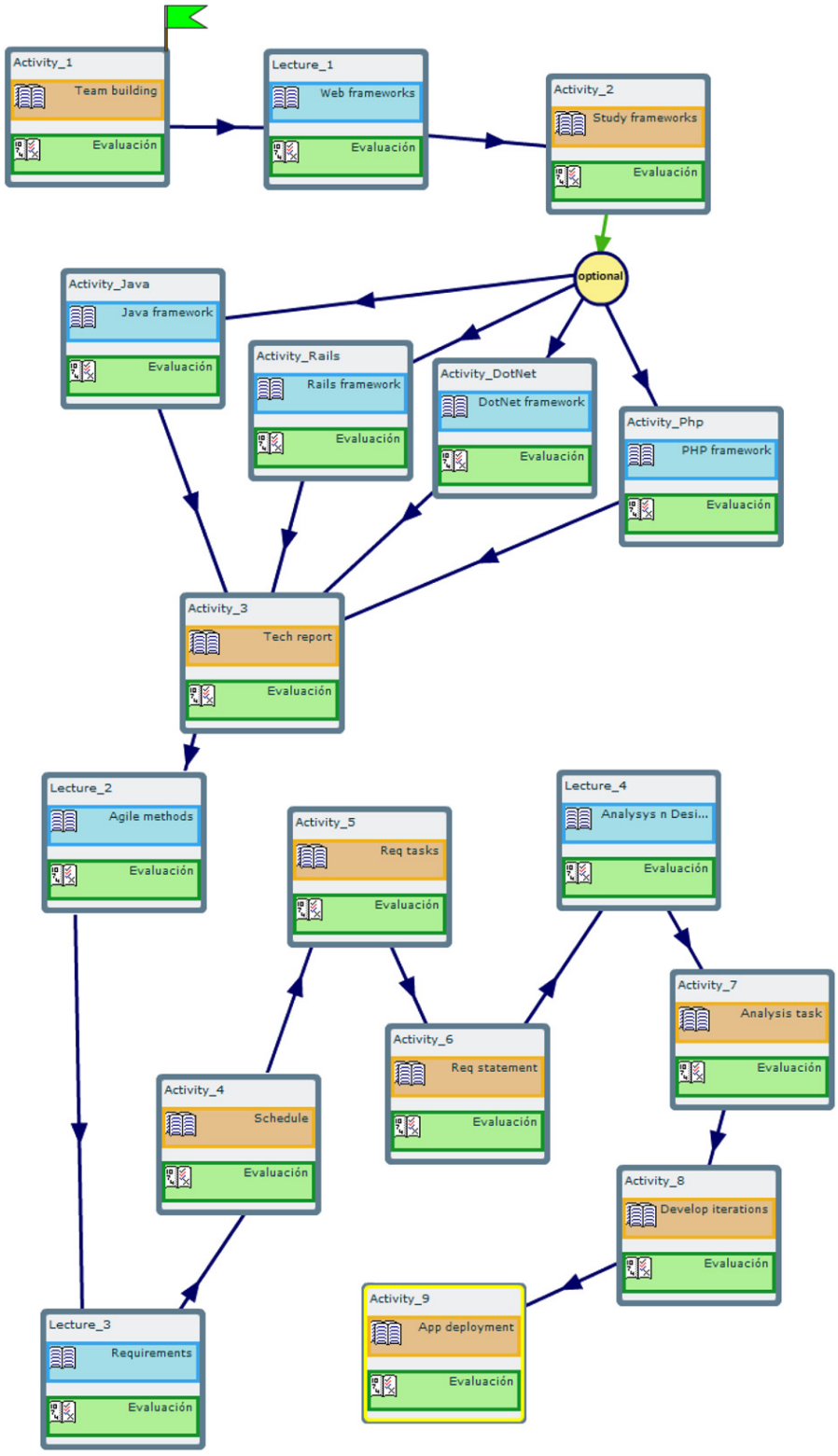

FIGURE 10 – Exemple de diagramme d'un scénario dans Flexo [DVT10]

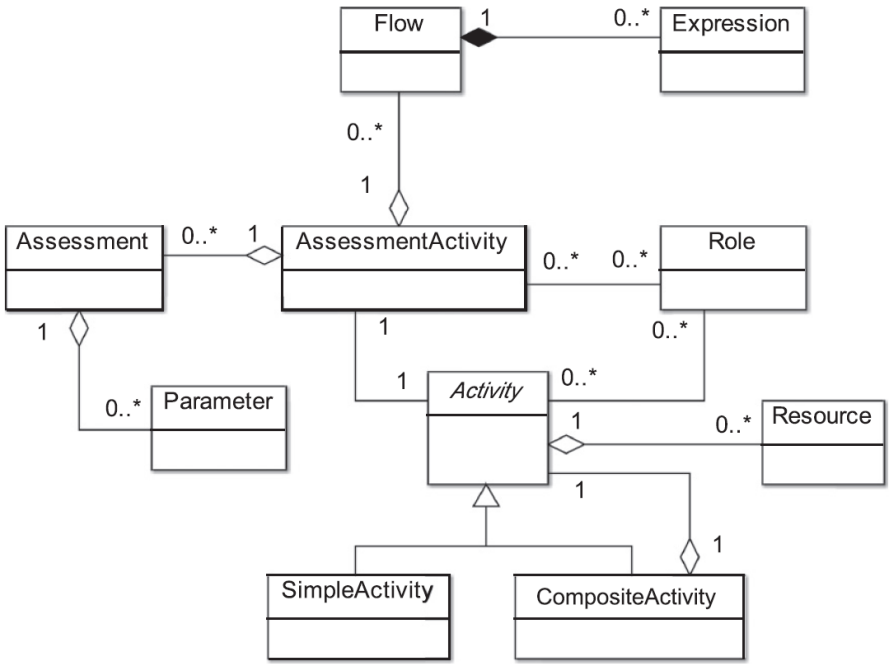

FIGURE 11 – Modèle conceptuel de Flexo [DVT10]

mentation de chacune des activités du scénario sur la plate-forme cible.

2.1.5.3 *Glue !PS*

Glue !PS [Pri+11] n'est pas à proprement parlé un VIDL, il s'agit plutôt d'un *middleware* qui vise à permettre l'interopérabilité entre les langages de scénarisation pédagogique et les plate-formes de formation. Il permet d'adapter le scénario « abstrait » (*i.e* tel qu'un enseignant le modéliserait, sans se préoccuper des contraintes liées à la plate-forme de mise en œuvre) en fonction des besoins d'opérationnalisation, puis de déployer le scénario sur la plate-forme (de façon semi-automatisée). Comme illustré dans la figure 12, présentant l'architecture du système, Glue !PS propose une approche générique reposant sur des composants adaptateurs pour assurer la compatibilité avec les différents langages de scénarisation et les différentes plate-formes de formation. Il utilise également un modèle de données intermédiaire, afin d'assurer l'interopérabilité. La figure 13 présente un exemple de *mapping* entre les éléments du langage de scénarisation et ceux de la plate-forme via le modèle de données intermédiaire. Dans cet exemple, le scénario originel (noté *learning-design :L3-Astronomy*) a été préalablement modélisé avec IMS-LD. Les différents rôles (entourés dans la figure) présents dans le scénario sont automatiquement traduits en rôles identiques dans le modèle de données intermédiaires, bien que la distinction *learner/staff* soit perdue. L'enseignant doit ensuite intervenir manuellement pour compléter le modèle in-

termédiaire avec des informations sur la constitution des groupes en fonction des rôles. Enfin, sur Moodle seront reportés les différents groupes et groupements correspondants.

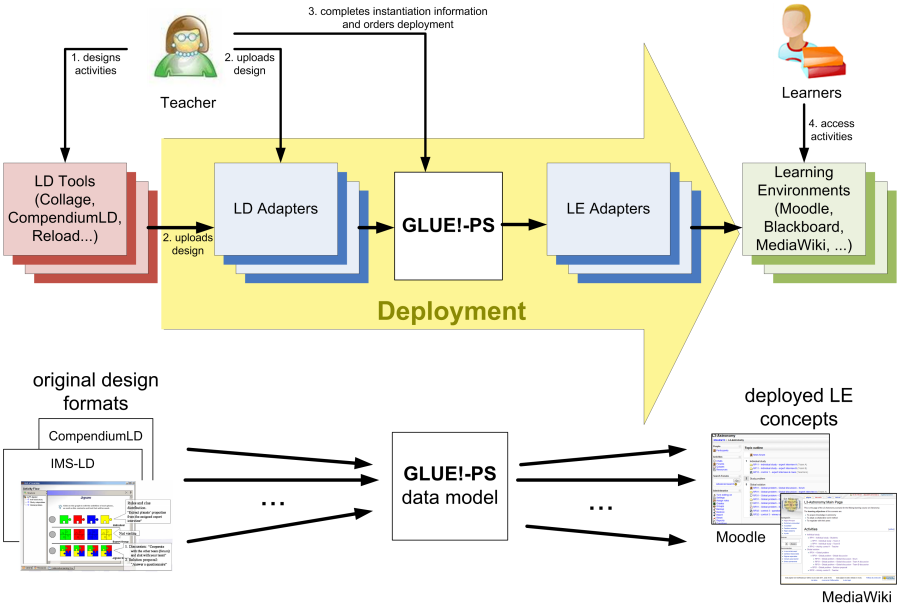

FIGURE 12 – Architecture du projet Glue !PS [Pri+11]

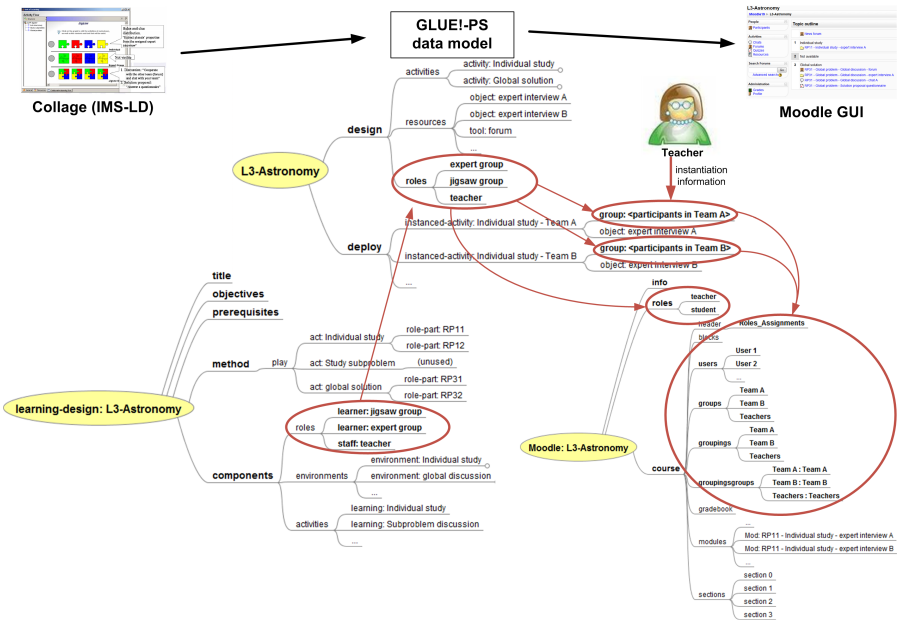

FIGURE 13 – Exemple de *mapping* via le modèle de données de Glue !PS [Pri+11]

2.1.6 *Analyse et critique des solutions existantes*

Le tableau 1 synthétise le positionnement des propositions de CADMOS et Flexo selon la classification de Botturi (Glue !PS n'est pas considéré car ce n'est pas un langage en soi).

TABLE 1 – Positionnement de CADMOS et Flexo selon Botturi

	CADMOS	**Flexo**
Stratification	à plat	à plat
Formalisation	formel	formel
Elaboration	spécification	spécification /implémentation
Perspectives	plusieurs	une seule
Notation	visuelle	visuelle

Le tableau 2 présente le positionnement des travaux étudiés selon les critères proposés par Nodenot.

TABLE 2 – Positionnement de CADMOS, Flexo et Glue !PS selon les critères de Nodenot

	CADMOS	**Flexo**	**Glue !PS**
Rôles et responsabilités	+		+
Modalités			
Connaissances			
Objectifs	+	+	+
Structure	+	+	+
Collaboration	+		
Liens avec l'infrastructure	+	+	+

Dans le cas de Glue !PS, les critères ont été évalués à partir du modèle de données intermédiaire.

Enfin, le tableau 3 positionne ces trois propositions selon les catégories de Laforcade. Il est à noter que Flexo et CADMOS proposent

TABLE 3 – Positionnements de CADMOS, Flexo et Glue !PS selon les catégories de Laforcade

	CADMOS	**Flexo**	**Glue !PS**
Centration	abstraite	abstraite	abstraite
Notation	visuelle	visuelle	informatique

à la fois une notation visuelle, pour l'humain, et une notation informatique : en XML pour CADMOS et dans un format texte spécifique pour Flexo.

CADMOS propose un éditeur graphique simple, reposant sur deux diagrammes. Le premier diagramme (figure 7) (*conceptual model*), uti-

lise des icônes pour représenter les activités, néanmoins ces icônes sont identiques quelle que soit l'activité, la différenciation est faite uniquement entre activité, activité composite et ressource. Le second diagramme (figure 8) exploite le placement horizontal (dans les *swim lanes*) pour représenter les différents acteurs/rôles et le découpage vertical en phases pour l'ordonnancement. Cette représentation est couramment utilisée dans les diagrammes de type organigramme, et s'associe facilement à une description de la dynamique d'un modèle. L'environnement CADMOS a l'avantage de proposer à la fois un export vers IMS-LD et Moodle, ce qui lui permet de conserver une compatibilité avec les plate-formes d'exécution de scénarios IMS-LD (peu utilisées en réalité) tout en étant compatible avec une des plate-formes les plus déployées au sein des institutions académiques (Moodle). Cette double compatibilité a un coût : le langage de scénarisation proposé doit se contenter du dénominateur commun entre la sémantique de Moodle et celle d'IMS-LD et tout ajout à cette intersection doit trouver un équivalent dans les deux domaines.

Comme l'illustrent les figures 14 et 15, le *mapping* entre les concepts du langage de scénarisation proposé par CADMOS et les concepts liés au métier de conception de Moodle est très simple et présente deux inconvénients majeurs. Il est fixe, ne peut être modifié par l'enseignant pendant la conception ni être adapté en fonction de la communauté de concepteurs visée, et ne propose presque aucune abstraction au métier de conception de Moodle. Le principal ajout concerne les éléments structurels (phase, tâches et tâches composites).

De façon similaire, Flexo propose une compatibilité avec IMS-LD et Moodle. Le mapping entre concepts du langage et concepts de la plate-forme n'est en revanche pas fixé (pour les activités) : l'enseignant-concepteur doit soit manipuler le fichier texte soit utiliser une interface à base de formulaires pour le spécifier. Cette approche apporte plus de flexibilité quant au choix de l'implémentation d'une activité pédagogique mais ne permet pas d'assister l'enseignant concepteur dans cette tâche. Chaque implémentation d'activité doit être spécifiée et non chaque type d'activité, ce qui peut constituer une tâche fastidieuse si le scénario est conséquent. Au niveau des structures, l'activité d'évaluation est reliée à un *topic* d'un cours dans Moodle. La structuration centrée sur l'évaluation est une orientation forte du processus de conception qui peut ne pas correspondre aux méthodes développées par les enseignants et peut les contraindre dans leur conception. De même que dans CADMOS, le type d'activité n'est pas sensiblement dénoté dans la représentation visuelle (au delà du label et du type composite/simple/évaluation) ni même modélisé au delà de la spécification de l'implémentation, ce qui n'incite pas l'enseignant-concepteur, en particulier débutant, à varier les activités pédagogiques ou à choisir l'activité la plus pertinente.

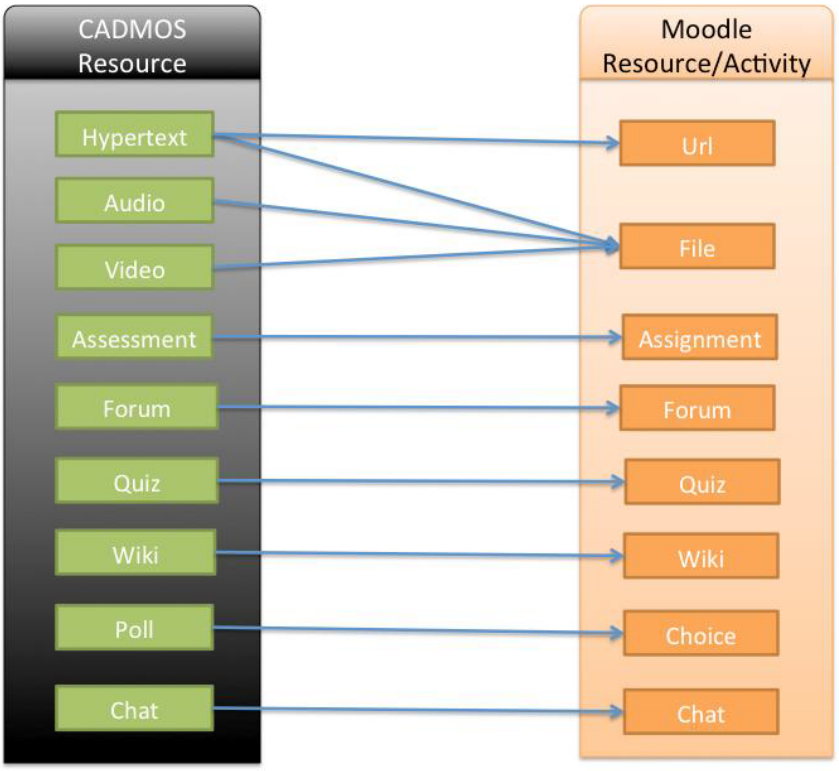

FIGURE 14 – Mapping entre CADMOS et Moodle pour les éléments de type ressource ou activité [Bol+12]

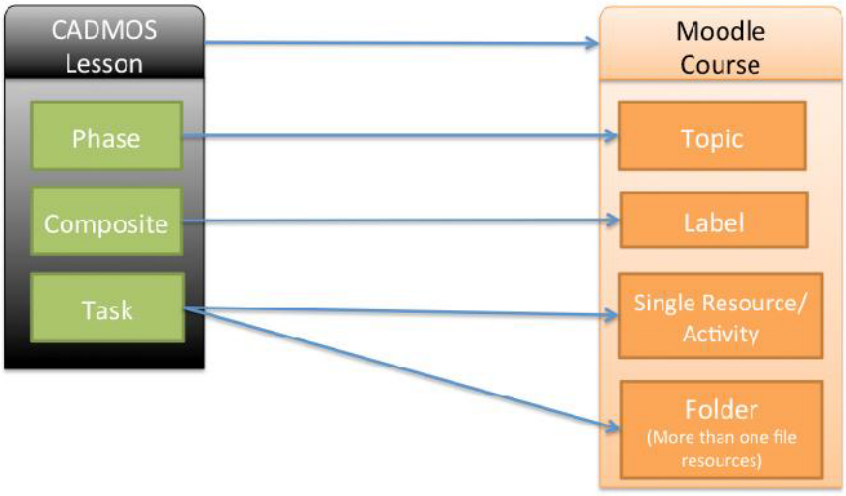

FIGURE 15 – Mapping entre CADMOS et Moodle pour les éléments structurels [Bol+12]

Glue !PS propose un modèle de données dont l'objectif n'est pas d'être un nouveau VIDL mais plutôt de représenter les fonctionnalités les plus communes dans les divers modèles de conception qui sont supportés dans les environnements d'apprentissage. Les auteurs de Glue !PS ont conscience que ce modèle de données intermédiaire est un facteur contraignant de leur approche puisqu'il détermine quelles conceptions peuvent être exprimées et déployées avec Glue !PS (et quel niveau de fidélité entre le modèle initial et le modèle final). Ils considèrent que ce modèle de données est suffisamment expressif pour fournir des traductions acceptables des conceptions initiales de l'enseignant pour un LMS spécifique [Pri+11]. Ils reconnaissent une perte d'information en général mais que dans la plupart des cas les déploiements correspondants possèdent les fonctionnalités essentielles du modèle de conception initial (à condition que le LMS soit compatible). Cette approche présente l'avantage de proposer une liberté totale sur le langage de scénarisation employé tout en garantissant l'opérationnalisation des scénarios, en particulier sur Moodle. Néanmoins, le développement d'adaptateurs peut s'avérer coûteux, et l'utilisation d'un modèle de données intermédiaire fixe, restreint fortement les possibilités de *mapping* entre langage de scénarisation et plate-forme.

2.2 LES VIDL D'UN POINT DE VUE DOMAIN SPECIFIC MODELING

Comme présenté dans la section 1.1.3, le projet GraphiT exploite l'IDM comme cadre méthodologique et utilise le *Domain Specific Modeling* comme cadre théorique et technique. La section 1.1.4 présente de façon générale l'IDM appliquée au contexte du projet. Cette section présente plus en détails en quoi consiste un langage de modélisation dans le cadre du DSM et comment nous l'appliquons aux langages de scénarisation pédagogique.

Selon [JCV12], un langage de modélisation est défini de la façon suivante :

> *Un langage de modélisation (L_m) est défini selon le tuple $\{AS, CS*, M_{ac}*, SD, M_{as}\}$ où AS est la syntaxe abstraite, $CS*$ est la (les) syntaxe(s) concrète(s), $M_{ac}*$ est l'ensemble des mappings de la syntaxe abstraite vers la (les) syntaxe(s) concrète(s), SD est le domaine sémantique et M_{as} est le mapping de la syntaxe abstraite vers le domaine sémantique.*

L'IDM favorise la définition de langages de modélisation dédiés à un domaine particulier (*Domain Specific Modeling Languages*) offrant ainsi aux utilisateurs des concepts propres à leur métier et dont ils ont la maîtrise. Ces langages sont généralement de petite taille et doivent être facilement manipulables, transformables, combinables, etc. Selon ces principes, la définition d'un système complexe fait généralement appel à l'utilisation de plusieurs DSML ayant des relations entre eux,

restreignant ainsi l'ensemble des combinaisons valides des modèles conformes à ces différents méta-modèles (c.-à-d. construits à l'aide de ces différents DSML).

Dans le cadre du projet GraphiT, deux sortes de langages de modélisation visuels sont visés :

— un L_m de premier niveau (appelons le L_{m1}) s'appuyant directement sur le méta-modèle de la plate-forme afin de proposer une conception de scénario de cours en relation directe avec le métier de conception de la plate-forme

— un L_m de second niveau (L_{m2}) dérivé du méta-modèle de la plate-forme afin de proposer une conception de scénario pédagogique plus expressive en relation indirecte avec le métier de conception de la plate-forme.

On a donc $L_{m1} = \{AS, CS, M_{ac}, SD, M_{as}\}$ où AS est le méta-modèle qui sera formalisé pour une plate-forme donnée, CS est une syntaxe concrète graphique orientée diagrammes.

De même, $L_{m2} = \{AS, CS, M_{ac}, SD, M_{as}\}$ mais avec cette fois-ci AS qui sera un nouveau méta-modèle dérivé de celui de la plate-forme, et CS est une autre syntaxe concrète diagrammatique.

L_{m1} et L_{m2} sont deux DSML qui permettent d'adresser la scénarisation pédagogique de situations d'apprentissage pour une même plate-forme de formation mais à des niveaux d'abstraction différents. L_{m1} n'entre pas dans le périmètre de cette thèse et a été proposé dans des travaux antérieurs [Abe13a].

2.2.1 *Syntaxe abstraite*

La syntaxe abstraite (AS) d'un langage de modélisation exprime, de manière structurelle, l'ensemble de ses concepts et leurs relations. Les langages de méta-modélisation tels que le standard MOF [2] de l'OMG, offrent les concepts et les relations élémentaires qui permettent de décrire un méta-modèle représentant la syntaxe abstraite d'un langage de modélisation. Pour définir cette syntaxe, nous disposons à ce jour de nombreux environnements et langages de méta-modélisation : Eclipse-EMF/Ecore [Fou16b], GME/MetaGME [Led+01], AMMA/KM3 [JBT06] ou encore Kermeta [MFJ05]. Tous ces langages reposent toutefois sur les mêmes constructions élémentaires. S'inspirant de l'approche orientée objet, les langages de méta-modélisation objet offrent le concept de classe pour définir les concepts d'un DSML. Une classe est composée de propriétés qui la caractérisent. Une propriété est appelée référence lorsqu'elle est typée par une autre classe, et attribut lorsqu'elle est typée par un type de donnée (par exemple booléen, chaîne de caractère ou entier).

Les langages de modélisation que nous visons sont basés sur des syntaxes abstraites à base de méta-modèle. Nous utilisons l'environ-

2. MetaObject Facility [Gro16a]

nement de méta-modélisation EMF/Ecore, qui a l'avantage d'avoir tout un écosystème d'outils et de langages associés qui permettent facilement de manipuler les modèles et méta-modèles décrits avec EMF/Ecore. Des travaux passés dans notre équipe de recherche [CTC13] [Abe13a] [Our12] ont déjà exploité EMF/Ecore et nous souhaitons mettre à profit cette expérience.

2.2.2 *Syntaxe concrète*

Les syntaxes concrètes (CS) d'un langage fournissent à l'utilisateur un ou plusieurs formalismes, graphiques ou textuels, pour manipuler les concepts de la syntaxe abstraite. Le modèle ainsi obtenu sera conforme à la structure définie par la syntaxe abstraite. La définition d'une syntaxe concrète consiste à définir un des mappings de $M_{ac}*$, et permet ainsi d'annoter chaque construction du langage de modélisation définie dans la syntaxe abstraite par une (ou plusieurs) décoration de la syntaxe concrète et pouvant être manipulée par l'utilisateur du langage. La définition du modèle d'une syntaxe concrète est à ce jour bien maîtrisée et outillée. Il existe en effet de nombreux projets qui s'y consacrent, principalement basés sur EMF (*Eclipse Modeling Framework*) : GMF (*Graphical Modeling Framework*)[Fou16d], Graphiti[Fou16e], GEMS [Fou16c], Sirius [Fou16g], etc. Si ces derniers sont principalement graphiques, des projets récents permettent de définir des modèles de syntaxe concrète textuelle. Nous citons par exemple les travaux des équipes INRIA Triskell (Sintaks) et ATLAS (TCS) qui proposent des générateurs automatiques d'éditeurs pour des syntaxes concrètes textuelles ; il existe également un projet open-source basé sur l'environnement Eclipse et compatible avec EMF nommé XText. Ces approches génératives, en plus de leurs qualités de généricité, permettent de normaliser la construction des syntaxes concrètes.

Le langage de modélisation que nous visons est basé sur une syntaxe concrète graphique dérivée du méta-modèle. Nous utilisons l'environnement de modélisation Sirius (pour L_{m2}) pour définir celle-ci. Sirius est un framework similaire au plus connu GMF (*Graphical Modeling Framework*, qui se base sur des méta-modèles au format Ecore réalisés avec EMF) mais se distingue dans son approche interprétée en opposition à GMF qui nécessite de générer du code java pour les éditeurs graphiques et la manipulation de modèles. Le framework GMF a déjà été utilisé dans d'autres projets de l'équipe impliquée dans le projet GraphiT [CTC13] [Abe13a]. Il a été utilisé pour L_{m1} le langage de modélisation graphique construit directement au dessus du méta-modèle de conception pédagogique de la plate-forme de formation. Le framework Sirius apparaît pertinent par son approche interprétée pour explorer, par prototypage, la conception de langage de scénarisation plus abstrait.

2.2.3 *Sémantique*

La seule volonté de définir des modèles contemplatifs utilisés uniquement pour communiquer avec des constructions plus abstraites sur le système, ne demande pas une définition plus complète du langage de modélisation. En effet, les syntaxes abstraite et concrète sont suffisantes pour manipuler graphiquement (ou textuellement) un langage et réfléchir de manière plus abstraite à la construction du système. Dans ce contexte, ce sont les noms choisis pour nommer les concepts qui sont le support de la sémantique, et donc de la réflexion. Toutefois, la manipulation automatique des modèles par des outils (pour des simulations, vérifications, tests, générations, etc.) nécessite une formalisation de la sémantique jusque là implicite. La sémantique d'un langage définit de manière précise et non ambiguë la signification des constructions de ce langage. Elle permet ainsi de donner un sens précis aux modèles construits à partir de celui-ci. Une sémantique est dite formelle lorsqu'elle est exprimée dans un formalisme spécifique et permet de vérifier la cohérence et la complétude de cette définition. Définir la sémantique d'un langage revient à définir le domaine sémantique et le mapping M_{as} entre la syntaxe abstraite et le domaine sémantique. Le domaine sémantique définit l'ensemble des états atteignables par le système, et le mapping permet d'associer ces états aux éléments de la syntaxe abstraite. Dans le contexte de l'IDM, au même titre que les autres éléments d'un langage de modélisation, la définition du domaine sémantique et du mapping prend la forme d'un modèle.

Dans le cadre de ces travaux, nous nous contenterons de définir la sémantique statique à travers les constructions de nos méta-modèles, les éventuelles règles de bonne formation complémentaires, et des informations supplémentaires en langage naturel.

2.3 TRAVAIL EXPLORATOIRE

En amont des travaux présentés dans ce manuscrit, un travail exploratoire [LL13] a été réalisé, sur une période de six mois, afin d'étudier la viabilité de l'approche *Domain Specific Modeling* pour la spécification de langage de scénarisation pédagogique, et d'identifier des techniques DSM permettant de garantir l'opérationnalisation des scénarios produits. Comme présenté dans la section précédente, il faut considérer deux langages de modélisation, l'un étant le VIDL que nous visons, l'autre le langage « de la plate-forme » qui permet l'opérationnalisation. L'enjeu est donc de rendre *compatibles* ces deux langages en intervenant sur les différents modèles composant un langage de modélisation (syntaxe abstraite, syntaxe concrète, mappings).

Trois approches ont pu être identifiées, nous les présenterons une à une puis reviendrons sur l'analyse qui a permis de choisir la solution que nous appliquons aujourd'hui dans le cadre de notre thèse.

2.3.1 *Approche basée sur la syntaxe concrète (1)*

Cette première approche consiste à ne modifier que la syntaxe concrète du langage et son mapping avec la syntaxe abstraite. Le méta-modèle de la plate-forme, en l'occurence Moodle, est conservé tel quel dans le langage d'opérationnalisation (voir figure 16). Les figures, icônes, labels, propriétés représentant la palette et les éléments graphiques peuvent être définis de manière à cacher les concepts de la plate-forme sous-jacents, ceci afin de proposer une représentation offrant une signification davantage pédagogique en accord avec les éléments que le langage doit offrir. Cette approche convient pour les cas tel qu'une activité d'auto-évaluation, où un seul outil de la plate-forme est employé pour mettre l'activité en place (mapping 1 à 1, ici l'outil *test*), mais s'avère problématique lorsque plusieurs outils peuvent être employés, auquel cas des critères de sélection de l'outil de la plate-forme sont nécessaires. Les outillages DSM ne permettent pas actuellement à partir d'un outil de la palette de l'éditeur d'instancier dynamiquement un concept différent en fonction de paramètres complexes. La seule alternative possible serait donc de proposer un élément de palette par combinaison de paramètres possible pour une activité dans l'éditeur. Cette solution peut rapidement amener à une surchage de l'interface utilisateur.

2.3.2 *Approche basée sur l'extension (sans persistance) (2)*

Les techniques utilisées dans l'approche 1 peuvent être exploitées à nouveau pour traiter les cas simples de correspondance entre un outil de la plate-forme et un concept pédagogique. Afin de palier aux cas menant à une surcharge de l'interface, il est possible d'étendre le méta-modèle de la plate-forme initial avec des nouveaux concepts (par exemple l'activité *Débat* avec une propriété booléenne *synchrone*) et lui associer des représentations graphiques. L'extension du méta-modèle initial (voir figure 17) entraîne la non-conformité des futurs modèles, et empêche donc leur opérationnalisation potentielle. Toutefois, nous pouvons intervenir à deux niveaux : au niveau de la définition même de ces extensions en positionnant un attribut *transient* sur chaque concept ajouté au méta-modèle de Moodle, afin d'empêcher sa persistance (exploitation des possibilités offertes par les langages de spécification de méta-modèle), et au niveau du code généré par l'outillage DSM. Ainsi, pour l'activité débat par exemple, si la propriété synchrone est vraie, l'élément *Débat* sera ignoré lors de la persistance mais remplacé par un élément *chat*.

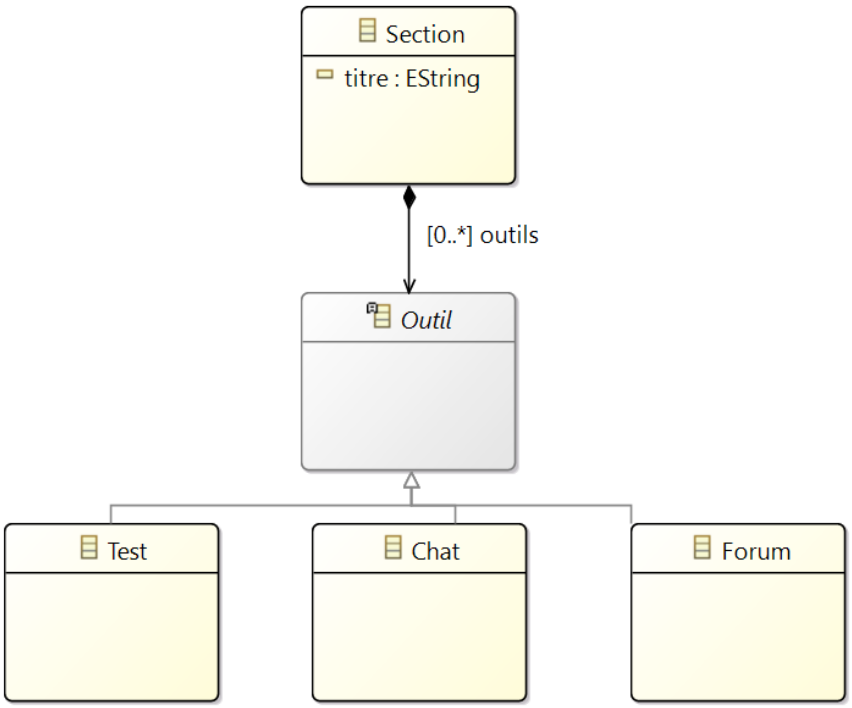

FIGURE 16 – Représentation partielle illustrant la formalisation de la syntaxe abstraite selon l'approche 1.

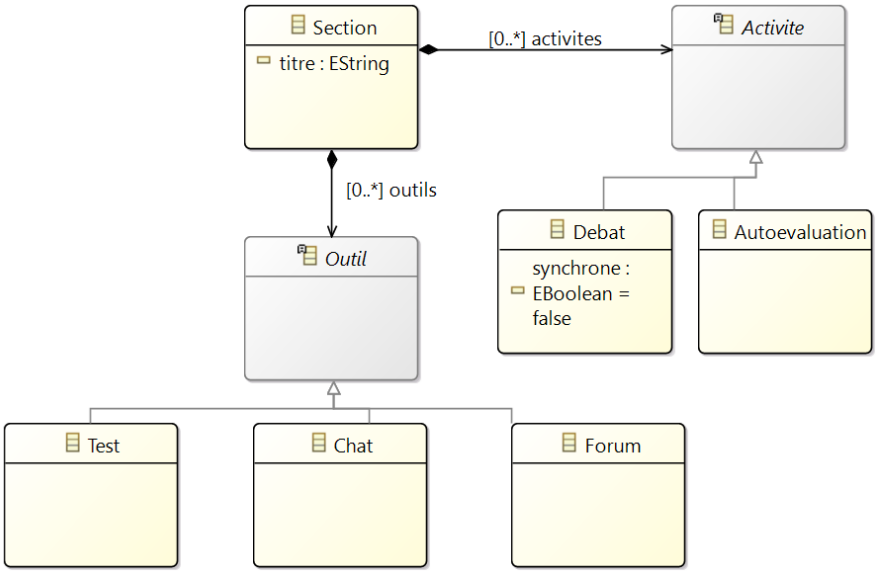

FIGURE 17 – Représentation partielle illustrant la formalisation de la syntaxe abstraite selon l'approche 2.

2.3.3 *Approche basée sur l'extension (avec persistance) (3)*

Cette approche s'appuie sur la formalisation d'un méta-modèle spécifique aux besoins et pratiques des enseignants-concepteurs (voir

figure 18). Ce nouveau méta-modèle spécifie ainsi les différentes activités pédagogiques retenues pour le langage, ainsi que des éléments supplémentaires liés à la structuration du scénario ou à la définition d'objectifs pédagogiques. Ces derniers ont été ajoutés arbitrairement afin de montrer que cette approche ne cherche pas à réaliser, dans un premier temps, de correspondances avec les services offerts par la plate-forme. La seule limite de cette spécification est celle des besoins à formaliser. Comme pour les autres approches, le processus consiste à spécifier les différents modèles pour la définition de la notation graphique, la palette, etc. Les futurs modèles/scénarios produits par l'éditeur n'ont aucune conformité avec le méta-modèle initial de l'approche basée sur la syntaxe concrète, et donc aucune assurance d'être opérationnalisable. Le méta-modèle représentatif de la plate-forme n'étant pas inclus dans ce méta-modèle spécifique, il est difficile d'opérer, comme dans l'approche précédente, par modification du code gérant la persistance. La transformation de modèles est une solution envisageable mais qui nécessite une étape supplémentaire, a posteriori de l'activité de scénarisation et a priori de l'opérationnalisation.

Les modèles produits par les éditeurs sont conservés au format XML, ce qui rend également applicable des techniques de transformations orientées graphe de type XSLT. Dans nos travaux nous avons expérimenté les règles ATL. Celles-ci sont plus ou moins complexes selon le niveau de correspondances à prendre en compte. Aussi, ces dernières doivent avoir été explicitées et précisées. Il est possible également que des éléments faisant sens au niveau de la conception pédagogique ne trouvent pas d'éléments correspondants sur la plate-forme, même par détournement des usages (traductions partielles produisant des scénarios incomplets / inconsistants), ou, à l'inverse, que les éléments sources soient insuffisants pour déterminer des choix de projection. Par exemple, les objectifs pédagogiques de notre cas d'étude pourraient ne pas faire sens sur Moodle et seraient alors perdus au moment de l'opérationnalisation. Selon l'importance de ces éléments les pertes peuvent conduire à rendre les autres éléments traduits incohérents.

2.3.4 *Analyse des trois approches*

L'application de l'outillage DSM selon les trois approches a donné lieu à une analyse selon quatre critères :

1. l'expressivité visuelle, correspondant au potentiel d'expressivité sémantique des modèles tel que perçu par les praticiens au travers de la manipulation de la notation visuelle
2. l'expressivité abstraite, correspondant à la sémantique des modèles, en relation avec la syntaxe abstraite sous-jacente, non perceptible directement par les praticiens

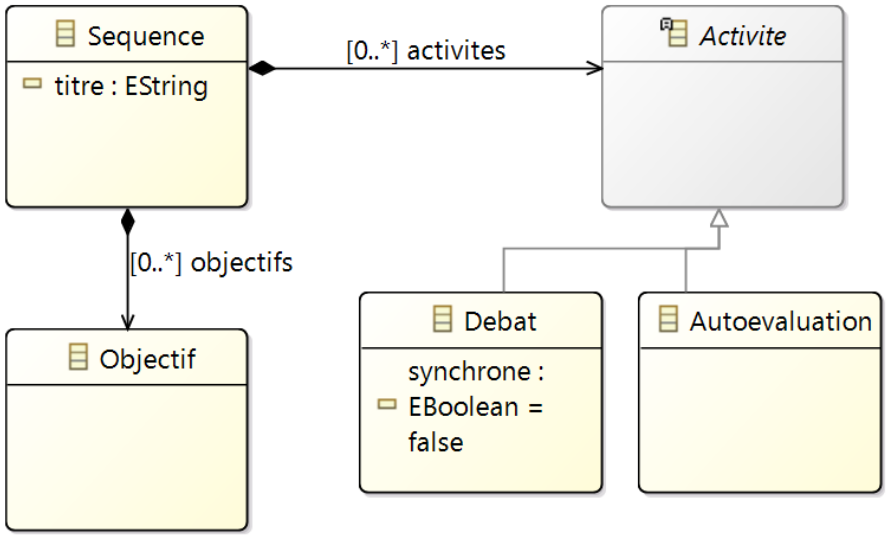

FIGURE 18 – Représentation partielle illustrant la formalisation de la syntaxe abstraite selon l'approche 3.

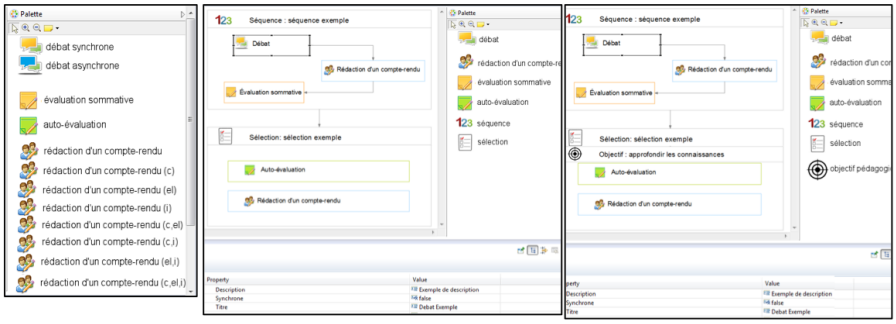

FIGURE 19 – Captures d'écran partielles des éditeurs obtenus avec l'approche 1 (gauche), 2 (centre) et 3 (droite)

3. le potentiel d'opérationnalisation des modèles produits
4. la sémantique du scénario une fois opérationnalisé (i.e. une fois traduit dans le dispositif en termes de configuration, structuration des outils/services/concepts/propriétés)

Le premier critère est le seul qui ne nécessitait pas d'expertise et de compétences informatiques et qui relevait d'un point de vue usager. Pour ce critère nous avons recueilli, sous la forme d'un entretien informel participatif, des retours qualitatifs de la part d'experts pédagogiques. La table 1 synthétise les résultats obtenus, annotés +, / ou - selon leur rapport aux deux objectifs de l'étude (expressivité selon le métier de conception de la plate-forme et selon les pratiques à proposer).

Du point de vue de l'expressivité perçue au travers du formalisme graphique, les approches 2 et 3 sont mieux adaptées car elles nécessitent moins de choix et réflexions sur la conception lors de la sé-

TABLE 4 – Comparatif des résultats d'analyse des 3 approches

Critères	**Approche 1**	**Approche 2**	**Approche 3**
Expressivité visuelle	Trop d'éléments (-)	Nombre d'éléments cohérent avec besoins praticiens mais contraint (/)	Nombre d'éléments cohérent avec besoins praticiens (+)
Expressivité abstraite	Limitée à celle capturée de la plate-forme (-)	Limitée à un périmètre proche de la sémantique capturée de la plate-forme (/)	Non limitée (+)
Conformité au méta-modèle de la plate-forme	Directe (+)	Nécessite traduction pendant la scénarisation (/)	Nécessite lourde transformation a posteriori de scénarisation (-)
Sémantique du scénario après mise en conformité	Conservée (+)	Conservée mais contrainte (/)	Scénario dispositif peut être inconsistant/incomplet (-)

lection des éléments de la palette. Par exemple, l'activité débat impose avec l'approche 1 de faire le choix synchrone/asynchrone sur la modalité de réalisation dès la sélection de l'activité ; un changement futur nécessitera de supprimer l'activité débat choisie et de la remplacer par l'autre. La palette de sélection peut ainsi se focaliser sur des éléments de conception dont les propriétés pourront être précisées plus tard. L'approche 3, occultant les aspects conformité au métier de la plate-forme, propose alors des éléments/propriétés et relations de conception directement corrélés aux besoins. Elle se révèle donc mieux adaptée sur ce point que l'approche 2. L'expressivité abstraite des modèles produits est directement corrélée aux choix initiaux définis en relation avec les 3 approches. La mise en conformité des modèles produits était directe pour l'approche 1 alors qu'elle nécessitait des adaptations pour les deux autres approches. Dans l'approche 2 nous avons exploité la formalisation même des méta-modèles selon l'outillage DSM (par exemple la propriété *transient* pour indiquer la non persistance du concept) ainsi que la modification du code généré de l'éditeur pour modifier la persistance du modèle. L'approche 3 nécessite en revanche une mise en correspondance totale avec le méta-modèle de la plate-forme a posteriori de la spécification du modèle. Les techniques de l'approche 2 ne sont plus adaptées. Nous avons procédé par de lourdes et complexes transformations de modèles.

Concernant la sémantique du scénario obtenu en conformité avec le métier de conception de la plate-forme, l'approche 3 ne peut pas garantir que celui-ci sera consistant et cohérent. Des travaux abordant la transformation de modèles entre métier des concepteurs et métier des plate-formes relèvent également ces faiblesses [Nod07] [Abd09]. L'approche 2 garantit le maintien de la sémantique initiale par l'élaboration conjointe de l'extension du méta-modèle et des techniques de mise en conformité ; le maintien est donc fortement dépendant du tissage réalisé pour étendre le méta-modèle initial de la plate-forme. L'approche 1 garantit en revanche le maintien de la sémantique par exploitation directe du méta-modèle initial.

Globalement, l'approche 3 correspond à l'approche traditionnelle d'élaboration d'un VIDL, avec le principal avantage (l'expressivité) mais aussi défaut (difficulté d'opérationnalisation). L'approche 1 a montré les limites d'expressivité de la syntaxe concrète lorsqu'elle est définie par dérivation de la syntaxe abstraite : cette approche permet de donner une à plusieurs représentations différentes pour un même concept et/ou relation issue du métier de la plate-forme mais est contrainte à conserver ce lien fort de dérivation (contrainte intrinsèque au processus de spécification de langage selon l'approche DSM). L'approche 2 est finalement une approche intermédiaire sur l'ensemble des critères : meilleur rapport expressivité / conformité avec le métier de la plate-forme. Elle requiert toutefois une forte expertise en méta-modélisation pour réduire le coût du codage né-

cessaire pour garantir la conformité des modèles. En revanche elle met en évidence que la sémantique et la notation du langage/éditeur doivent être spécifiées en relation forte avec la capacité de traduction vers le métier identifié et formalisé de la plate-forme. La correspondance des besoins/pratiques en termes d'éléments métiers de la plate-forme ne peut se limiter à une traduction seulement informatique et doit impliquer les praticiens : elle doit être également explicitée.

Le cadre méthodologique du *Domain Specific Modeling* mais aussi plus largement de l'Ingénierie Dirigée par les Modèles doit être approfondi à commencer par les pistes mises en lumière par ce travail exploratoire : le tissage entre le métier des praticiens et celui de la plate-forme doit être explicité et formalisé afin que des praticiens puissent vérifier, voire définir, la sémantique associée à la correspondance du scénario pédagogique en terme de conception/configuration de la plate-forme.

2.4 TRANSFORMATIONS ET COMPOSITION DE MODÈLES

Comme présenté dans la section 1.1.4, passer d'un scénario pédagogique abstrait à un cours prêt à être déployé sur une plate-forme revient, sous l'angle de l'Ingénierie Dirigée par les Modèles, à effectuer une transformation de modèle. La sous-section suivante présente plus en détail ce qu'est une transformation de modèle et comment elle peut être utilisée dans notre cas d'étude.

2.4.1 *Transformation de modèle*

2.4.1.1 *Définition*

La transformation de modèle est une opération qui permet, à partir d'un modèle M_a conforme à un méta-modèle MM_a, de produire un modèle M_b conforme à un méta-modèle MM_b. Si MM_a et MM_b sont identiques, la transformation est dite *endogène*, sinon elle est *exogène*. Si la transformation permet de changer le niveau d'abstraction, elle est considérée *verticale*, si M_a et M_b sont de même niveau elle est *horizontale* [Como8]. Il existe de nombreux cas d'utilisation pour les transformations de modèle, en voici quelques exemples (également illustrés dans la figure 20)

— Normalisation (endogène, horizontale)
— Restructuration (endogène, horizontale)
— Raffinement (endogène, verticale)
— Migration (exogène, horizontale)
— Rétro-conception (exogène, verticale)

De nombreux langages sont à ce jour disponibles pour écrire des transformations de modèle. On retrouve d'abord les langages généralistes qui s'appuient directement sur la représentation abstraite du

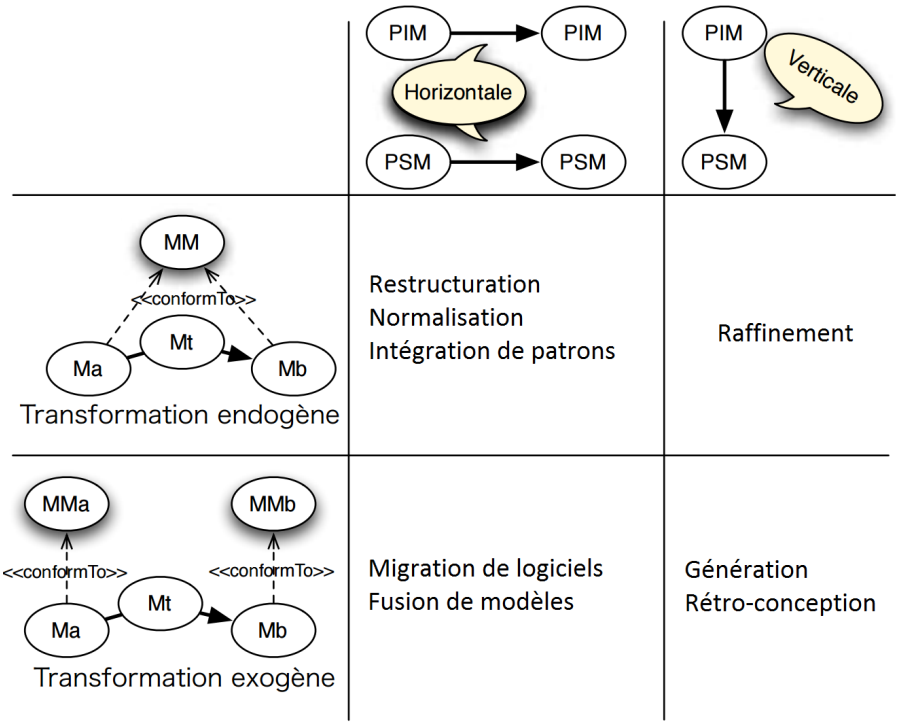

FIGURE 20 – Types de transformation et leurs principales utilisations [Como8]

modèle. On citera par exemple l'API d'EMF qui, couplée au langage Java, permet de manipuler un modèle sous la forme d'un graphe. Dans ce cas, c'est à la charge du programmeur de faire la recherche d'information dans le modèle, d'expliciter l'ordre d'application des règles, de gérer les éléments cibles construits, etc.

Afin d'abstraire la définition des transformations de modèle et rendre transparent les détails de mise en œuvre, l'idée a été de définir des DSML dédiés à la transformation de modèle. Cette approche repose alors sur la définition d'un méta-modèle dédié à la transformation de modèle et des outils permettant d'exécuter les modèles de transformation. Une telle approche est suivie par l'outillage ATL (*ATLAS Transformation Language*) [Jou+06]. Il s'agit d'un langage hybride (déclaratif et impératif) qui permet de définir une transformation de modèle à modèle (appelée Module) sous la forme d'un ensemble de règles. Une transformation prend en entrée un ensemble de modèles (décrits à partir de méta-modèles en Ecore ou en KM3). Afin de donner un cadre normatif pour l'implantation des différents langages dédiés à la transformation de modèle, l'OMG a défini le standard QVT[Gro16c] (*Query/View/Transformation*). Le méta-modèle de QVT est conforme à MOF et OCL [3] est utilisé pour la navigation dans les modèles. Il existe d'autres langages de transformation de

3. Object Constraint Language (OCL) est un langage d'expression de contraintes utilisé entre autre par UML [Gro16b]

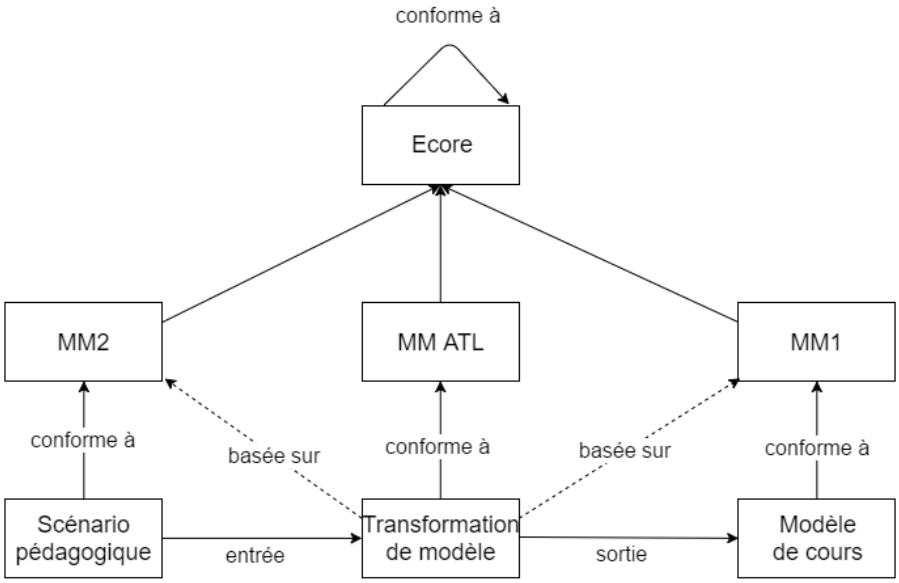

FIGURE 21 – Transformation de modèle entre Lm_2 et Lm_1 pour une plate-forme de formation donnée

modèles comme ETL [Pai+09] du projet Epsilon, Kermeta [MFJ05] permet d'opérer des transformations, GReaT [Bal+07] propose également une approche orientée transformation de graphes.

2.4.1.2 *Application*

Pour rappel, dans le cadre du projet, deux langages Lm_1 et Lm_2 exploitent respectivement le métier de conception de la plate-forme. Ils le font à deux niveaux : le premier s'appuie directement sur le méta-modèle de la plate-forme de formation, le second propose une abstraction supplémentaire afin de proposer aux futurs concepteurs des éléments de conception pédagogique moins proches de la sémantique et du paradigme implicite de conception de la plate-forme et plus proches de leurs pratiques. Toutefois, afin de rendre opérationnalisables sur la plate-forme les modèles conformes à Lm_2 ils devront être importés à travers une API et donc devront être conformes au méta-modèle explicité et formalisé pour la plate-forme. Afin de retrouver cette conformité, il apparaît qu'une première transformation de modèle est nécessaire.

Il s'agit d'une forme de transformation exogène (méta-modèles MM_1 et MM_2 différents) et verticale (différent niveaux d'abstraction) : ce cas de figure est appelé *transformation de génération*. La figure suivante illustre ce cas et précise également l'outillage envisagé : ATL comme outil/méta-modèle de transformation.

Cette transformation permet de rétablir la conformité des modèles après conception d'un scénario, afin de permettre l'opérationnalisation. Il s'agit d'un de nos objectifs, mais ce mécanisme ne répond pas

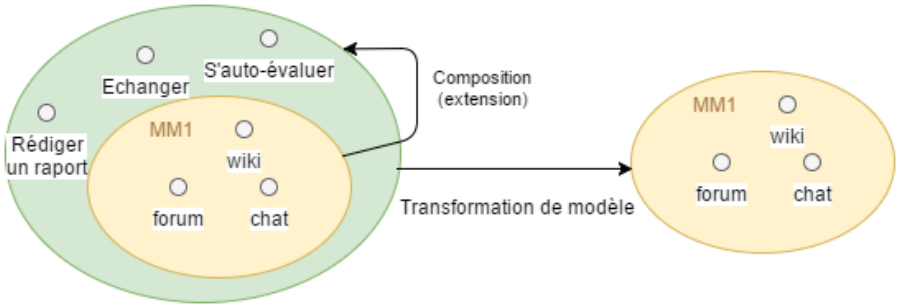

FIGURE 22 – Composition et transformation de modèle entre Lm_2 et Lm_1

à notre problématique d'extension du méta-modèle, telle qu'elle a été abordée dans la section 2.3.4 et précédemment. Le principe étant de concevoir par extension, un nouveau méta-modèle (celui du langage de scénarisation Lm_2) à partir de celui de la plate-forme. Ce type d'opération, dans le cadre de l'IDM, est considéré comme de la composition de modèles, même si dans notre cas, les modèles sont des méta-modèles. La figure 22 synthétise ces problématiques.

2.4.2 *Composition de modèles*

2.4.2.1 *Introduction*

La composition de modèles est une opération qui combine un ou plusieurs modèles au sein d'un seul autre. Elle peut s'appliquer au niveau M1 (modèles) ou au niveau M2 (méta-modèles). Cette approche est issue de la modélisation orientée aspect, qui favorise la séparation de pré-occupations en préconisant la modélisation indépendante des aspects d'un système. L'objectif de la composition de modèles est alors d'unifier, une fois les différents aspects modélisés, l'ensemble des modèles en un modèle global. Selon [Nguo8], il est possible de classifier les approches de composition de modèles selon plusieurs critères :

- la quantité des modèles en entrée.
- le rôle des modèles composés : symétrique ou asymétrique
- le type des modèles composés : structurel ou comportemental.
- l'hétérogénéité des modèles source : composition de même méta-modèle (endogène) ou différents méta-modèles (exogène).
- la quantité des modèles en sortie : un (le modèle composite) ou plusieurs (modèles d'entrée + composite).
- le type de composition : par opérateur ou par relations.
- les éléments de composition.
- la méthode de création du modèle composite.
- l'effet de composition : poids lourd ou léger.

— l'orientation de la composition : transformation ou composition pure
— le domaine de recherche

La composition de modèles étant une thématique de recherche récente, le nombre de solutions outillées pour l'opérer est assez faible, nous pouvons toutefois citer :

— AMW (*ATLAS Model Weaver*) exploite le tissage et la transformation de modèle (à l'aide du langage ATL) [DV09]
— EML (*Epsilon Merging Language*) propose une approche orientée fusion de modèle et exploite les autres langages du projet Epsilon (EOL, ETL) [Pai+09]
— Kompose [Fle+08] propose une approche orientée fusion basée sur les signatures et exploite le langage Kermeta.
— EMF [Fou16b] peut être utilisé pour la composition de modèles, dans une approche à base de relations.

2.4.2.2 *Application*

Dans notre cas, l'opération de composition portera sur **deux** (méta) modèles en entrée : MM1 et MM2. Le concept de rôle des modèles composés se prête en particulier à la modélisation orientée aspect, dans notre cas, nous pouvons considérer que la composition est **symétrique**. Les modèles composés sont **structurels** (il s'agit de méta-modèles). L'opération est **endogène** : les deux méta-modèles sont conformes au méta-méta-modèle Ecore. Le méta-modèle en entrée sera **conservé** inchangé, en plus du méta-modèle composite en sortie. Les critères suivants dépendent de l'implémentation choisie pour la composition. Dans notre cas, la composition se fait au niveaux des méta-modèles, **par relation**, les éléments de composition sont donc des **relations** (*eReferences* en Ecore) et la méthode de création : **l'établissement de références**. Dans le cas de la composition par relations, il s'agit d'une opération **poids-léger** de **composition pure**. Le domaine de recherche dans notre cas serait la **méta-modélisation**.

2.5 BILAN

Dans ce chapitre, nous avons présenté les VIDL, ou langage de scénarisation pédagogiques graphiques, et avons étudié plusieurs de leurs classifications. Nous avons ensuite présenté et analysé trois approches issues de la littérature permettant de modéliser un scénario pédagogique et de l'opérationnaliser sur un LMS, en particulier Moodle. Les caractéristiques étudiées étaient principalement : l'expressivité pédagogique du langage, la perte (ou non) d'informations lors de l'opérationnalisation, la flexibilité du système quant aux paramètres de l'opérationnalisation (quelles fonctionnalités de la plate-forme utiliser, avec quels paramètres, etc.). Cet état de l'art nous a permis de

positionner nos travaux à la fois selon les différentes classifications et vis-à-vis de l'existant.

Dans un second temps, nous avons présenté comment un langage de scénarisation pédagogique peut être considéré comme un DSML et comment doit être conçu un tel langage (syntaxe abstraite, concrète, sémantique etc.). Cette section a permis d'introduire un travail préalable à cette thèse, qui visait à explorer comment le DSM et l'IDM pouvaient permettre d'augmenter l'expressivité d'un langage de scénarisation pédagogique en ajoutant des concepts au dessus d'un méta-modèle du métier de conception d'une plate-forme de formation. Ce travail exploratoire concluait sur la pertinence d'une approche à base d'extension de méta-modèle et de transformation de modèles.

La section suivante s'est donc attachée à présenter les transformations de modèles et en quoi elles peuvent être utiles à notre proposition. Il s'agissait ici de rétablir la conformité des modèles au méta-modèle de la plate-forme, dans une optique d'opérationnalisation, lorsque ceux-ci sont initialement conformes au méta-modèle étendu. L'extension du méta-modèle a ensuite été étudiée sous l'angle de la composition de modèles. Nous avons notamment caractérisé l'opération de composition qui permettra de produire le méta-modèle étendu.

3

APERÇU DES CONTRIBUTIONS

3.1 SYNTHÈSE ET POSITIONNEMENT

Nous avons étudié différentes classification des langages de scénarisation pédagogique graphiques (ou VIDL). Ces classifications permettent de positionner ces langages selon différents points de vue : le contenu du scénario, le type de notation, la structure du langage, le niveau de détails, la sémantique etc. Nous avons étudié trois solutions issues de l'état de l'art qui nous semblaient proches de notre objectif : proposer un langage de modélisation de scénarios formels, permettant l'opérationnalisation sur un LMS, en particulier Moodle. Ces trois solutions ont été présentées et critiquées puis positionnées selon les différentes classifications.

Nous avons alors pu, positionner notre propre approche selon ses mêmes critères. Nous visons à proposer un langage de scénarisation pédagogique **graphique**, **basé** sur la sémantique pédagogique (aussi appelée métier de conception) de la plate-forme cible (Moodle) mais qui propose, par extension, une **expressivité enrichie** afin de permettre une conception plus proche des pratiques des enseignants-concepteurs. Ce langage doit permettre la conception à plusieurs niveaux de granularité, il doit à la fois couvrir la **spécification** du scénario (à un niveau plus pédagogique) et son **implémentation** (à un niveau plus technique). La description à niveau implémentation doit servir l'opérationnalisation du scénario : il doit être suffisamment **formel** et détaillé pour permettre sa mise en œuvre sur la plate-forme. La problématique est alors de définir comment il est possible d'augmenter l'expressivité d'un langage de scénarisation basé sur le métier de conception pédagogique de la plate-forme sans perdre la possibilité d'opérationnaliser les scénarios.

Bien que les solutions de l'état de l'art étudiées semblent répondre, au moins partiellement, à ces exigences, elles ne proposent aucune (ou de manière limitée dans le cas de Flexo) flexibilité dans l'articulation entre éléments de spécification et d'implémentation. Autrement dit, la traduction des éléments de langage en fonctionnalités ou ressources de la plate-forme est fixée par le système et/ou ne permet pas de conserver la sémantique initiale. Il est alors nécessaire d'étudier comment formaliser ces *mappings* (ou correspondances) et comment permettre leur personnalisation.

Le postulat du projet GraphiT est que l'IDM et le DSM apportent un cadre théorique et technique qui nous permette de répondre à ces problématiques. Nous avons donc étudié en quoi un langage de scénarisation peut être considéré comme un langage de modélisation au sens du DSM et comment il est structuré. La sémantique pédagogique, le métier de conception, est principalement portée par la syntaxe abstraite du langage de modélisation et est formalisée dans un méta-modèle. La notation visuelle du langage est définie dans sa syntaxe concrète. Nous avons présenté un travail préalable qui a

étudié comment ces deux constituants du langage de modélisation peuvent être exploités pour augmenter son expressivité, en particulier ajouter des éléments abstraits basés sur ceux existant au sein du métier de conception de la plate-forme. Cette étude conclut sur la pertinence d'étendre le méta-modèle du métier de conception de la plate-forme, et d'exploiter les transformations de modèles afin de rétablir la conformité des modèles et permettre l'opérationnalisation. La problématique de l'extension du méta-modèle de la plate-forme nous a amené à étudier la composition de modèles et comment elle peut s'appliquer à notre proposition. Il en est de même pour la transformation de modèle et la mise en conformité des scénarios en vue de leurs opérationnalisations.

3.2 MÉTHODOLOGIE

La méthodologie de recherche (illustrée dans la figure 23) que nous avons appliquée durant cette thèse a suivi une approche en Y.

Dans un premier temps, et afin d'appréhender le contexte de nos travaux, nous nous sommes familiarisés avec la plate-forme cible, à savoir **Moodle**. Afin de comprendre les pratiques et les problématiques des enseignants utilisateurs de cette plate-forme il apparaissait nécessaire de comprendre son fonctionnement et l'étendu de ses fonctionnalités. Nous avons pu par la suite étudier les travaux pré-existants produits dans le contexte du projet GraphIT et dont dépendraient les travaux présentés dans ce manuscrit. Il était nécessaire d'étudier le **métier de conception de la plate-forme**, à la fois comme artefact, le méta-modèle, et comment il avait été construit. Ce méta-modèle ayant servi au développement d'une **API d'import** de scénario pédagogique sous Moodle, nous avons également étudié son fonctionnement. En parallèle nous avons initié une **étude de l'existant en termes de VIDL** (et plus largement langages de modélisation pédagogique y compris non visuels) et entamé un recueil des **besoins et pratiques** des enseignants-concepteurs concernant la scénarisation pédagogique en lien avec l'utilisation d'une plate-forme de formation en ligne (ou LMS). Cette étude sera par la suite étayée par une enquête à plus large échelle initiée par le projet. Cette première étape a permis d'établir un ensemble de **besoins et d'exigences concernant tant le langage de scénarisation que l'éditeur associé**. Un des besoins principaux concernait l'intégration d'éléments pédagogiques abstraits au langage. Ces éléments reposant sur des fonctionnalités de la plate-forme une fois mis en œuvre, il apparaissait nécessaire d'établir des **correspondances** entre ces deux niveaux d'abstraction.

Nous avons également étudié, indépendamment de la scénarisation pédagogique et des plate-formes de formation en ligne, la littérature concernant **l'Ingénierie Dirigée par les Modèles et le *Domain Specific Modeling***. Les conclusions du travail exploratoire précédant cette

thèse, présenté en section 2.3, nous ont amenés à étudier en particulier les aspects **composition et transformation de modèles** ainsi que la **spécification de langage de modélisation et la génération d'éditeurs** à l'aide d'outillages DSM. Pour cela nous nous sommes appuyés sur la littérature, sur une étude pratique des outillages existants, ainsi que sur l'expérience de l'équipe de recherche.

L'étape suivante était alors de croiser les besoins pour le langage de scénarisation, l'éditeur et les correspondances avec notre approche IDM/DSM. Le besoin de correspondances a ainsi amené une étude du **tissage de modèles** et la proposition d'une solution l'exploitant afin de formaliser et opérer les correspondances. Suivent alors les contributions principales que sont les **spécifications** de notre langage de scénarisation pédagogique et son éditeur ainsi que le **développement d'un prototype**.

Enfin, une expérimentation avec des utilisateurs finaux permettra de vérifier la validité de notre approche.

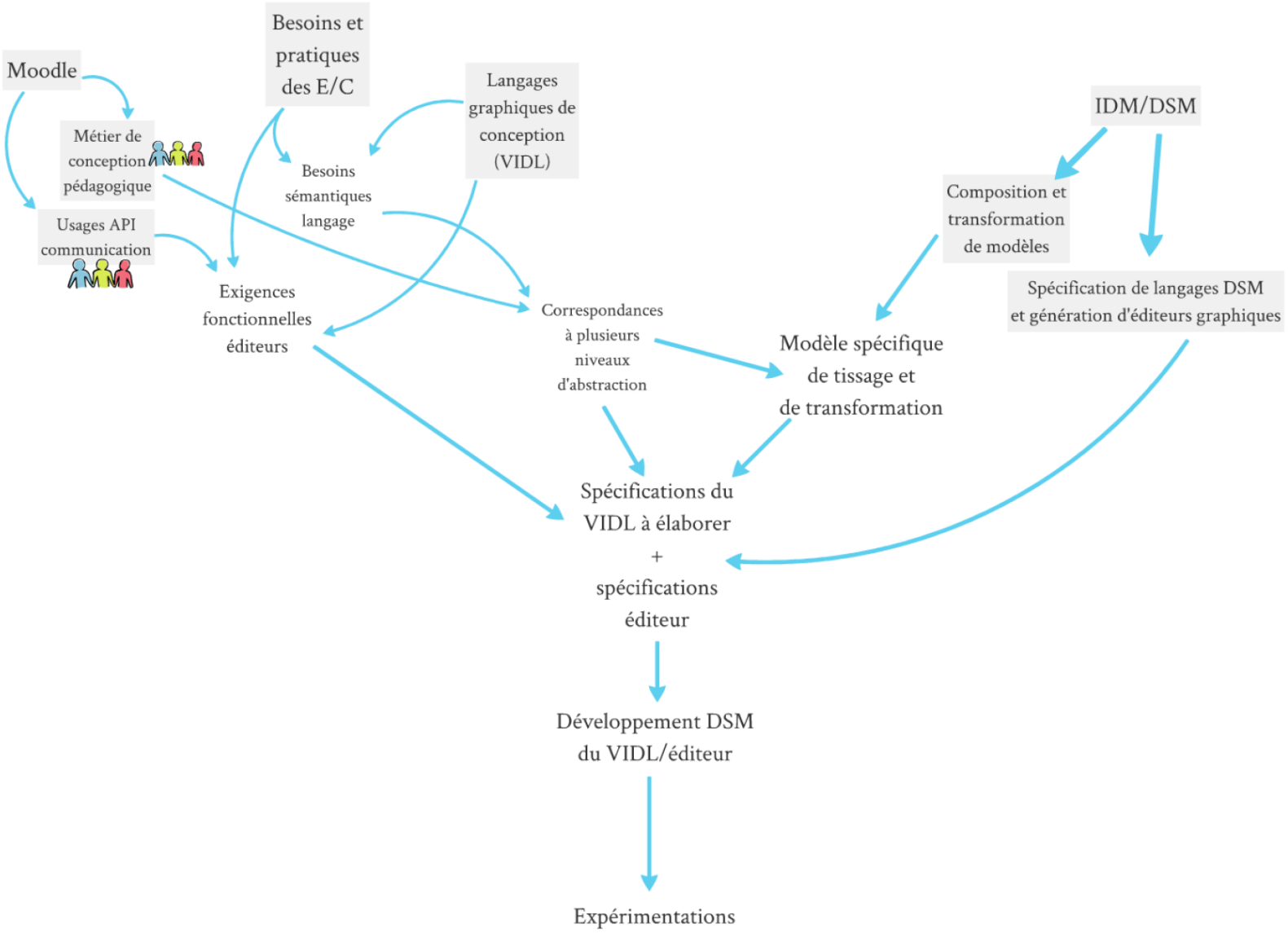

FIGURE 23 – Schéma de la méthodologie de recherche suivant une approche en Y

3.3 APERÇU DES CONTRIBUTIONS ET ORGANISATION DU DOCUMENT

Bien qu'elle dépasse le cadre de cette thèse et contribue au projet GraphiT dans son entièreté, l'enquête présentée dans le chapitre 4 permet d'identifier les besoins et les pratiques de nos utilisateurs finaux, les enseignants-concepteurs. Elle nous a permis de proposer **une méthode d'identification d'activités pédagogiques**, une contribution plus directe de nos travaux présentée dans ce même chapitre.

La problématique de formalisation des correspondances entre les blocs de conception pédagogiques abstraits et leurs implémentations en terme de fonctionnalités de la plate-forme nous a amené à proposer une solution basée sur le tissage de modèles. Cette proposition, décrite plus en détails dans le chapitre 5, consiste en **un méta-modèle de tissage** générique permettant de spécifier un modèle de tissage capturant la sémantique des correspondances. Afin d'exploiter ce modèle, nous proposons une **transformation de haut niveau** spécifique qui permet de générer, à son tour, des transformations de modèles. Ces dernières peuvent être intégrées à un éditeur de scénario pour opérer effectivement les correspondances.

Le chapitre 6 présente les **spécifications** de notre proposition de langage de scénarisation pédagogique. Cette contribution repose sur la spécification de la syntaxe abstraite formalisée dans un méta-modèle ainsi que sur la spécification de la syntaxe concrète via l'outillage Sirius. Les autres constituants d'un langage de modélisation (sémantique et *mappings*) ne sont pas décrits spécifiquement car confondus dans les deux éléments précédents.

La contribution finale de cette thèse consiste en un **prototype d'éditeur** de scénario exploitant ces spécifications. La conception de ce prototype ainsi que la description de ses fonctionnalités sont présentés dans le chapitre 7.

4

RECUEIL ET ANALYSE DES BESOINS

Dans ce chapitre, nous présentons une étude des besoins qui a été menée dans le cadre du projet GraphiT. Dans un premier temps, nous présentons les résultats d'une enquête sous forme de questionnaire en ligne et d'entretiens individuels menés auprès d'enseignants-concepteurs. Dans une seconde section, nous nous appuyons sur d'autres entretiens, à l'échelle locale cette fois, pour identifier des besoins et des exigences (fonctionnelles ou non) quant à la spécification d'un langage de scénarisation pédagogique graphique et le développement d'un éditeur associé. La troisième section est dédiée à une proposition de méthodologie pour l'identification d'activités pédagogiques et la sélection de leurs implémentations.

4.1 ENQUÊTE ET ENTRETIENS

L'enquête menée dans le cadre de GraphiT visait les objectifs suivants :

— vérifier une partie des hypothèses initiales du projet ;
— avoir des retours critiques sur l'approche et les orientations du projet ;
— aider à fixer le périmètre métier de conception pédagogique des plate-formes à considérer ;
— orienter les travaux de recherche sur les besoins sémantiques à capturer pour la spécification de langages de conception graphiques centrés plate-formes.

Et dans dans une moindre mesure :

— lister des usages pédagogiques centrés sur les fonctionnalités et outils de la plate-forme ;
— lister des exigences métiers pour un éditeur graphique (usages, aspects IHM, ergonomie, fonctionnalités...) ;
— lister des exigences en ce qui concerne l'utilisation de l'API d'import/export (export dans l'éditeur, import/export dans l'espace-cours de la plate-forme, processus général d'utilisation de l'API entre l'éditeur externe et la plate-forme...).

Un livrable documentaire [Pod14] est entièrement dédié à cette enquête et à son analyse, nous reviendrons dans ce manuscrit uniquement sur les résultats principaux afin de justifier et contextualiser nos travaux.

4.1.1 *Analyse quantitative exploratoire par questionnaire*

4.1.1.1 *Déroulement*

La première étape de cette enquête fut menée à l'aide d'un questionnaire en ligne comportant un maximum de 23 questions. Certaines questions conditionnant l'apparition d'autres questions, tous les répondants n'ont pas eu le même nombre de questions. La fi-

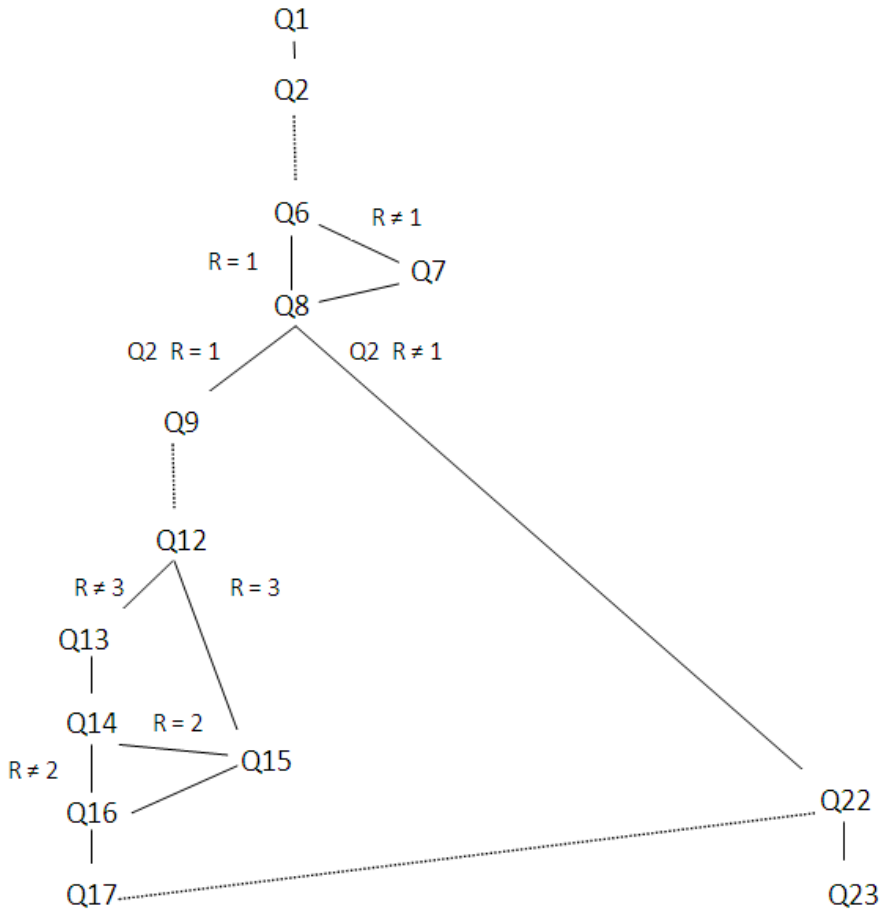

FIGURE 24 – Enchaînement des questions conditionnées pour le questionnaire d'enquête GRAPHIT

gure 24 présente le cheminement du questionnaire en fonction des questions. La question 2 en particulier portait sur les plate-formes de formation utilisées par le répondant. S'il ne choisissait pas Moodle (plusieurs choix étaient possibles) il n'accédait pas aux questions 9 à 22. Le questionnaire fut diffusé auprès d'universités francophones (France, Maroc et Tunisie), d'écoles supérieures et de divers réseaux et associations autour des TICE.

4.1.1.2 *Résultats*

203 réponses furent enregistrées suite à la diffusion du questionnaire. 76% des répondants au questionnaire sont rattachées à une Université (30% de l'Université du Maine). 80% des personnes ayant répondu au questionnaire utilisent la plate-forme de formation Moodle, 16% Claroline et 12% Dokéos. Bien que dans l'absolu le nombre de répondants à l'enquête est limité, cette dernière statistique ren-

force notre hypothèse quant à la popularité de Moodle au sein des établissements d'enseignement supérieur.

Concernant les usages pédagogiques, les enseignants utilisent la plate-forme de formation pour divers usages. 78% des répondants au questionnaire utilisent la plate-forme de formation en complément de leurs interventions en présentiel et la moitié (53%) pour une formation à distance. Les usages des espaces de cours n'étant pas les mêmes, les objectifs pédagogiques sont également variés. 91% des répondants au questionnaire transmettent des documents à leurs étudiants sur l'espace cours (37% d'entre eux n'ont que cet objectif pour leur espace de cours). Ces 37% représentent des enseignants qui n'exploitent pas pleinement la plate-forme à leur disposition, que ce soit par **manque d'expertise quant à l'utilisation de celle-ci ou par inadéquation vis-à-vis de leurs pratiques de conception pédagogique**.

57% des personnes interrogées se sont initiées seules à la plate-forme de formation. 34% par ce seul mode et les autres ont utilisé aussi d'autres méthodes, comme la formation pour 32%, l'accompagnement par un service TICE pour 23% et/ou l'aide de collègues pour 22% des répondants. 44% des répondants au questionnaire se disent moyennement compétents pour l'utilisation de la plate-forme Moodle contre 27% qui se disent compétents voire très compétents (12%). Les personnes se considérant compétentes sur la plate-forme de formation sont 73% à avoir approfondi leurs connaissances seules (58% n'ont utilisé que ce mode). 17% ont utilisé d'autres méthodes comme l'aide de collègues pour 27% des répondants, l'accompagnement par un service TICE pour 25% et/ou la formation pour 21%. On constate que de nombreux répondants se sont formés eux même à utiliser la plate-forme et que proportionnellement, peu d'enseignants ont recours aux formations via les cellules TICE ou autres organismes de formation. Cela renforce l'hypothèse quant à **l'existence de pratiques de conception pédagogique propres à chaque enseignant, voire communauté d'enseignants, grâce aux échanges entre collègues et éventuelles formations locales**.

Concernant l'utilisation de Moodle, 45% des utilisateurs de Moodle pensent que l'interface globale de l'espace cours est claire (voire très claire pour 6%). 58% des utilisateurs de Moodle pensent qu'il est facile de se repérer sur l'espace de cours (dont 22% très facilement). Par contre, 63% des utilisateurs de Moodle pensent que les écrans de paramétrage sont peu compréhensibles, voire pas du tout pour 25% des répondants. Cette statistique renforce l'hypothèse qu'**abstraire les paramétrages des outils de la plate-forme via un outil externe peut permettre aux enseignants d'exploiter plus facilement les fonctionnalités de la plate-forme**. Si l'on regarde plus en détail l'utilisation des fonctionnalités : une majorité des répondants utilisent les fonctionnalités « simples » comme le déplacement gauche/droite (64%) et la visibilité/non visibilité (84%). La moitié des répondants note les

productions de leurs étudiants sur l'espace moodle et/ou crée des groupes.

Plus de la moitié (62%) des répondants dit utiliser la fonctionnalité « restreindre la disponibilité », mais ils ne sont que 34% à utiliser le suivi d'achèvement. Sur l'ensemble des fonctionnalités proposées par Moodle, il y en a 15 sur 22 qui ne sont pas connues par 50% (ou plus) des répondants à la question. Par contre, pour les 7 autres fonctionnalités, 50% (ou plus) des répondants à la question les utilisent régulièrement et parfois pour différents usages pédagogiques. Celles ci sont celles qui sont souvent considérées comme les plus simples : fichier, étiquette, dossier, page, forum, devoir... Pour les outils de communication, le forum (49%) est majoritairement utilisé face au chat (10%). Pour les outils d'exercices, les devoirs (47%) et tests (37%) sont préférés à Hot Potatoes (15%) et Leçon (19%). Pour les outils collaboratifs, le wiki (23%) est préféré au Journal (8%) ou à l'Atelier (8%). Les outils Gallerie Lightbox, Paquetage SCORM et Référentiel sont très peu utilisés (de 4 à 11%).

Concernant la scénarisation, 77% des utilisateurs de Moodle préparent leur scénario pédagogique (ou du moins en partie pour 39%) avant de l'intégrer sur la plate-forme. 54% des personnes utilisant Moodle ont rencontré des difficultés lors du transfert de leur scénario sur la plate-forme et ont dû le modifier, et 12% n'ont pas réussi à les résoudre. Cela confirme **un besoin des enseignants pour l'assistance à l'opérationnalisation de leurs scénarios**. En complément, une proportion moyenne (presque un tiers des répondants à la question) souhaite être accompagnée sur la conception en amont et/ou à la mise en oeuvre du scénario sur la plate-forme.

Seules les personnes utilisant Moodle et préparant leur scénario en amont, en partie ou en totalité, ont répondu pour les résultats suivants : les répondants utilisent un ou plusieurs modes de construction de leur scénario en amont de l'espace de cours, plus de la moitié des répondants utilisent la représentation mentale (61%), le papier (57%) et 45% s'aident d'un outil informatique. Parmi les personnes utilisant un outil informatique pour concevoir leur scénario, 88% utilisent un logiciel de bureautique, un quart un outil de représentation mentale et un quart un outil dédié à la scénarisation pédagogique. 65% des répondants « bureautique » utilisent le traitement de texte pour concevoir leur scénario. Les personnes utilisant un outil de représentation mentale ont cité des logiciels de carte heuristique tels que X-mind ou Mindmap. Les personnes utilisant un outil dédié à la scénarisation pédagogique ont cité des outils tels que Exelearning, Opale, Motplus et Scenari (plus pour la création de contenu).

4.1.2 *Complément d'analyse qualitative : entretiens individuels*

4.1.2.1 *Déroulement*

Après analyse des réponses aux questionnaires, 47 invitations à un entretien individuel ont été lancées. La sélection s'est faite en fonction de plusieurs critères : utilisation de la plate-forme Moodle, établissement de rattachement, scénarisation en amont et utilisation courante des fonctionnalités de moodle. 20 entretiens ont été réalisés à l'aide de l'outil skype ou du téléphone. Seulement 3 entretiens ont été réalisés en face à face pour des questions de temps et de coût.

L'objectif de l'entretien était de faire expliciter aux enseignants leurs utilisations et besoins sur la plate-forme de formation Moodle à partir des éléments suivants : cadre de la formation, organisation de la formation, scénarisation (comment), activités de Moodle, formations à Moodle (s'il y a lieu), remarques sur Moodle (interface, ergonomie, paramétrage...), remarques sur une proposition d'outil de scénarisation pédagogique graphique tel que définit dans les travaux exploratoires, etc.

4.1.2.2 *Résultats*

Les enseignants reçus en entretien utilisent la plate-forme Moodle quotidiennement. Certains d'entre eux ont également utilisé d'autres plate-formes, telles que Ganesha et Sakaï. Deux d'entres eux ont aussi développé une plate-forme dans leur établissement. Elles ont été arrêtées au profit de Moodle pour des raisons financières.

Les enseignants interrogés utilisent la plate-forme en majorité pour des compléments de présentiel (19 sur 23). Moodle est beaucoup utilisé comme lieu de stockage des documents de cours. Quelques uns d'entre eux présentent l'espace de cours Moodle à leurs étudiants pour qu'ils visualisent où trouver les ressources après le cours. Les avantages qu'ils y accordent sont l'aspect écologique du fait de la distribution moins importante des photocopies, l'aide pour les absents au cours, la remédiation. Des utilisations plus marginales sont aussi exprimées comme l'utilisation de moodle pour créer un Mooc ou pour construire un projet pédagogique et productif par les étudiants.

Les enseignants ont appris à utiliser la plate-forme grâce à plusieurs sources. Ils essayent d'abord seuls (16) puis ils sont aidés de leurs collègues (7) ou de formations proposées par leur établissement (10), voire de conseils individualisés par le service TICE (6). Quasiment la totalité des enseignants a pour objectif de donner accès à des documents pour leurs étudiants. Les enseignants déposent leurs ressources après le cours cela éviterait l'absentéisme au cours selon eux. L'espace Moodle sert également de remédiation pour se préparer à l'examen en déposant des annales corrigés, des révisions, des

synthèses etc. ou avant le cours, en préparation d'un TD par exemple. Plusieurs enseignants utilisent les outils d'exercices et d'évaluations (devoir, tests), surtout pour des grands groupes, afin de proposer des auto-évaluations et des évaluations aux étudiants.

La moitié des enseignants reçus en entretien disent maîtriser (10) ou se sentent experts (2) de Moodle alors que très peu d'entre eux vont au delà du scénario : cours et exercices/évaluations sur leur espace. 2 personnes ont souligné que ce qui est important c'est ce que veut faire l'enseignant ou l'équipe et non la plate-forme en elle-même. Par contre, tous les enseignants n'ont pas forcément d'idées pour concevoir un scénario sur la plate-forme et s'aident des activités proposées par celle-ci pour le construire. 9 enseignants créent leur scénario de cours en amont de l'utilisation de la plate-forme, soit en se basant sur le présentiel (ce qu'ils connaissent) soit en partant d'une problématique pédagogique et en en déduisant des objectifs pédagogiques, des modalités et des ressources.

Pour les personnes se basant sur le présentiel, ils essayent de l'adapter à la plate-forme (7), ce qui n'est pas toujours évident pour 6 enseignants qui ont dû modifier leur scénario et 2 qui n'ont pas réussi à résoudre leurs problèmes et ont abandonné. Un enseignant a indiqué qu'il fallait plus formaliser sur la plate-forme que lors du cours puisque l'étudiant est seul derrière son écran. Pour créer leur scénario les enseignants utilisent plusieurs ressources. Pour les enseignants utilisant un outil informatique, la moitié (12) partent d'un document issu d'un outil de bureautique pour créer leur scénario. Un quart des enseignants interrogés commencent tout simplement par le papier crayon pour formaliser leur scénario et presque la moitié utilise aussi un outil de représentation mentale comme les cartes conceptuelles.

Les enseignants utilisent (de peu à régulièrement) quasiment tous les fonctionnalités visibilité/invisibilité et le déplacement gauche/droite pour organiser/hiérarchiser leur cours. Les activités les plus utilisées, par les enseignants reçus, dans leur espace Moodle sont le devoir (16), le forum (15), le test (12) et le wiki (10), suivies par le sondage et le questionnaire (9). Les ressources dossier et fichier sont utilisées par la majorité des enseignants interrogés, suivies par l'URL (18) et la Page (16).

Certains enseignants sont déçus de la plate-forme et ne trouvent pas qu'elle aide à la collaboration. Ils ont des idées mais ne savent pas les intégrer sur la plate-forme. Une enseignante avait envie de travailler par projet mais elle n'est pas convaincue par l'aspect collaboratif de Moodle. Les enseignants pensent que la plate-forme est plutôt simple d'utilisation dans l'ensemble. Par contre s'ils veulent concevoir quelque chose de plus élaboré il faut prendre le temps. Par rapport aux écrans de paramétrage, ils pensent qu'ils sont nombreux ce qui peut être positif quand ils ont une bonne idée de leur objectif mais leur semble un peu« usine à gaz. De plus ils ne connaissent

pas forcément la signification de certains paramètres. Ils doivent souvent **chercher par eux même pour savoir ce qu'un paramètre implique comme conséquences pour l'étudiant**. Ils utilisent beaucoup **les paramètres par défaut et créent donc des activités relativement simples**.

Les enseignants ont été interrogés sur la pertinence d'un outil de scénarisation conçu selon notre approche, sur le fait qu'il serait graphique et externe à Moodle, ainsi que sur la fréquence d'usage (en amont, en cours de réalisation). Quasiment la totalité des enseignants interrogés pense que l'outil proposé peut être pertinent. Selon eux, il faut que l'outil soit assez souple pour répondre à des objectifs pédagogiques larges. Le fait qu'il soit externe et graphique a séduit la quasi totalité des enseignants. Cela leur semble plus facile que la représentation soit graphique et ils trouvent intéressant que l'outil soit accessible hors connexion pour concevoir leur scénario n'importe où et à n'importe quel moment.

4.1.3 *Conclusions*

Cette enquête a permis de valider le choix de notre plate-forme cible, Moodle. Elle a également permis de valider certaines de nos hypothèses : **les enseignants sous-exploitent les fonctionnalités proposées par la plate-forme, en partie à cause d'interfaces trop complexes** et d'un manque de formation, ce qui rend pertinent notre approche visant à proposer un langage de scénarisation abstrait de la plate-forme qui en masquerait la complexité.

Les enseignants-concepteurs semblent porter un intérêt pour la scénarisation pédagogique et, même si peu utilisent des outils dédiés, les enseignants interviewés ont trouvé pertinente notre approche à base d'éditeur de scénario externe à la plate-forme.

4.2 ENTRETIENS INFORMELS

En marge de l'enquête réalisée dans le cadre du projet GraphiT, nous avons pu discuter, de manière plus informelle, avec des enseignants, enseignants-chercheurs et ingénieurs de la cellule TICE utilisateurs de Moodle à l'Université du Maine. Ces discussions ont été menées à la fois lors d'entretiens individuels et de réunions du Groupe d'Analyse des Pratiques Pédagogiques local.

En termes de besoins fonctionnels pour un outil-auteur graphique dédié à Moodle, les enseignants ont évoqué le besoin de ne pas être contraints dans leur méthode de scénarisation : le fait d'avoir accès à différents niveaux d'abstraction ne doit pas imposer une approche top-down (de la spécification vers l'implémentation). Ainsi, ils souhaitent pouvoir **mixer les concepts de spécification** (des briques pédagogiques abstraites) et ceux **d'implémentation** (les briques de

base issues du métier de la plate-forme comme les outils, ressources, etc.). Un autre besoin identifié était de pouvoir visualiser une implémentation possible (traduite dans le métier de conception de Moodle) d'une brique pédagogique sans avoir à la spécifier eux-mêmes (**implémentation par défaut**), tout en ayant la possibilité de la modifier manuellement (**adaptation de l'implémentation**). Cette approche de conception devrait aider les concepteurs à s'approprier les concepts pédagogiques et à maîtriser leurs traductions en éléments de la plate-forme.

Un autre point soulevé concernait la possibilité de déclarer dans le scénario des informations qui n'ont pas d'implémentation directe sur la plate-forme ou qui ne seront pas visibles par les étudiants : indications pour une session en présentiel ; précisions sur des objectifs pédagogiques ; informations sur les étudiants, précisions sur les activités durant le déroulement de la session d'apprentissage etc. (**contrôle sur la visibilité de certaines informations**). Enfin, un besoin de conception que nous avons identifié est celui de pouvoir séquencer les activités au sein de **structures avancées** (séquences, activités au choix, etc.) pour lesquelles le contenu ne serait dévoilé qu'après la réalisation de l'activité précédente. Cette possibilité est offerte par Moodle dans sa version actuelle, mais nécessite un travail de paramétrage assez complexe (suivi d'achèvement et restriction d'accès) que les enseignants apprécieraient de ne pas avoir à faire manuellement.

A nouveau, la plupart des personnes interrogées, validaient l'idée d'un éditeur de scénario externe spécifique à Moodle et l'utilisation d'un bloc pour importer les scénarios dans l'interface du cours (l'aspect externe permettant de concevoir des scénarios en dehors de la plate-forme, hors-ligne, et l'aspect graphique permettant de mieux visualiser le scénario dans son ensemble lors de la conception). Ils ont approuvé l'approche que nous proposions, en insistant sur l'intérêt de pouvoir manipuler des exemples d'usage d'outils de Moodle dans l'éditeur. Ils ont également mis en avant le besoin d'utiliser un langage/outil de conception qui couvre différents usages pédagogiques sans devenir pour autant trop générique. Certains ont exprimé le besoin de pouvoir continuer la conception avec l'éditeur après l'import afin d'adapter le scénario, même s'ils avaient conscience que la modification de celui-ci à la fois avec l'éditeur et directement sur Moodle risquait de poser des problèmes techniques (comment interpréter les modifications sur la plate-forme dans l'éditeur à un niveau plus abstrait ?).

4.3 MÉTHODE D'IDENTIFICATION DES ACTIVITÉS PÉDAGOGIQUES

En fonction des besoins identifiés durant les entretiens avec des enseignants-concepteurs, nous avons proposé de concevoir notre langage de scénarisation autour du concept d'activité pédagogique. Nous

considérons qu'une **activité pédagogique consiste en une abstraction d'un usage pédagogique spécifique d'une fonctionnalité de la plate-forme**. Ainsi, à une activité pédagogique, nous faisons correspondre une implémentation reposant sur une ou plusieurs fonctionnalités de la plate-forme associée à son paramétrage (les paramètres d'un outil de la plate-forme pouvant permettre des usages différents). Afin d'identifier les outils les plus appropriés à l'implémentation d'une activité pédagogique nous avons suivi une méthode en trois étapes :

1. analyse, pour chacun des outils proposés par Moodle, de ses usages récurrents (méthode bottom-up),
2. identification d'outils permettant des usages identiques (top-down),
3. spécification des critères discriminants permettant la sélection de l'outil le plus adéquat.

Moodle, dans sa version 2.4, propose 7 types de ressources (Livre, Page, Étiquette, Paquetage IMS, Fichier, Dossier et URL) et 13 activités (Forum, Base de données, Glossaire, Devoir, Leçon, Test, Atelier, Paquetage SCORM, Outil externe, Sondage, Consultation, Wiki et Feedback). Après avoir étudié les usages récurrents de chacun de ces outils, nous avons remarqué que certains d'entre eux pouvaient avoir des usages détournés. Par exemple, un forum peut être utilisé pour discuter autour d'une thématique, mais peut également servir de Foire Aux Questions (FAQ) ou permettre aux étudiants de partager des fichiers. Dans un second temps, nous avons identifiés quels outils avaient des usages en commun, par exemple, pour une FAQ il est possible d'utiliser un forum, un wiki, ou un glossaire.

Nous avons pu alors définir des critères discriminants afin d'aider l'enseignant à décider quel outil utiliser lorsque le choix se présente. Il est possible de représenter ces critères dans une matrice de décision qui se construit selon les règles suivantes :

— **R1.** Le nom de l'activité pédagogique est formulé du point de vue de l'étudiant (sauf si elle leur est invisible), exemple : Répondre à un sondage plutôt que Créer un sondage.
— **R2.** Les outils permettant de mettre en œuvre l'activité pédagogique considérée sont représentés sur les colonnes.
— **R3.** Les critères discriminants sont représentés sur les lignes.
— **R4.** Les critères discriminants doivent exprimés, le plus possible, une caractéristique « pédagogique » et doivent être formulés comme des questions fermées. (oui/non)
— **R5.** Les cellules à l'intersection d'un outil et d'un critère doivent contenir toutes les valeurs possibles de critères qui permettent de choisir cet outil.
— **R6.** Un critère discriminant doit permettre de discriminer au moins un outil.

TABLE 5 – Exemple de matrice de décision

Activité pédagogique	**O1**	**O2**	**O3**	**O4**
C1	Oui/Non	Non	Oui/Non	Oui
C2	Oui/Non	Non	Oui/Non	Oui
C3	Non	Non	Non	Oui
C4	Oui/Non	Non	Non	Non
C5	Non	Non	Oui/Non	Non
C6	Oui	Non	Non	Non
C7	Oui	Non	Oui	Non

— **R7.** La matrice est complète si il n'y a pas de combinaisons de critères parfaitement identiques menant à deux outils.

Une matrice incomplète nécessite d'ajouter des critères supplémentaires, jusqu'à satisfaire la règle R7.

Le tableau ci-dessous présente un exemple pour l'activité Répondre à un sondage, pour laquelle quatre outils sont disponibles : test (O1), sondage (O2), feedback (O3) et consultation (O4). Nous proposons 7 critères discriminants :

— (C1) Il y a-t-il plusieurs questions ?
— (C2) Il y a-t-il uniquement des questions à choix multiples ?
— (C3) Voulez vous utiliser des questionnaires prédéfinis ?
— (C4) Est-ce en temps limité ?
— (C5) Est-ce anonyme ?
— (C6) Est-ce noté ?
— (C7) Il y a-t-il un feedback après la validation du sondage ?

Selon la matrice du tableau 1, l'outil consultation (O4) par exemple, permet de sonder sur plusieurs questions, proposant des choix multiples et des questionnaires prédéfinis. Pour cet outil, il n'est pas possible d'imposer une limite de temps pour répondre, ni d'anonymiser les réponses ou de les noter. On ne peut pas non plus donner de feedback à l'apprenant. Cette matrice doit également être complétée par des informations sur les paramètres de l'outil sélectionné, qu'ils soient généraux (peu importe les valeurs des critères), ou contextuels (la valeur du paramètre dépend d'une réponse à un des critères).

4.4 BILAN

Dans le cadre du projet GraphiT, nous avons mené une enquête auprès d'enseignants concepteurs et donc potentiels utilisateurs de nos propositions. Cette enquête a consisté en un questionnaire en ligne, dont nous avons présenté les résultats, puis d'entretiens individuels réalisés par une ingénieure pédagogique auprès d'enseignants sélectionnés parmi les répondants au questionnaire. Ces entretiens

ont permis d'approfondir les réponses des enseignants au questionnaire et de valider nos hypothèses : choix de la plate-forme cible, sous-exploitation du LMS etc.

D'autres entretiens, individuels et en groupes, ont permis d'identifier des besoins en conception pédagogique et plus particulièrement des exigences concernant un potentiel langage de scénarisation pédagogique graphique outillé. Ces exigences portent à la fois sur l'expressivité et la structure du langage, mais également sur les fonctionnalités de l'éditeur associé et serviront par la suite à établir les spécifications de ces deux contributions.

Dans un second temps, sur la base des besoins exprimés par les enseignants, nous avons proposé de placer le concept d'activité pédagogique, au centre de notre proposition de langage de scénarisation. Une activité pédagogique est définie en lien avec son implémentation en termes d'outils de la plate-forme (fonctionnalité, ressource ou autres). Afin d'identifier l'outil le plus à même de mettre en œuvre une activité pédagogique donnée, nous avons proposé une méthodologie à base de critères discriminants représentés dans une matrice (un tableau).

Cette méthode peut également servir de support à la discussion avec un expert pédagogique représentant la communauté d'enseignants-concepteurs visés. La représentation sous forme de matrice, permet de synthétiser les informations relatives à l'implémentation d'une activité pédagogique, mais ne constitue pas un modèle exploitable. Si nous souhaitons mettre en oeuvre les correspondances entre outils de la plate-forme et activités pédagogiques, il nécessaire de les formaliser. Selon notre approche IDM/DSM, nous proposons d'utiliser un modèle.

5

FORMALISATION DES CORRESPONDANCES

Nous proposons de concevoir un langage de scénarisation pédagogique centré sur le concept d'activité pédagogique. Cette brique de conception du scénario doit, selon notre approche centrée plate-forme, trouver une équivalence (ou correspondance) en termes d'implémentation sur celle-ci. L'identification des correspondances entre activité pédagogique et outils de la plate-forme a été présentée dans le chapitre précédent. Selon le cadre IDM/DSM que nous appliquons, les différents concepts constituant le langage de scénarisation sont formalisés dans un méta-modèle. Dans la section 4.3 nous avons présenté une méthode d'identification des activités pédagogiques associées à leurs implémentations en terme d'outils (fonctionnalités, ressources, etc.) de la plate-forme. Cette méthode propose de constituer une matrice de décision dont le but est de guider l'enseignant-concepteur dans le choix de l'implémentation la plus adaptée à son besoin pédagogique, en fonction de critères. Dans ce chapitre, nous nous intéressons en particulier à la formalisation et à l'exécution de ces correspondances entre activités pédagogiques et outils de la plate-forme.

5.1 CORRESPONDANCES ENTRE ACTIVITÉS PÉDAGOGIQUES ET OUTILS

Dans notre cas, les concepts à mettre en relation sont définis dans un méta-modèle : à chaque activité pédagogique ainsi qu'à chaque outil correspond une classe dans notre méta-modèle. L'approche la plus simple pour modéliser une correspondance serait d'établir directement une relation (composition ou héritage) entre la classe d'une activité pédagogique et celle de son implémentation dans le méta-modèle du langage. Néanmoins, cette solution ne permet pas de capturer la complexité de la sélection d'une implémentation d'activité telle que présentée dans le chapitre précédent. Pour rappel, une même activité pédagogique peut selon la valeur de certaines propriétés (appelés discriminants) avoir des implémentations différentes. La figure 25 illustre cette situation avec l'exemple de l'activité *répondre à un sondage* : C1..7 sont des critères de sélection, le caractère ¬ indique que la valeur attendue pour ce critère est négative (le . correspond à un ET logique). Ce type de correspondances, *dynamiques*, n'est pas modélisable directement au sein du méta-modèle du langage. Nous proposons alors, de capturer la dynamique de ces correspondances dans un modèle distinct.

5.1.1 *Un besoin de tissage de modèles*

Notre proposition s'appuie sur le fait de modéliser une correspondance comme un ensemble de liens entre une activité pédagogique et ses potentielles implémentations. En Ingénierie Dirigée par les Mo-

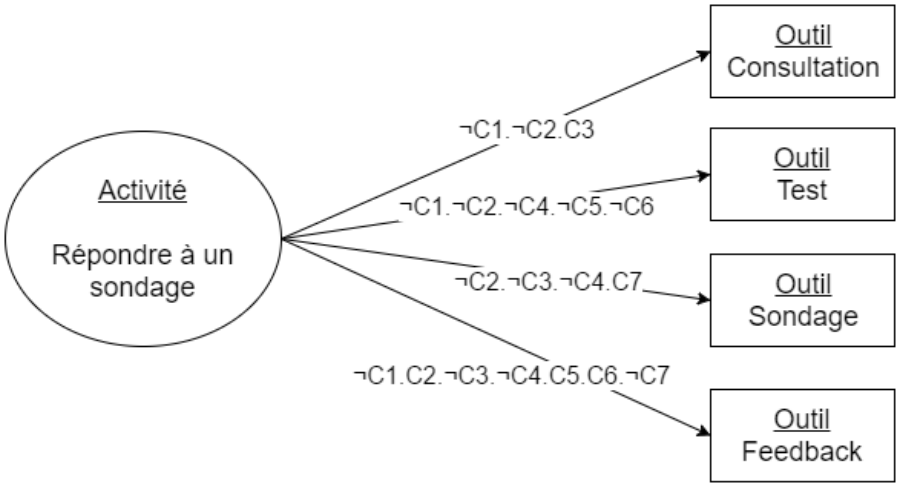

FIGURE 25 – Exemple de correspondances pour l'activité *répondre à un sondage*

dèles, la pratique qui consiste à définir un modèle capturant les liens entre les éléments d'autres modèles est appelée *tissage de modèles*. Dans notre cas, les éléments à tisser sont des classes définies dans des méta-modèles, on peut considérer qu'il s'agit de tissage de méta-modèles.

Le modèle capturant les liens entre les deux modèles tissés est appelé *modèle de tissage* et doit à son tour être conforme à un méta-modèle (de tissage). Selon l'outillage employé, le méta-modèle de tissage peut être pré-défini, c'est le cas pour AMW (*Atlas Model Weaver*) [DV09] qui propose un méta-modèle de tissage extensible. Cet environnement n'étant plus maintenu et posant des problèmes de compatibilité avec les versions récentes d'autres librairies, nous avons choisi de ne pas l'utiliser. Nous avons donc défini notre propre méta-modèle de tissage, répondant à nos besoins quant à la modélisation de correspondances.

5.1.2 *Notre méta-modèle de tissage*

Ce méta-modèle est illustré dans la figure 26. Le concept racine de ce méta-modèle est le modèle de tissage (*WeavingModel*) qui, par composition, contient les différentes correspondances (*Binding*). Une correspondance doit être créée pour chaque type d'élément source (activité pédagogique) et peut contenir plusieurs cibles (*Target*, outils de la plate-forme). Chaque cible peut être sujette à une ou plusieurs *Condition*s, elles-mêmes combinables par des opérateurs logiques. Ces conditions permettent de tester la valeur des attributs (ou critères) *discriminant* le choix de l'implémentation d'une activité. Pour chaque élément cible, il est possible de définir la valeur de ses attributs (*SetAttr*) lorsqu'il sera instancié dans le scénario.

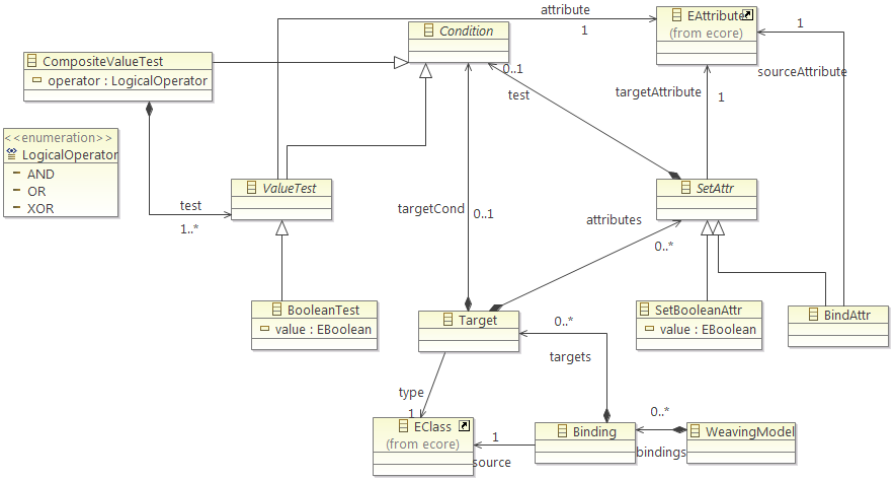

FIGURE 26 – Méta-modèle de tissage (tronqué pour illustration)

5.1.3 *Mise en œuvre outillée avec Epsilon*

La figure 27 présente un modèle de tissage tel que nous suggérons son édition, via l'éditeur à trois panneaux générique ModeLink proposé par le projet Eclipse Epsilon [Pai+09]. Le panneau central permet d'éditer le modèle de tissage, le panneau de gauche permet de sélectionner les activités pédagogiques dans le méta-modèle du langage de scénarisation et le panneau de droite permet de sélectionner les outils de la plate-forme dans le méta-modèle de celle-ci. La figure 28 présente, plus en détail, un exemple de modèle de tissage. Dans cet exemple, une seule correspondance est définie, pour l'activité *AnswerAPoll*, qui peut être implémentée à l'aide de quatre outils différents (*Quiz*, *Choice*, *Feedback* et *Survey*). Par exemple, l'outil *Quiz* sera instancié si l'activité a les propriétés *anonymous* et *pre_populated* à faux et la propriété *graded* à vrai. Dans le cas où l'outil Quiz est instancié, l'attribut *name* sera valué à *MyQuiz*.

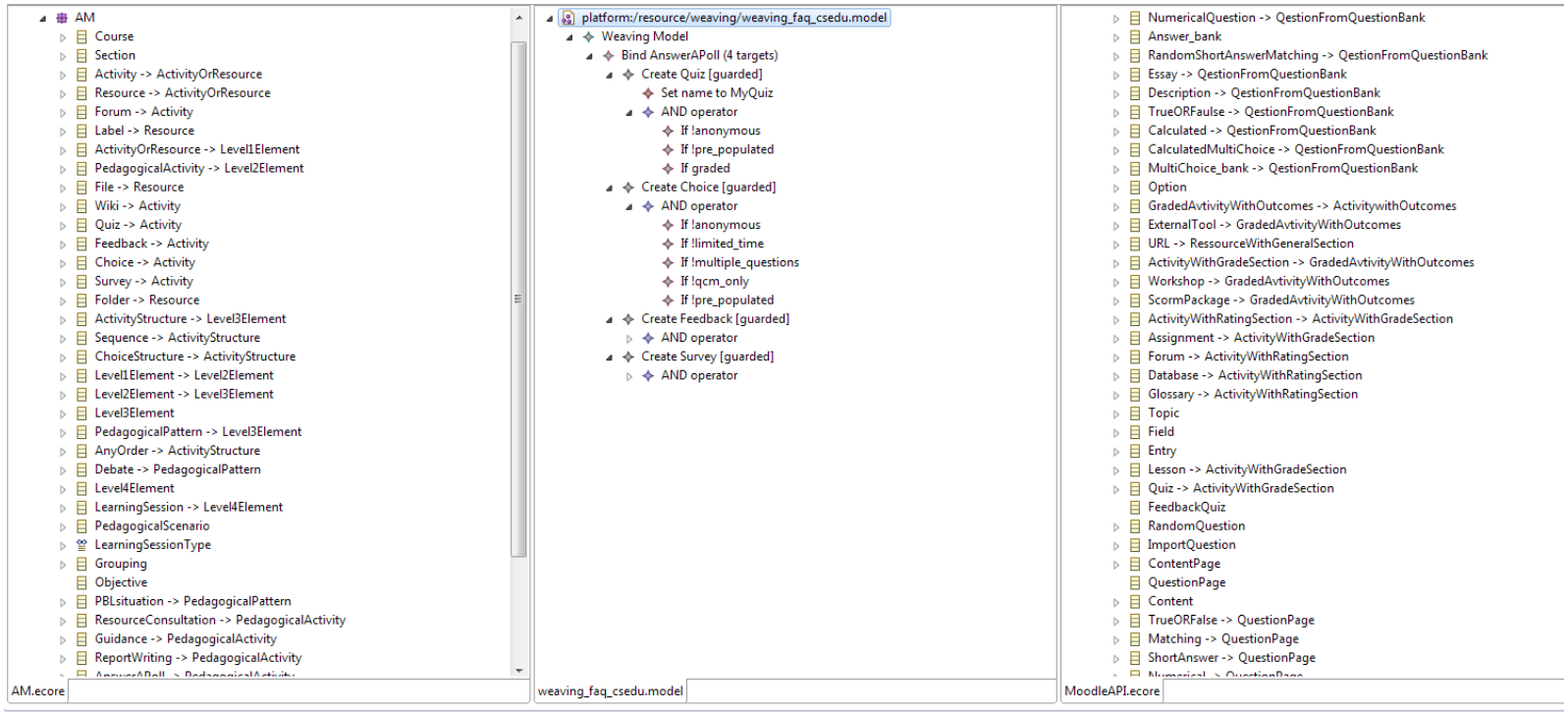

FIGURE 27 – Capture d'écran de l'éditeur 3 panneaux ModeLink lors de la définition d'un modèle de tissage

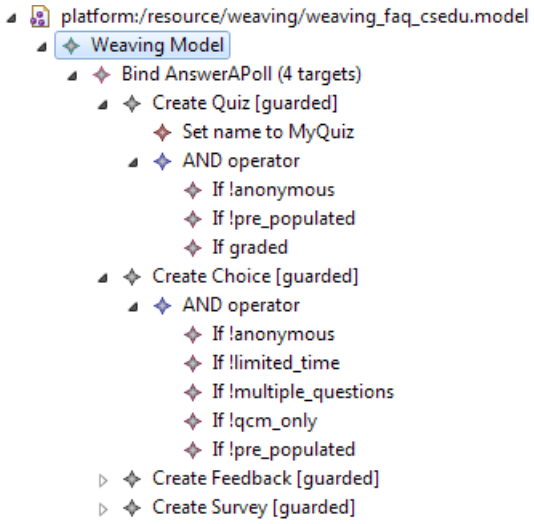

FIGURE 28 – Exemple de modèle de tissage

5.2 APPLICATION DU TISSAGE

Formaliser les correspondances dans un modèle ne suffit pas à les rendre opérationnelles. Notre objectif est de permettre à l'enseignant-concepteur d'obtenir l'implémentation (par défaut, car elle est ensuite modifiable) de l'activité pédagogique, au sein de l'éditeur de scénario, qu'il a souhaité instancier en fonction des propriétés de celle-ci. Pour cela, nous proposons de transformer le modèle du scénario pendant son édition pour y ajouter l'implémentation d'une activité pédagogique. Cette transformation de modèle dépend directement du modèle des correspondances (modèle de tissage). Une approche possible serait de transcrire manuellement les correspondances formalisées dans un langage de transformation de modèle mais cela s'avérerait coûteux en terme de développement et n'exploiterait pas efficacement le modèle de tissage.

De façon similaire au fonctionnement de l'outillage AMW, nous proposons de générer automatiquement les transformations de modèles opérant les correspondances à partir du modèle de tissage. Le principe est de concevoir une transformation de haut-niveau (*High Order Transformation*) qui prendra en entrée le modèle de tissage et qui produira en sortie une ou plusieurs transformations de modèles. La figure 29 synthétise le positionnement des différents modèles impliqués. Il est à noter que le modèle de tissage référence les méta-modèle d'entrée et de sortie et que la transformation de haut niveau (HOT) s'appuie également sur ceux-ci, bien que ce ne soit pas représenté sur la figure (par soucis de lisibilité). La programmation d'une transformation de haut niveau à l'aide d'un langage de transformation tel qu'ATL dans le cas d'AMW, peut s'avérer complexe. Nous proposons d'utiliser, comme dans les approches classiques de généra-

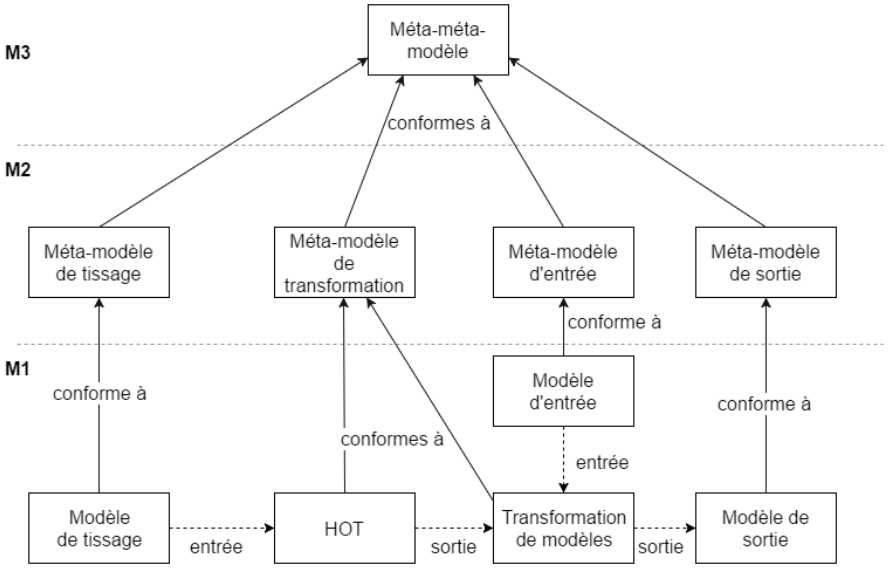

FIGURE 29 – Fonctionnement général du tissage de modèles tel qu'opéré par AMW

tion de code à partir de modèles, une transformation modèle-à-texte. Cette transformation permet, à l'aide d'un langage dédié à la manipulation de modèles, de définir un *template* qui contient à la fois des parties fixes(la structure du texte produit : dans notre cas la structure d'une transformation de modèle) et des parties variables : le code permettant de trouver les informations dans le modèle de correspondances. La figure 30 présente un extrait de code de notre implémentation d'une HOT [1] à l'aide du langage EGL [2] du projet Eclipse Epsilon[Pai+09], la transformation produite est en EOL [3] (autre langage d'Epsilon). La figure 31 présente une transformation de modèle générée par la transformation de haut niveau (HOT).

La figure 32 présente notre application du tissage de modèles. Les langages du projet Epsilon pour les transformations modèle-à-modèle ou modèle-à-texte (EOL et EGL) ne sont pas considérés comme des modèles conformes à un méta-modèle. La transformation produite par notre équivalent de HOT, permet d'enrichir le scénario pédagogique avec les informations concernant l'implémentation des activités pédagogiques, c'est pourquoi le modèle de scénario est à la fois l'entrée et la sortie.

1. *High Order Transformation*
2. *Epsilon Generation Language*
3. *Epsilon Object Language*

```
var sourceVarName:String = sourceClassName.firstToLowerCase();
%]


operation EObject addMapping(element:EObject) {
    if(self.hasProperty("mapedL1Elements")) {
        self.mapedL1Elements.add(element);
    }
    else if(self.hasProperty("mapedL2Elements")) {
        self.mapedL2Elements.add(element);
    }
    else if(self.hasProperty("mapedL3Elements")){
        self.mapedL3Elements.add(element);
    }
}

operation EObject bind[%=sourceClassName%]():EObject {
    var target = [%=sourceClassName%].all.selectOne(it|it.id == self.id);
    [% for(targetElement:Target in binding.targets) {
        var cond:Boolean = targetElement.targetCond.isDefined();
        if(cond) {%]
            if([%=targetElement.targetCond.formatCondition()%]){
        [%}
            var targetVarName = targetElement.instanciate();%]
                self.addMapping([%=targetVarName%]);
        [%if(cond){%]
            }
        [%}
    }%]
}

[% operation Target instanciate() : String{
    var targetClassName:String = self.type.getName();
    var targetVarName:String = targetClassName.firstToLowerCase();
%]
    var [%=targetVarName%] = new source!`[%=targetClassName%]`;
    [%//Set attributes
    for(setAttr:SetAttr in self.attributes) {
        var cond:Boolean = setAttr.test.isDefined();
        if(cond){%]
            if([%=setAttr.test.formatCondition()%]) {
        [%}%]
```

FIGURE 30 – Extrait de code d'une transformation modèle-à-texte en EGL

```
    }
    else if(self.hasProperty("mapedL2Elements")) {
        return self.mapedL2Elements;
    }
    else if(self.hasProperty("mapedL3Elements")){
        return self.mapedL3Elements;
    }
}

operation EObject setCompo(mapping) {
    if(self.hasProperty("mapedL1Elements")) {
        self.mapedL1Elements.add(mapping);
    }
    else if(self.hasProperty("mapedL2Elements")) {
        self.mapedL2Elements.add(mapping);
    }
    else if(self.hasProperty("mapedL3Elements")){
        self.mapedL3Elements.add(mapping);
    }
}

operation EObject bindBrainstorming():Sequence {
    if(not (self.getCompo().size() > 0)) {
        var label = new `Label`;
        label.labelText = "Constituer des groupes de 4";
        label.name = label.labelText;

        var label2 = new `Label`;
        label2.labelText = "Discuter sur le sujet suivant: ...";
        label2.name = label2.labelText;


        var forum = new `Forum`;
        forum.name = "Forum de discussion - Brainstorming";


        label.nextE = label2;
        self.setCompo(label);
        label2.nextE = forum;
        self.setCompo(label2);
        self.setCompo(forum);
    }
    self.getCompo().asSequence().println();
    return self.getCompo().asSequence();
}
```

FIGURE 31 – Extrait d'une transformation de modèles générée en EOL

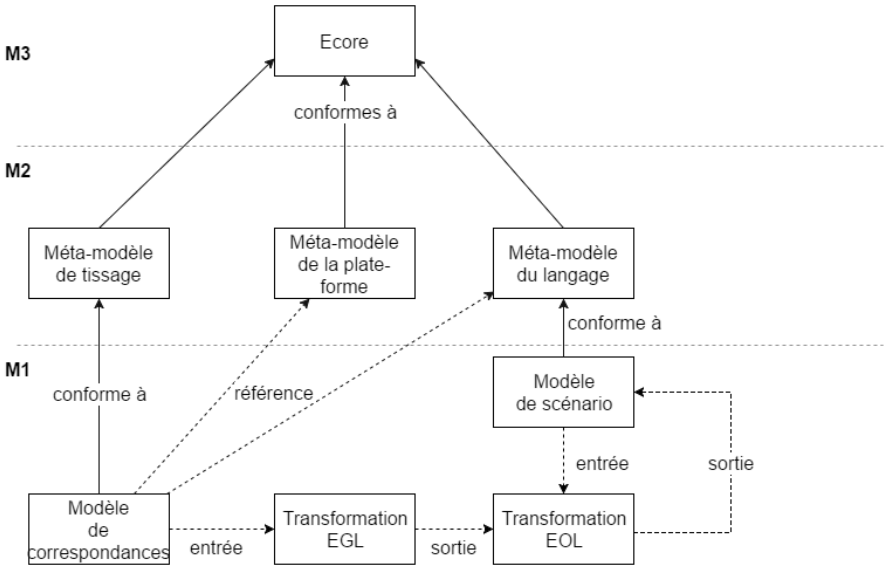

FIGURE 32 – Application du tissage de modèles

5.3 ÉLÉMENTS NON TRAITÉS PAR LE TISSAGE

Le mécanisme à base de tissage de modèles que nous proposons ne traite que les activités pédagogiques. Ces éléments, bien que centraux dans notre langage, ne suffisent pas à constituer un scénario complet. Il est nécessaire de proposer des concepts qui permettent de structurer le scénario : ces éléments se traduiront sous forme de cours, sections, indentations etc. sur la plate-forme d'opérationnalisation. En particulier les éléments cours et sections sont obligatoires pour l'import d'un scénario sur Moodle, leur correspondance doit donc être fixe (non modifiable) et sera prise en charge par la transformation *d'export* abordée dans la section 2.4.1.2. Cette transformation, contrairement à celles produites par le tissage, sera exécuté après l'édition du scénario par l'enseignant-concepteur.

5.4 BILAN

Dans ce chapitre, nous avons proposé une solution à base de tissage de modèles afin de formaliser les correspondances entre activité pédagogiques et outils de la plate-forme utiles à leurs implémentations. Selon notre approche, la méthode d'identification présentée dans la section 4.3, permet la discussion entre experts pédagogiques (enseignants, ingénieurs pédagogiques etc.) et une première « formalisation » de ces correspondances. Dans un second temps, un expert en modélisation peut, sur la base de la matrice produite dans la phase précédente, utiliser l'éditeur de modèle de tissage (voir figure 27) afin de modéliser les correspondances. Ce modèle permet alors, via une transformation de haut niveau, de produire des transformations de modèles qui seront intégrées à l'éditeur de scénario. Cette approche permet de spécifier l'implémentation de toutes les activités du même type, mais ne répond pas au besoin de personnalisation de l'implémentation **d'une instance** en particulier d'activité pédagogique. Les chapitres suivants présentent les spécifications du langage et de l'outil de scénarisation, ils décriront notamment quelles solutions, en terme de méta-modélisation et de conception de l'éditeur, nous avons proposées afin de répondre à ce besoin.

6

SPÉCIFICATIONS DU LANGAGE DE SCÉNARISATION

Dans ce chapitre nous présentons les syntaxes abstraites et concrètes de notre langage de scénarisation pédagogique graphique centré plate-forme. La syntaxe abstraite est formalisée dans un méta-modèle qui vous est présenté ci-après. Ce méta-modèle est conçu selon une architecture à quatre niveaux : le premier étant directement le méta-modèle de la plate-forme, les niveaux supérieurs ajoutent chacun des concepts au langage par abstraction. Notre proposition étant un langage de scénarisation pédagogique **graphique**, nous présenterons également la notation visuelle (syntaxe concrète) associée aux concepts du méta-modèle du langage.

6.1 SYNTAXE ABSTRAITE : MÉTA-MODÈLE

Le méta-modèle est illustré dans les figures 33, 34, 35 et 36. Il inclut à la fois le méta-modèle de la plate-forme (tronqué sur les illustrations, pour conserver leurs lisibilités) et un ensemble d'extensions qui constituent les différentes abstractions que nous avons ajoutées au langage. Pour la clarté de l'explication, les figures sont des extraits du méta-modèle complet qui comprend davantage de concepts et d'attributs.

6.1.1 *Syntaxe abstraite : niveau 1*

Ce méta-modèle a été conçu en quatre niveaux, le premier niveau (figure 33) correspond directement au méta-modèle de la plate-forme. Les classes *Level2Element* et *PedagogicalActivity* ne font pas partie de ce niveau, mais permettent de situer cette illustration dans le méta-modèle complet. Le seul ajout notable est la classe abstraite *Level1Element* qui sert de racine à la hiérarchie des classes de ce niveau et permet la composition avec le reste de l'extension du méta-modèle de la plate-forme (les autres niveaux : voir section 2.4.2.2). Conserver un lien (par la relation de composition entre *Level1Element* et *PedagogicalActivity*) entre un élément du niveau supérieur et son implémentation permet de répondre au besoin de **visualisation et de personnalisation des implémentations** par l'utilisateur.

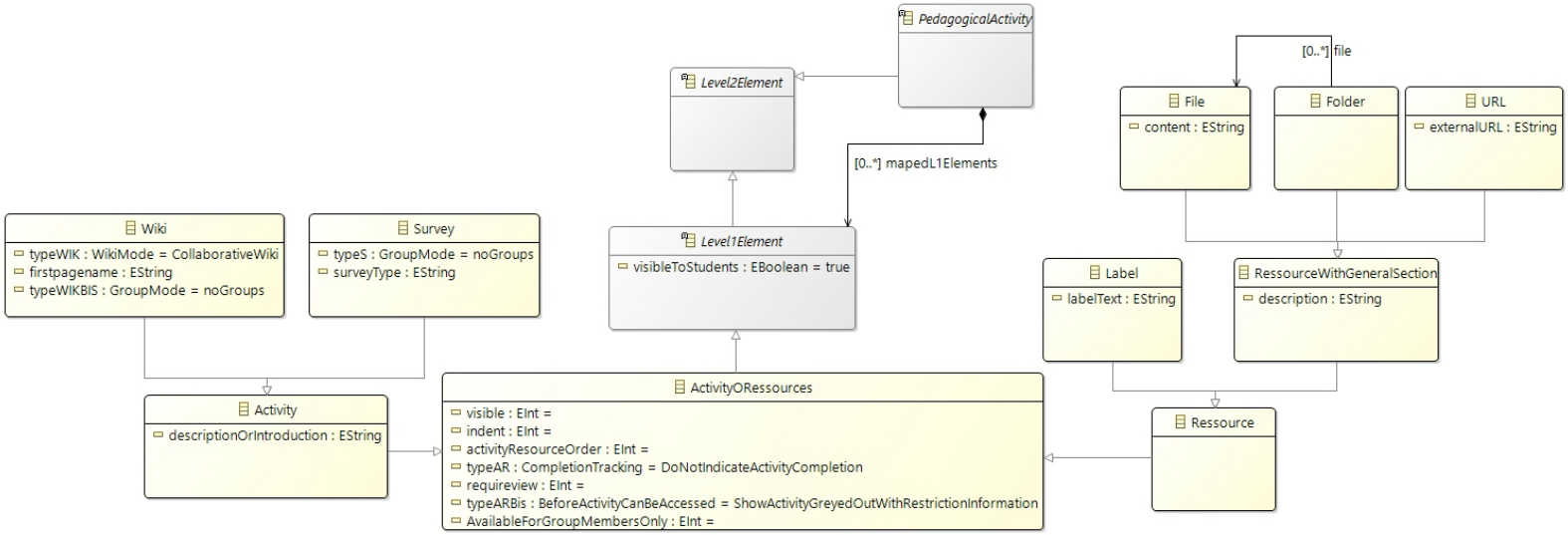

FIGURE 33 – Niveau 1 de la syntaxe abstraite

6.1.2 *Syntaxe abstraite : niveau 2*

Le second niveau (figure 34) constitue notre principale abstraction : les activités pédagogiques. De nouveau, une classe abstraite (donc non instanciable dans le modèle), *Level2Element*, sert de point de départ à la hiérarchie de classes. Deux classes héritent de celle-ci : *TutorActivity* et *PedagogicalActivity*. La première rassemble (par héritage) les concepts qui définissent une activité propre au tuteur de la session d'apprentissage et qui n'apparaîtra pas sur la plate-forme. La seconde rassemble les activités pédagogiques à proprement parler : *WriteAReport*, *ResourceConsultation*, *Assessment*... Le concept de *TeacherNote* est équivalent à *TutorActivity* sauf qu'il sera opérationnalisé mais non-visible par les apprenants. Le concept de *Guidance* correspond à une indication textuelle donnée aux apprenants : il ne s'agit pas directement d'une activité à réaliser mais plutôt d'une consigne. Certaines activités, *WriteAReport* par exemple, possèdent des attributs qui correspondent aux critères discriminants tels que présentés dans la section 4.3, ils seront utiles à l'exécution des correspondances comme décrit dans le chapitre 5.

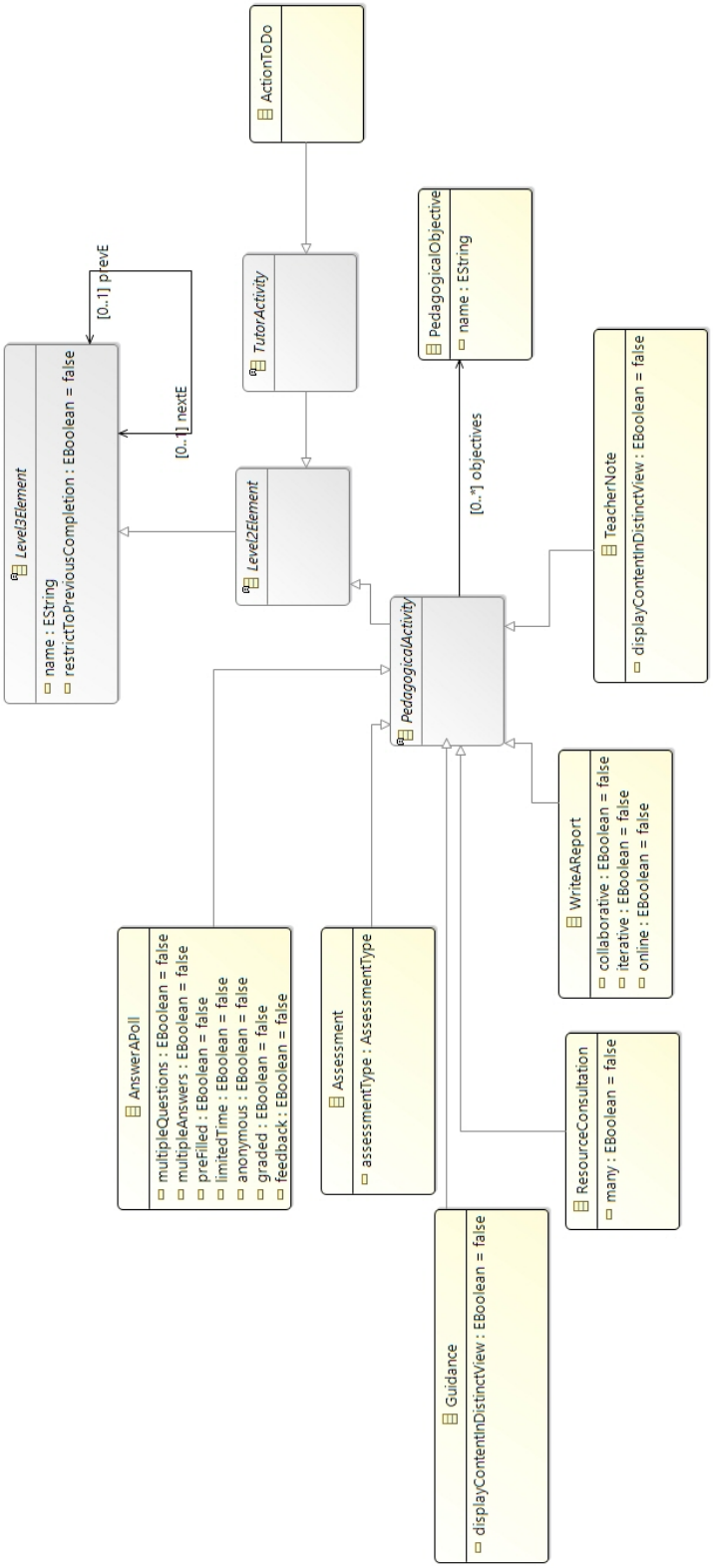

FIGURE 34 – Niveau 2 de la syntaxe abstraite

6.1.3 *Syntaxe abstraite : niveau 3*

Le troisième niveau (figure 35) contient les structures d'activités : *ChoiceStructure*, *Seqence*[1] et *AnyOrder*. La structure *AnyOrder* est la structure la plus simple : l'apprenant doit réaliser les activités contenues dans cette structure dans l'ordre qu'il souhaite. La *ChoiceStructure* indique à l'apprenant de réaliser une activité parmi celles proposées. La *Seqence* oblige l'étudiant à réaliser l'ensemble des activités qu'elle contient dans l'ordre indiqué, l'attribut *restrictionStrategy* permet de choisir si les activités restantes sont visibles mais non accessibles ou complètement invisibles (du point de vue de l'apprenant). L'implémentation des structures exploitera les mécanismes de suivi d'achèvement et de restriction d'accès proposés par Moodle.

Une *ActivityStructure* est composée de *Level3Element*, qui correspondent au contenu de la structure. Afin de répondre au besoin de **mixer les concepts de spécification et d'implémentation**, nous avons établi une relation d'héritage entre *Level2Element* et *Level3Element*, et *Level1Element* et *Level2Element*. Ainsi, il est possible dans une structure d'activités de mixer des éléments des niveaux 1 et 2 (activités pédagogiques et outils de la plate-forme). Ces relations d'héritages sont visibles sur les figures 34 et 33.

6.1.4 *Syntaxe abstraite : niveau 4*

Le quatrième et dernier niveau (figure 36) contient uniquement le concept de *LearningSession* qui sert d'élément englobant une session d'apprentissage et qui sera implémenté à l'aide de sections dans Moodle. Les *LearningSession* peuvent avoir un type, une durée, être visible ou non par les apprenants et peuvent être opérationnalisées ou non (par héritage). Une session contient des éléments de type *Level3Element* ce qui, par héritage, correspond à n'importe quel élément des niveaux inférieurs. Ainsi, l'enseignant peut ignorer ou mélanger les différents niveaux d'abstraction que nous proposons.

1. Il ne s'agit pas d'une erreur orthographique, les classes sont nommées de manière à ne pas créer de conflit avec des mots réservés des langages que nous utilisons et d'autres classes du méta-modèle de la plate-forme

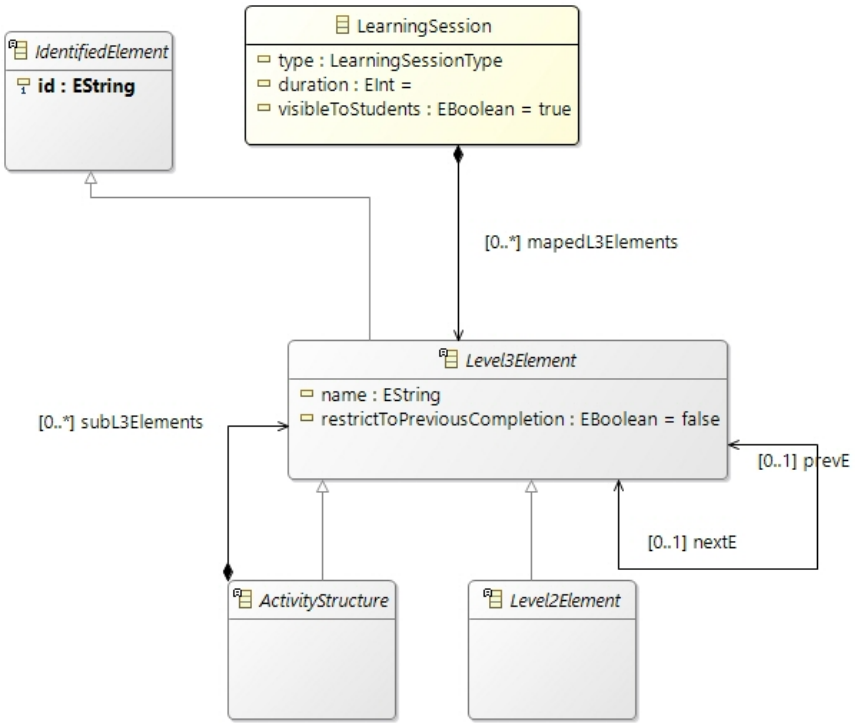

FIGURE 35 – Niveau 3 de la syntaxe abstraite

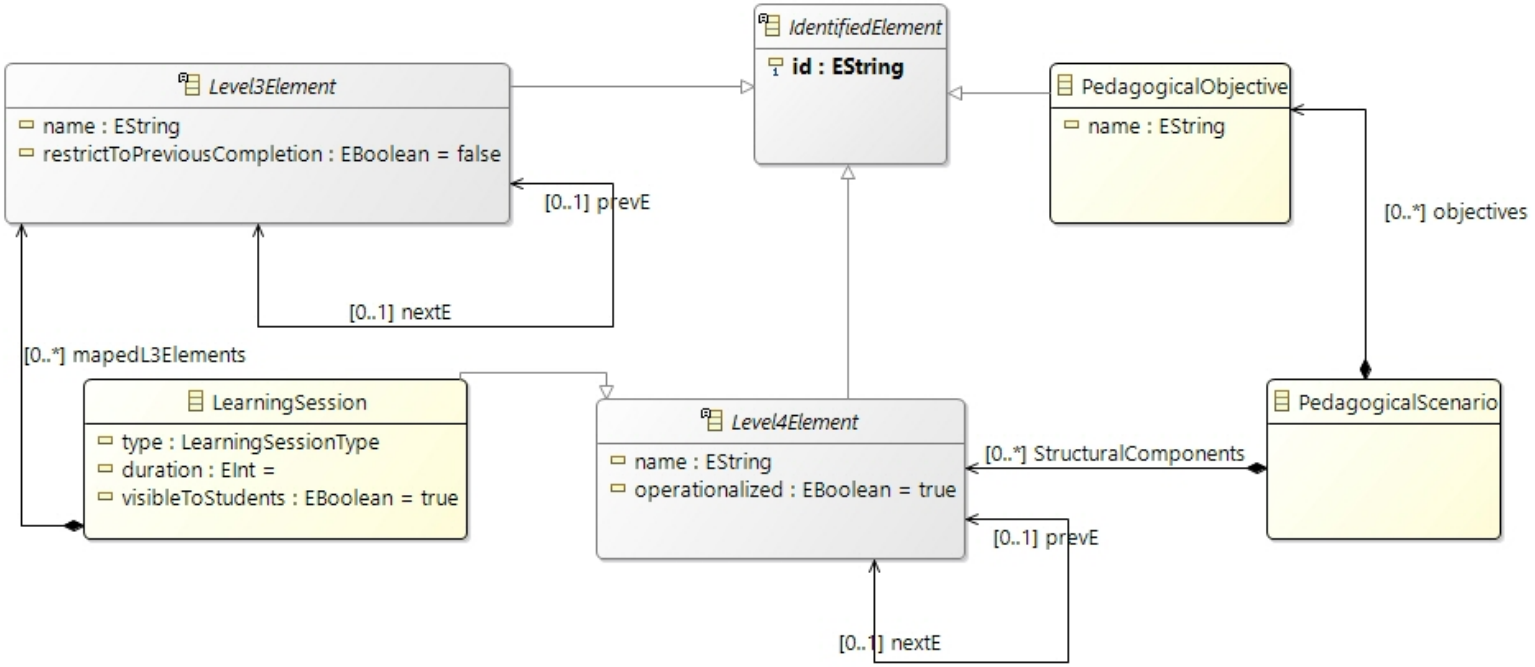

FIGURE 36 – Niveau 4 de la syntaxe abstraite

L'élément racine d'un scénario est le *PedagogicalScenario* qui contiendra des *LearningSession* ainsi que des *PedagogicalObjective*. Ces objectifs pédagogiques, bien que structurellement contenus dans le scénario sont liés à une activité pédagogique. La classe *IdentifiedElement* ne contribue pas à la sémantique du langage mais permet de faire porter, par héritage, un identifiant à tous les composants du scénario. Cet identifiant est utile au fonctionnement de la transformation de modèle qui permet l'export du scénario. Les différentes relations de composition, au delà de leurs valeurs sémantiques, seront exploitées par l'outillage DSM afin de représenter graphiquement l'imbrication des différents niveaux.

6.2 CORRESPONDANCES DES ÉLÉMENTS DES NIVEAUX 3 ET 4

Les correspondances mises en œuvre par le tissage de modèles, comme présentées dans le chapitre 5, ne concernent que les activités pédagogiques et leurs implémentations. Pour les éléments des niveaux 3 et 4, les correspondances ne sont pas modifiables et sont prises en charge par la transformation de modèle déclenchée lors de l'export du scénario. Le tableau 6 présente ces correspondances.

TABLE 6 – Correspondances fixes des éléments des niveaux 3 et 4

Concept abstrait (VIDL)	**Implémentation (Moodle)**
PedagogicalScenario	Course
Session	Section
PedagogicalObjective	Outcome
PedagogicalGroup	Group
Seqence	Label (+ paramètres d'indentation et de restriction d'accès du contenu)
AnyOrder	Label (+ paramètres d'indentation du contenu)
ChoiceStructure	Label (+ paramètres d'indentation et de restriction d'accès du contenu)

Le traitement des concepts de scénario et de session consiste simplement à instancier les concepts équivalents (cours et section) du méta-modèle de la plate-forme et à y ajouter leurs contenus après transformation. Les structures d'activités sont plus complexes à traiter puisqu'il en existe plusieurs types avec leurs sémantiques propres. Chaque structure est traduite sous forme d'un label (ou étiquette) et utilise l'indentation du contenu, sur Moodle, pour représenter l'unité de la structure. La différence se fait dans l'usage de la restriction d'accès au contenu, via le suivi d'achèvement.

La séquence est une structure qui vise à forcer l'apprenant à réaliser un ensemble d'activités dans un ordre donné. L'ordre d'apparition des activités (des outils une fois transformés) est conservé et est significatif. L'accès à chaque élément contenu dans la séquence est restreint à l'achèvement de l'élément précédent (sauf le premier). Ainsi, l'apprenant aura accès aux activités au fur et à mesure de sa progression dans la séquence.

Pour la structure *AnyOrder*, l'ordre n'est pas significatif, l'apprenant peut réaliser les activités contenues dans cette structure dans l'ordre qu'il souhaite. L'ensemble du contenu est directement accessible, ainsi il n'est pas nécessaire de faire usage de la restriction d'accès.

La structure *Choice* invite l'apprenant à réaliser une activité, au choix, parmi celles contenues dans la structure. Pour implémenter cette structure nous proposons d'utiliser la restriction d'accès d'une manière différente : l'accès à chaque activité de la structure est conditionnée au non-achèvement de toutes les autres. Ainsi, dès que l'apprenant réalise une des activités contenues dans la structure *Choice*, toutes les autres deviennent inaccessibles.

6.3 SYNTAXE CONCRÈTE : NOTATION GRAPHIQUE

La notation graphique n'a pas fait l'objet d'une étude approfondie [Mooo9] et vise simplement à permettre de distinguer (via les couleurs, les formes, les icônes) les éléments composant le diagramme. Elle reste subjective et discutable. La syntaxe concrète de notre langage de modélisation est spécifiée à l'aide de l'outillage Sirius [Fou16g]. Sirius est un outillage DSM qui permet de concevoir des langages de modélisation graphique. Il se base sur d'autres projets Eclipse tel que EMF (*Eclipse Modeling Framework*) et GMF (*Graphical Modeling Framework*). Vis-à-vis de GMF, l'outillage « standard » utilisé jusqu'à récemment, Sirius se démarque par une approche ne reposant plus sur la génération de code, mais plutôt sur l'interprétation de modèle. La spécification de la syntaxe concrète et du *mapping* avec les éléments de la syntaxe abstraite se font dans un seul modèle appelé *Visual Specification Model* (VSM). Concrètement, nous utilisons Sirius pour spécifier la représentation graphique de chacun des concepts du métamodèle (si elle a lieu d'exister) ainsi que le comportement de chacun des outils de la palette auxquels aura accès l'utilisateur de l'éditeur. La spécification du comportement des outils se fait à l'aide d'un langage dédié, en partie graphique, et s'appuie sur le langage Acceleo [Fou16a]. Le choix de cette technologie a également influencé nos choix quant à la notation du langage, du fait de ses limites (fonctionnement partiel des conteneurs, choix limité de formes prédéfinies etc.).

La sous-section suivante présente la notation graphique des éléments des quatre niveaux présentés précédemment. L'ensemble des éléments présentés ne constitue pas la totalité des éléments du langage mais une sélection représentative permettant d'illustrer la conception de la notation.

6.3.1 *Sessions d'apprentissage*

Comme le montre la figure 37, les sessions d'apprentissage (*LearningSession*) sont représentées sous forme d'un rectangle gris, accompagné d'une icône. Cette icône est la même quel que soit le type de session (*lecture*, *practical work*, etc.). Le label (texte au centre du rectangle) est composé d'un préfixe entre crochets et du nom de la session saisi par l'utilisateur. Le préfixe est ajouté automatiquement et correspond au type de session. Une icône en forme d'horloge en bordure du rectangle indique la durée prévue de la session (*duration*). Si la session ne sera pas opérationnalisée (*operationalized*), une icône barrée apparaît sur le rectangle.

FIGURE 37 – Représentation graphique d'une *LearningSession*

Notre objectif en concevant cette notation est de permettre à l'utilisateur de visualiser directement sur la représentation graphique les principales informations sur l'élément et lui éviter ainsi de chercher dans ses propriétés.

6.3.2 *Structures d'activités*

Les structures d'activités telles que *Seqence*, *ChoiceStructure*, *AnyOrder* sont représentées par des rectangles colorés (en dégradé vers le blanc) comme sur la figure 38. La couleur ainsi que l'icône dépendent de la structure utilisée et, comme pour les sessions, le label est préfixé par le type de structure entre crochets. Le contenu de la structure est directement représenté l'aire du rectangle. L'utilisateur est libre de positionner les éléments internes à la structure comme dans le reste du diagramme et le rectangle est redimensionnable.

6.3.3 *Activités pédagogiques*

Comme le montre la figure 39, les activités pédagogiques sont représentées par des rectangles « arrondis » blancs. Encore une fois, l'icône et la partie entre crochets du label indiquent le type d'acti-

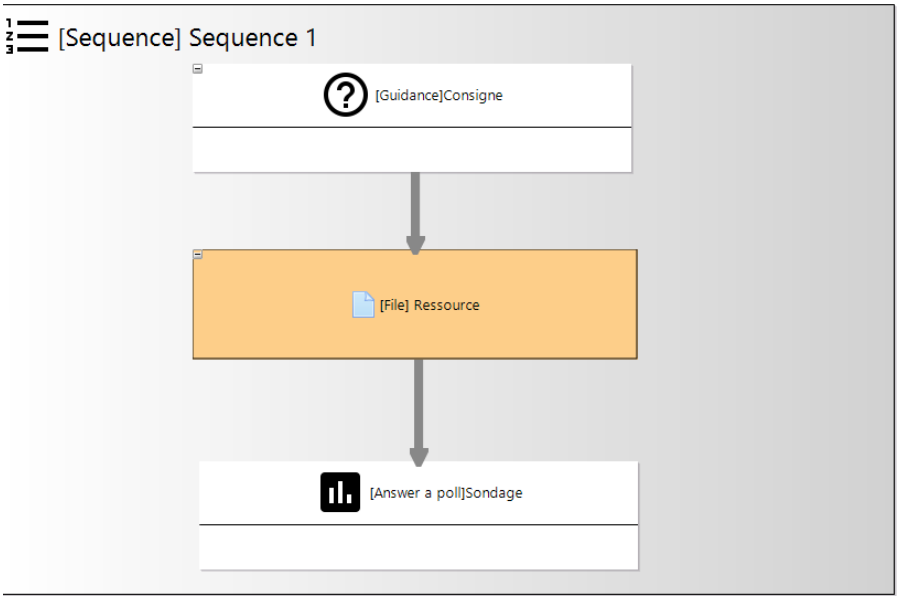

FIGURE 38 – Représentation graphique d'une *Seqence*

vité. Si des groupes sont assignés à une activité pédagogique ils sont représentés par une icône en bordure du rectangle, cette icône est repositionnable par l'utilisateur. Également, si des objectifs pédagogiques sont associés à une activité, ils sont représentés sous forme de liste dans le rectangle.

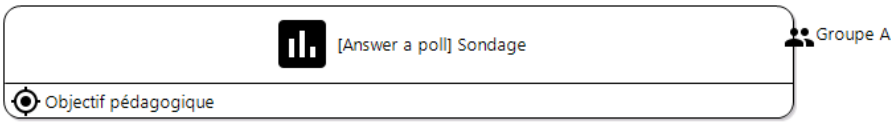

FIGURE 39 – Représentation graphique d'une activité pédagogique *AnswerAPoll*

6.3.4 *Outils Moodle*

Les outils de la plate-forme sont représentés par des rectangles colorés (voir figure 40), sans dégradé ni couleur blanche, afin de ne pas être confondus avec des activités pédagogiques ou des structures. L'icône associée à chaque outil est celle qui l'identifie dans l'interface utilisateur de Moodle. Chaque outil a sa propre couleur et comme pour les autres éléments, le type d'outil est indiqué dans le label entre crochets.

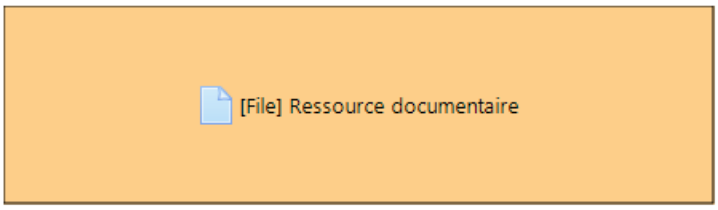

FIGURE 40 – Représentation graphique d'un outil Moodle : *File*

6.4 BILAN

Dans ce chapitre, nous avons présenté les spécifications de notre proposition de langage de scénarisation pédagogique. Sur le plan de la sémantique, nous avons présenté le méta-modèle du langage et expliqué nos choix de conception : un architecture à quatre niveaux exploitant les relations de composition et d'héritage afin de répondre aux besoins utilisateurs de mixer les niveaux d'abstractions et de personnaliser les implémentations, tels que présenté dans la section 4.2. Sur le plan visuel, nous avons présenté la syntaxe concrète (notation graphique). Cette notation, bien que n'ayant pas fait l'objet d'une étude particulière, avait le double objectif de permettre à l'utilisateur de distinguer visuellement les différents éléments du langage (par les couleurs, les icônes, les labels...) et de permettre une lecture rapide des informations principales sur un élément donné (durée, groupes, objectifs etc.).

Nous nous sommes limités aux considérations sur le langage de modélisation, néanmoins certains besoins exprimés par les enseignants-concepteurs portaient sur l'aspect fonctionnel. Dans le chapitre suivant, nous décrirons plus en détails la conception de l'éditeur de scénario et ses fonctionnalités en lien avec les besoins et exigences identifiées.

7

IMPLÉMENTATION D'UN PROTOTYPE D'ÉDITEUR DE SCÉNARIO

Nous proposons d'exploiter le méta-modèle présenté dans la section 6.1 ainsi que les spécifications sur la notation graphique afin de développer un éditeur de scénario à l'aide de l'outillage DSM Sirius. Dans ce chapitre, nous présentons les principales fonctionnalités du prototype d'éditeur que nous avons réalisé ainsi que l'approche technique que nous avons suivie afin de les implémenter. Nous décrivons également l'intégration, dans l'éditeur, des transformations issues du tissage de modèle et de la transformation d'export.

7.1 FONCTIONNEMENT GÉNÉRAL

7.1.1 *Conception avec Sirius*

L'outillage Sirius fonctionne de manière « interprétée » : un unique modèle nommé VSM [1] permet de spécifier les représentations graphiques d'éléments sémantiques et de concevoir les fonctionnalités de l'éditeur à l'aide d'outils. La conception du VSM se fait de manière graphique via un éditeur arborescent (voir figure 41), mais certains champs requièrent d'utiliser un langage de requêtes nommé AQL [2]. La figure 42 illustre l'utilisation d'AQL pour personnaliser le label d'un noeud sur le diagramme.

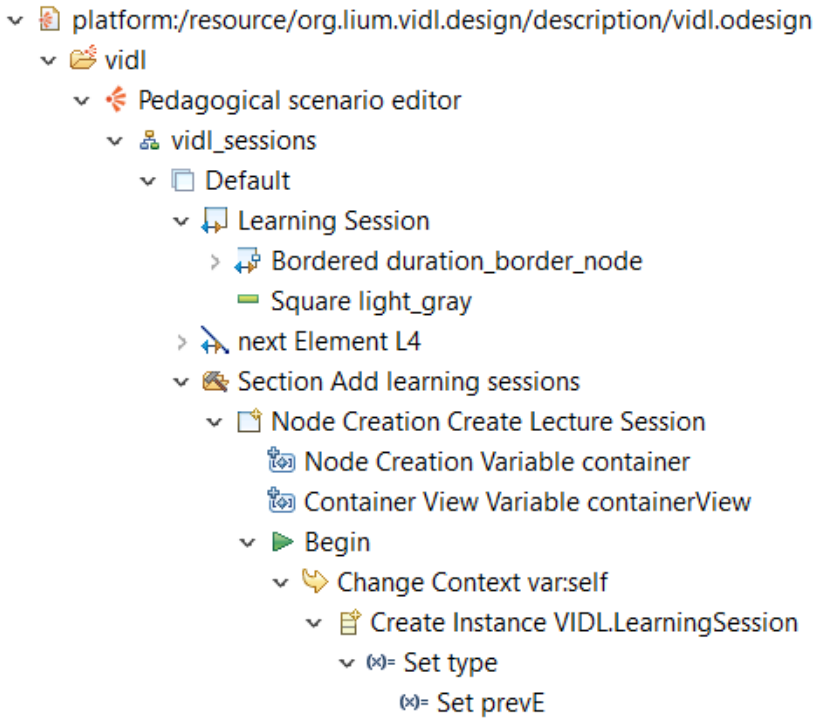

FIGURE 41 – Extrait du *View Specification Model* de notre prototype

1. *Visual Specification Model*
2. *Acceleo Query Language*

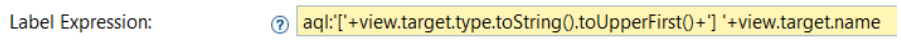

FIGURE 42 – Personnalisation de label avec une expression AQL

Dans un VSM, il est possible de définir un ou plusieurs diagrammes ; pour chaque concept susceptible d'apparaître sur un diagramme il est nécessaire de définir une représentation sous forme de nœud ou d'arc. Un nœud peut être soit un nœud simple (*Node*), soit un conteneur (*Container*). Un conteneur peut contenir d'autres nœuds (*SubNode*) ou d'autres conteneurs. Chaque nœud doit définir un style de représentation visuelle : rectangle, ellipse, losange, image personnalisée etc. Cet élément de représentation permet de personnaliser la couleur, le label, le style du label et d'autres propriétés de mise en forme.

Il existe deux types d'arcs : si l'arc représente une classe du méta-modèle il s'agit d'un *Element based edge*, s'il représente une relation dans le méta-modèle il s'agit d'un *Relation based edge*. Comme pour les noeuds, il est nécessaire de définir une représentation pour les arcs (couleurs, extrémités, tailles etc.).

Les outils doivent obligatoirement être définis dans une section, qui correspond à un groupe d'outils dans la palette d'outils de l'éditeur final. Il existe de nombreux types d'outils : les outils de création apparaissent dans la palette d'outils et permettent « d'instancier » un concept dans le diagramme. Les outils d'éditions sont utilisés de façon plus contextuelle :

— supprimer d'un élément (à l'aide du menu contextuel ou de la touche *suppr.*) ;
— renommer un élément (à l'aide du double-clic lent ou de la touche *F2*) ;
— re-positionner les extrémités d'un arc ;
— glisser-déposer un noeud ;
— coller un élément (à l'aide du menu-contextuel ou des touches *CTRL-V*) ;
— double-cliquer sur un élément ;
— ...

Chaque type d'outil possède des propriétés spécifiques, mais leurs comportements (ce qui est exécuté lorsque ils sont utilisés) sont toujours définis par le concepteur de la même façon. Pour cela, Sirius propose un langage de programmation visuel simplifié qui s'appuie également sur AQL. Le comportement d'un outil est défini par une arborescence d'opérations : une opération correspond à une instruction simple (instancier ou supprimer un élément sémantique, définir la valeur d'un attribut...) et peut être utilisée dans des structures conditionnelles (*If*, *Switch*) ou des boucles (*For*). La figure 43 présente une partie de la définition de l'outil de création d'une activité *Answer a poll*.

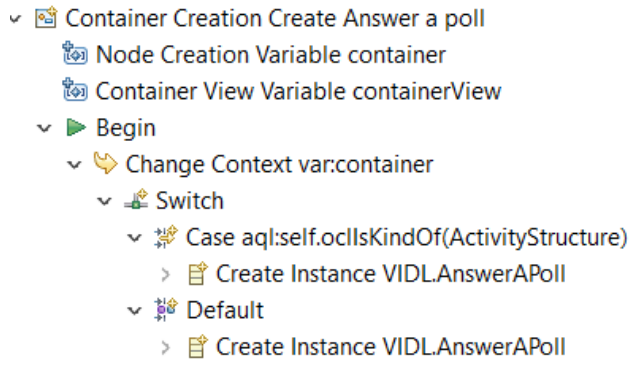

FIGURE 43 – Utilisation d'une structure conditionnelle dans la spécification d'un outil de création de noeud

La documentation pour les concepteurs d'éditeurs avec Sirius [Fou16f] présente en détails le fonctionnement de ce langage ainsi que les différents outils et représentations existantes.

7.1.2 *Fonctionnement de l'éditeur de scénario*

Nous proposons un éditeur de scénarios pédagogiques conçu avec Sirius selon les spécifications présentées dans le chapitre précédent. Cet éditeur est un prototype basé sur l'environnement de développement Eclipse et utilise donc la même interface utilisateur. La figure 44 présente une capture d'écran de l'éditeur où sont mises en avant les zones principales de l'interface :

1. zone de « dessin » où l'utilisateur peut composer le diagramme représentant le scénario,
2. palette d'outils permettant d'ajouter des éléments dans la zone de dessin,
3. volet des propriétés permettant, entre autres, de modifier les propriétés sémantiques d'un élément du scénario,
4. aperçu général du diagramme,
5. explorateur de projets, fichiers et diagrammes.

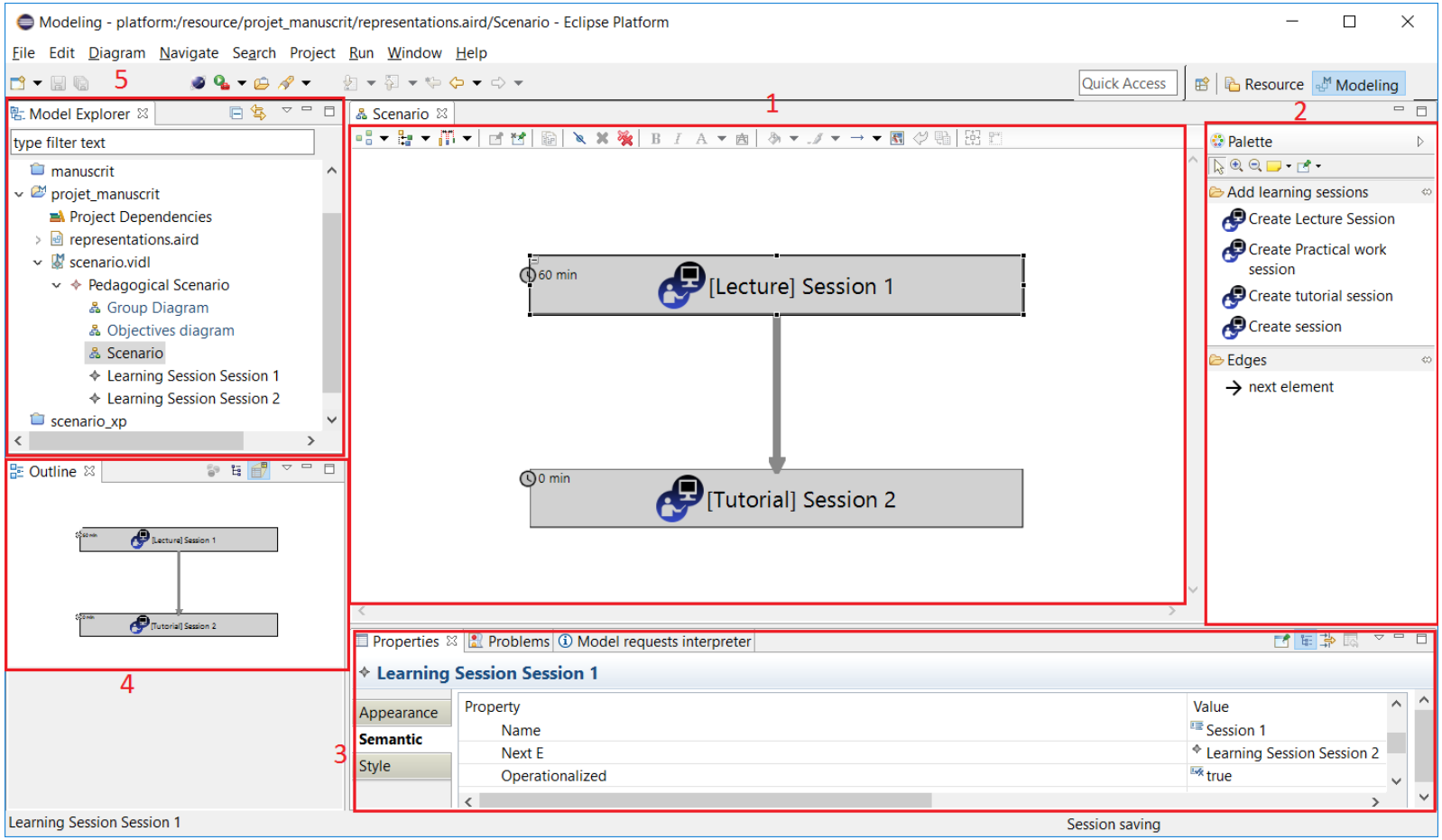

FIGURE 44 – Capture d'écran de l'interface de l'éditeur de scénario

L'agencement de l'interface est celui proposé par la perspective *Modeling* d'Eclipse que nous recommandons. Néanmoins, l'utilisateur est libre de redimensionner, déplacer ou masquer ces différents volets en fonction de ses besoins et habitudes.

Conformément à notre positionnement en termes de langage de scénarisation pédagogique graphique (voir 2.1.4) nous proposons de structurer la conception d'un scénario en « couches ». Ces couches seront des diagrammes correspondant globalement aux quatre niveaux de la syntaxe abstraite (voir 6.1). Chaque diagramme représente le contenu d'un élément du niveau supérieur, excepté le niveau 3, celui des structures, qui ne propose pas de diagramme spécifique puisque, conformément aux spécifications de la notation, le contenu d'une structure est directement représenté dans le conteneur. Le schéma de la figure 45 illustre cette structuration. Dans cette figure, les transitions en pointillés sont déclenchées par un double-clic de l'utilisateur sur l'élément du diagramme correspondant. Bien que nous structurons notre éditeur en trois types de diagrammes, il existera concrètement plusieurs « instances » de ces trois types de diagrammes comme illustrés sur la figure 45.

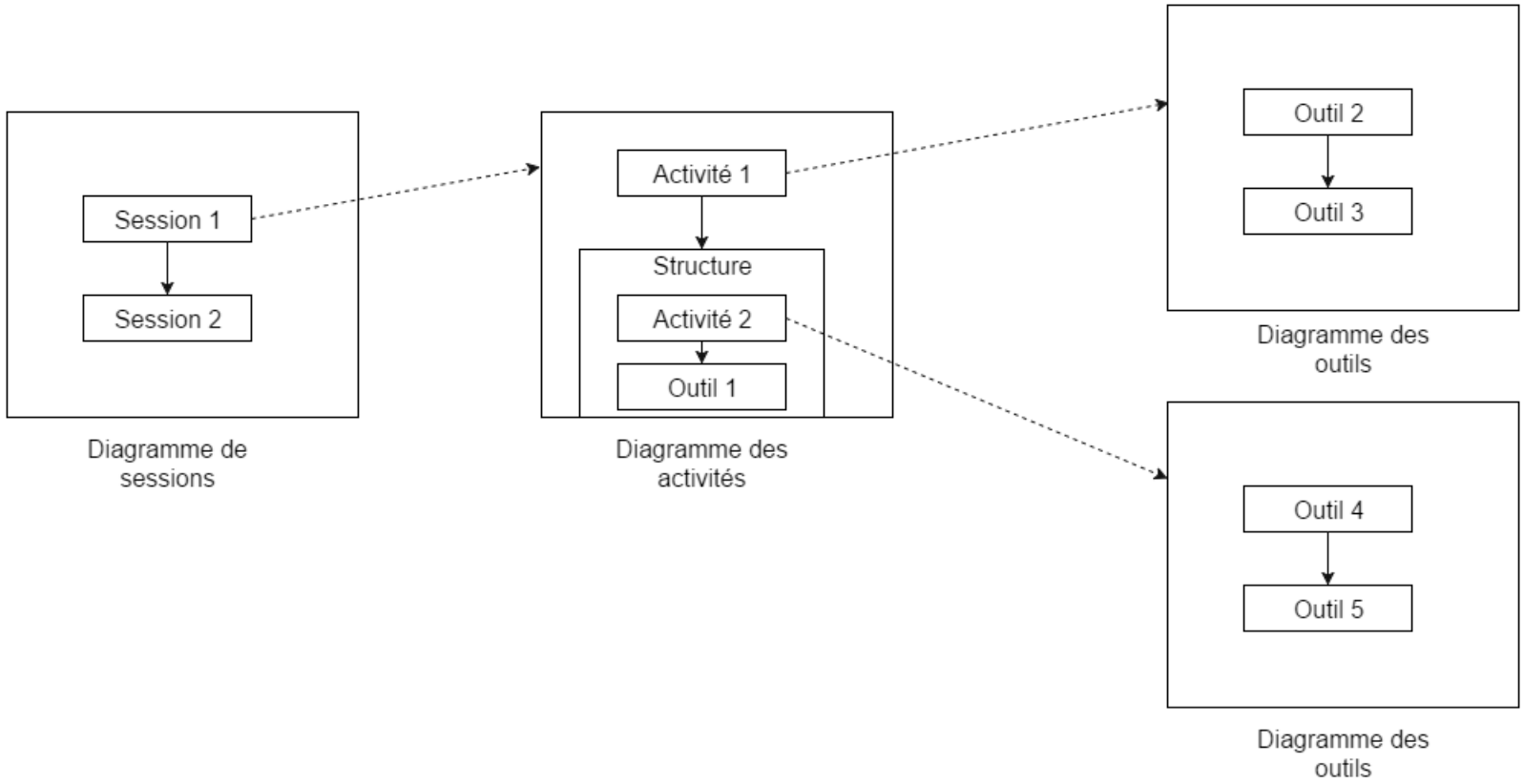

FIGURE 45 – Structuration en diagrammes de l'éditeur

7.2 SCÉNARIO D'UTILISATION

Nous proposons de présenter les fonctionnalités de l'éditeur au travers d'un scénario d'utilisation. Ce scénario illustre une session de conception typique : bien qu'en réalité un scénario pédagogique contiendra plus d'éléments, les actions réalisées par l'utilisateur restent identiques et seront simplement répétées. Nous présentons diagramme par diagramme, les interactions d'un utilisateur afin de modéliser un scénario simple.

7.2.1 *Diagramme de sessions*

Au début de la modélisation d'un scénario avec l'éditeur, l'utilisateur instancie une session d'apprentissage (*Learning Session*). Pour ce faire, il a accès via la palette (à droite sur la figure 46) aux outils adéquats : il est possible de *glisser-déposer* ou bien de cliquer une fois sur l'outil puis une fois sur le diagramme initialement vide afin de créer une première session.

L'utilisateur crée donc une première session de type *Lecture*, la sélectionne puis appuie sur la touche F2 pour la renommer. Il utilise ensuite la fenêtre des propriétés (en bas sur l'illustration 46) pour modifier la durée de cette session. S'agissant d'un cours magistral qui n'apparaîtra pas sur la plate-forme, il désactive l'opérationnalisation de cette session à l'aide du clic-droit et de l'entrée du menu contextuel correspondant.

L'enseignant-concepteur crée ensuite une seconde session, de type *Practical Work*, la renomme et modifie sa durée comme précédemment. Lorsqu'il a ajouté cette deuxième session, elle a automatiquement été désignée comme suivant la session précédente, faisant apparaître une flèche entre les deux sessions. L'ordre lui convenant, l'utilisateur ne repositionne pas cette flèche.

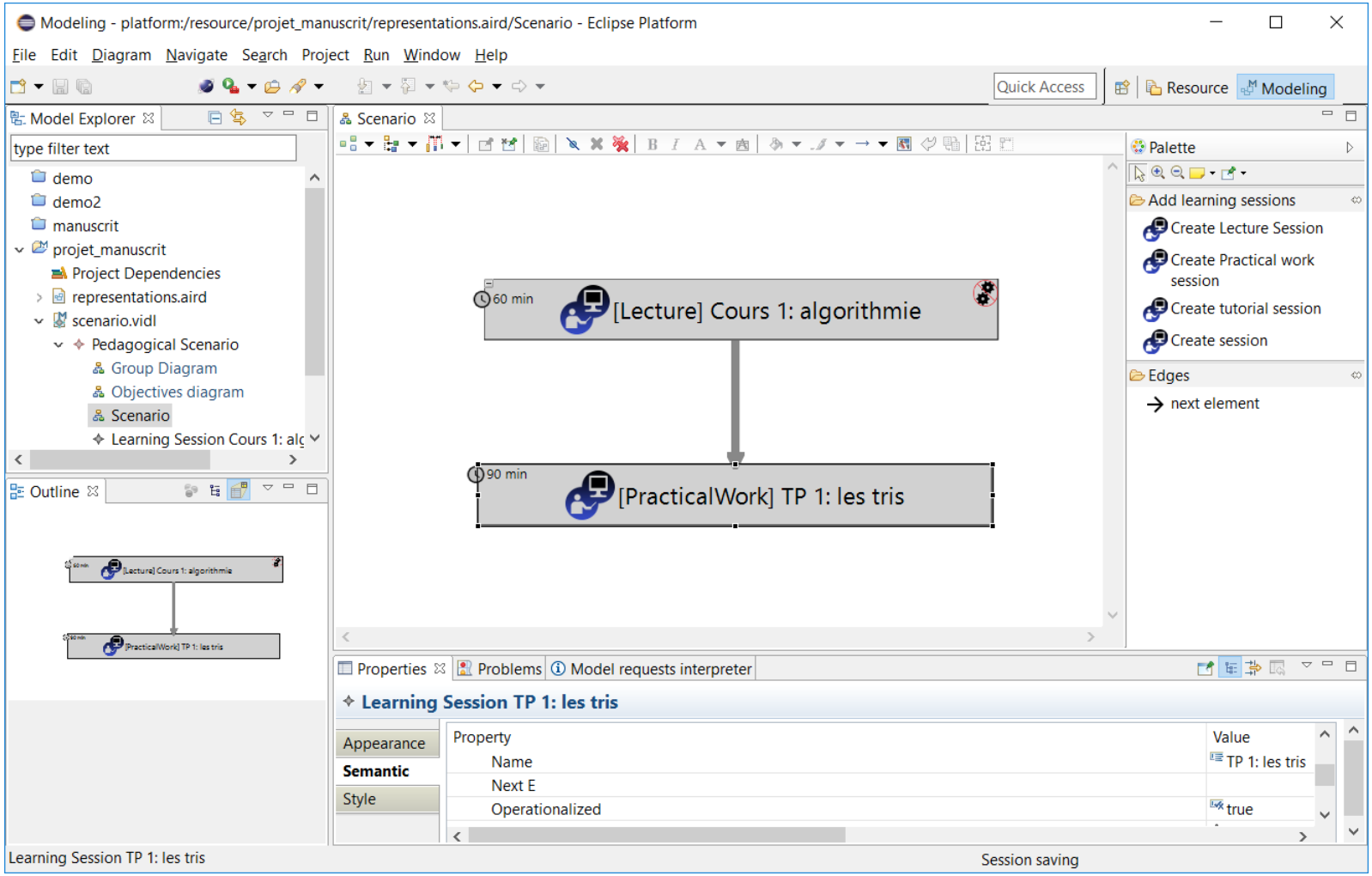

FIGURE 46 – Scénario d'utilisation : diagramme de sessions

L'utilisateur double-clique ensuite sur la session de travaux pratiques afin d'ouvrir un autre diagramme lui permettant de modéliser le contenu de cette session. S'ouvre une boite de dialogue qui lui permet de personnaliser le nom de ce sous-diagramme, il choisit de conserver celui par défaut qui correspond au nom de la session.

7.2.2 *Diagramme des activités*

Un nouveau diagramme, vide, apparaît dans l'espace de dessin et dans les onglets. L'utilisateur crée une séquence, qu'il renomme, puis crée dans cette séquence une première activité de type *Resource Consultation*. Il crée ensuite dans cette même structure un label en utilisant l'outil de la palette adéquat dans la section *Add Moodle tools*. La séquence se termine par une activité de type *Report Writing*, pour laquelle l'utilisateur change la propriété *online* à *false*. La séquence est suivie de deux activités : une *Resource Consultation* dont la propriété *many* est positionnée à *true* et une *Teacher Note* rappelant à l'enseignant de noter les rapports des étudiants une fois déposés (la figure 47 illustre le diagramme résultant).

L'utilisateur double-clique sur l'activité pédagogique de type *Report Writing*[3]. Cela ouvre un autre diagramme lui permettant de visualiser et de modifier l'implémentation de cette activité. Comme présenté dans la section 4.2 et détaillé dans le chapitre 5, le mécanisme **d'implémentation par défaut** est déclenché à ce moment si aucune implémentation n'est déjà définie pour l'activité sélectionnée. Dans le cas de l'activité « Rapport sur les tris », la propriété *online* à *false* a amené le mécanisme de correspondances à proposer l'outil *Assignment* (devoir) comme implémentation.

3. Ne fonctionne ni sur une structure, car il n'est pas prévu de modifier leurs implémentations, ni sur un outil Moodle qui est déjà en soit une implémentation.

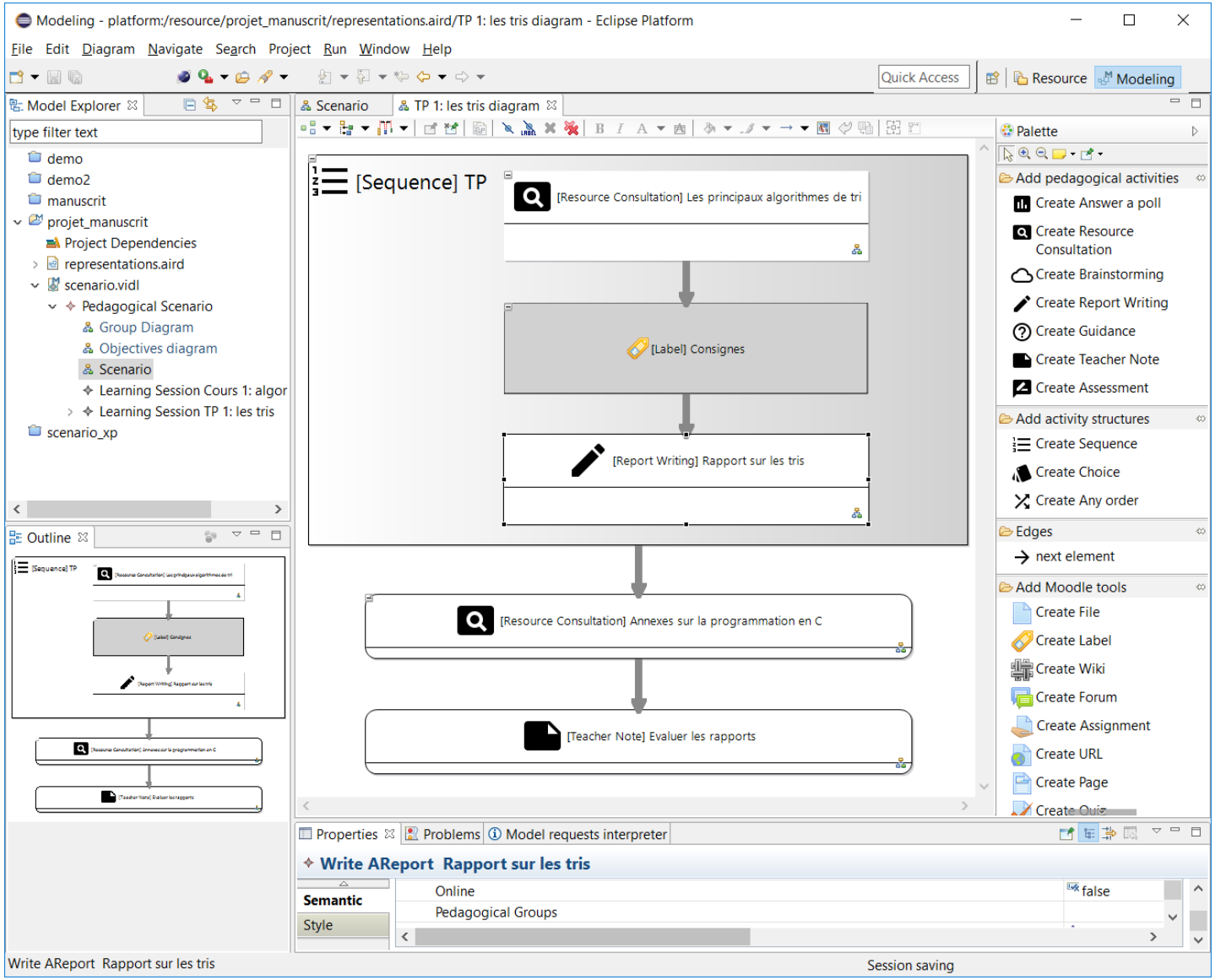

FIGURE 47 – Scénario d'utilisation : diagramme des activités

7.2.3 *Diagramme des outils Moodle*

Le diagramme présente donc l'implémentation par défaut de l'activité « Rapport sur les tris ». L'utilisateur souhaite compléter cette implémentation en ajoutant un forum afin que les apprenants puissent s'entraider avant de déposer leurs rapports individuels. Il sélectionne l'outil *Create Forum* dans la palette et l'applique sur le diagramme (voir figure 48).

L'utilisateur clique ensuite sur l'onglet correspondant au diagramme précédent (diagramme des activités de la session de travaux pratiques), et double-clique sur l'activité « Annexes sur la programmation en C » afin de vérifier son implémentation. Celle-ci n'étant pas déjà définie, le mécanisme de correspondances proposera (suivant la propriété *many* à *false*) l'outil *Folder* (dossier) comme implémentation par défaut (voir figure 49). L'utilisateur peut ensuite répéter ces mêmes étapes pour les autres activités.

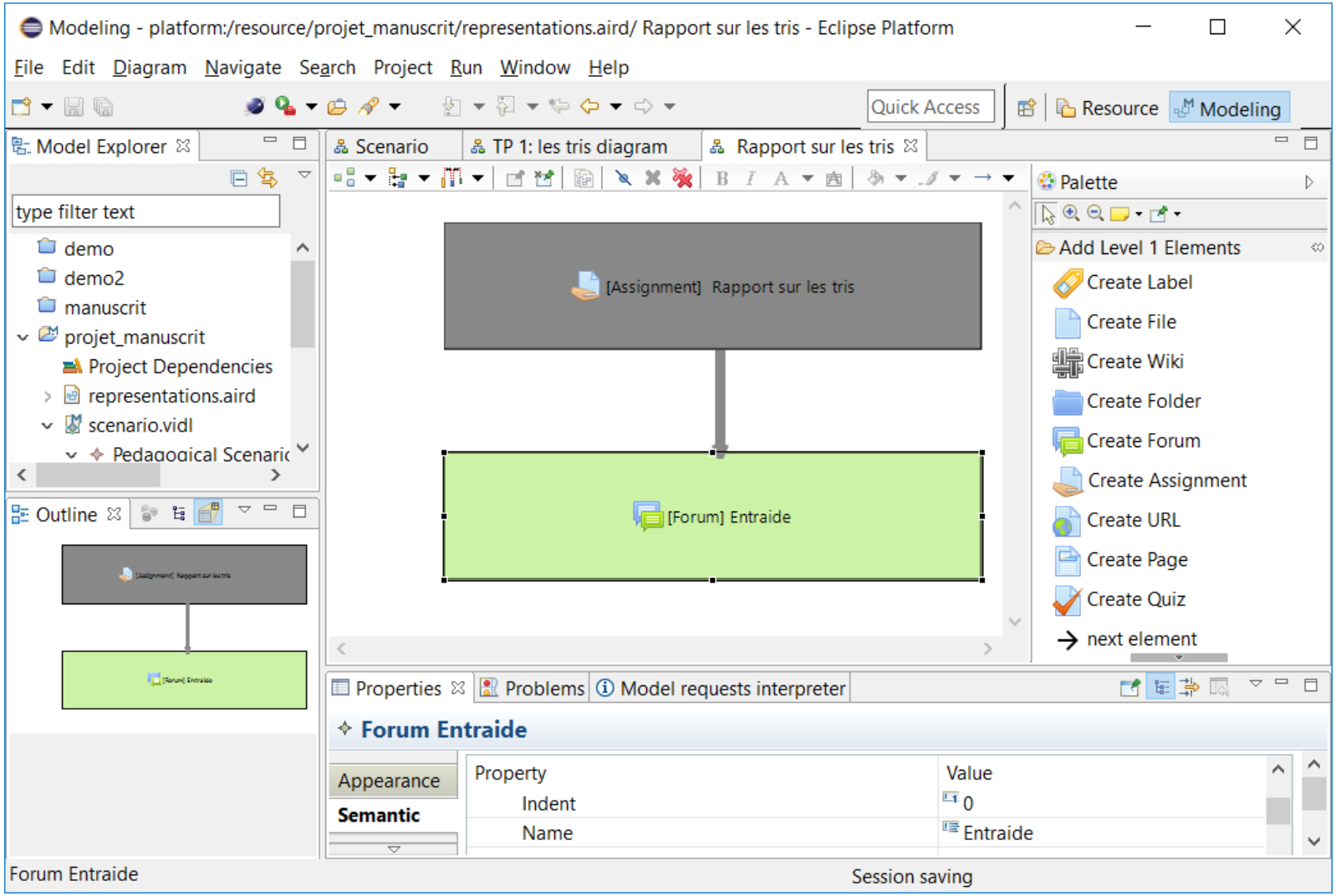

FIGURE 48 – Scénario d'utilisation : diagramme des outils Moodle pour l'activité « Rapport sur les tris »

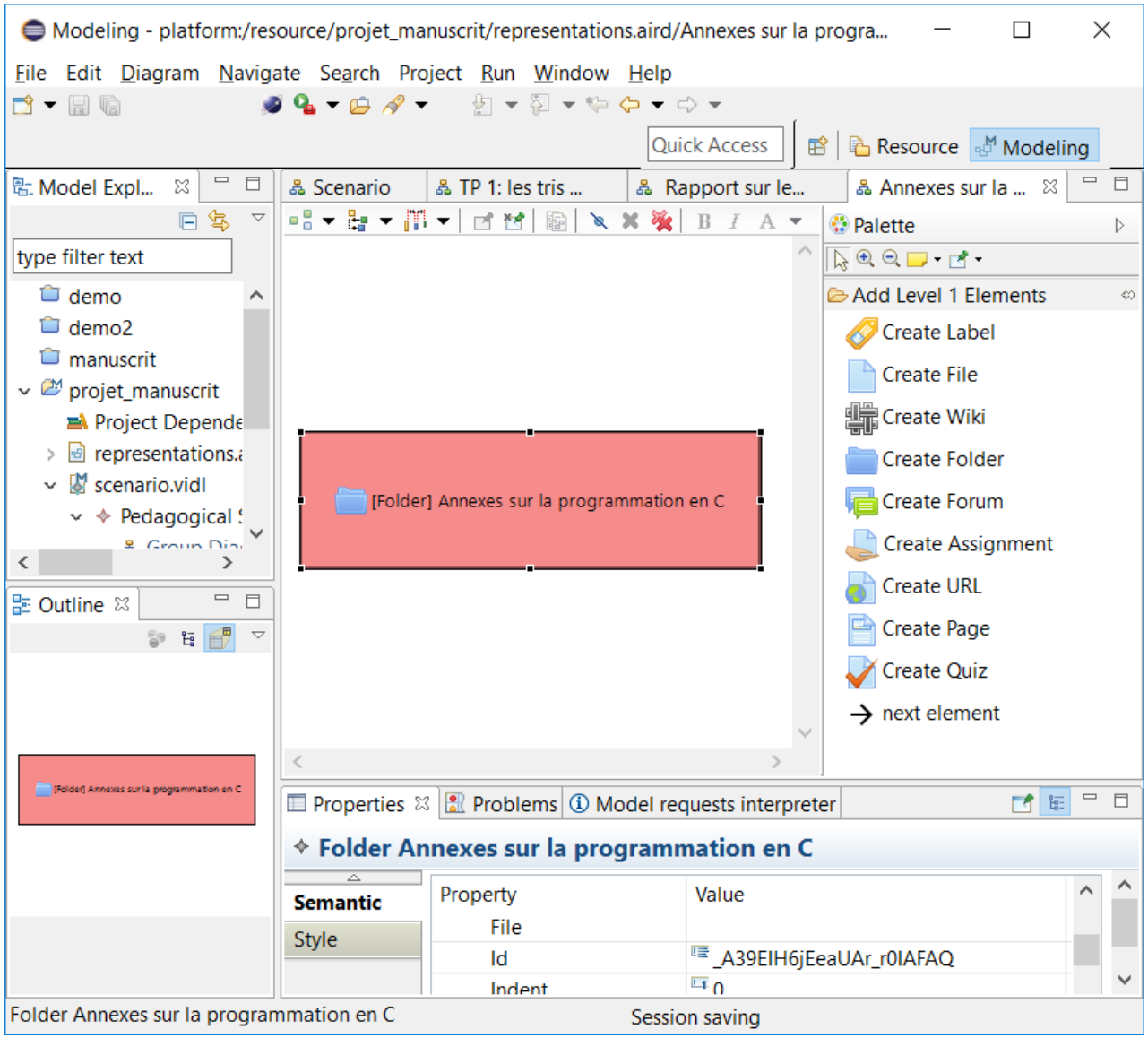

FIGURE 49 – Scénario d'utilisation : diagramme des outils Moodle pour l'activité « Annexes sur la programmation en C »

7.2.4 *Export du scénario*

Une fois la modélisation du scénario terminée, l'utilisateur revient sur l'onglet du diagramme des sessions (nommé Scénario par défaut). Il effectue un clic-droit sur le diagramme et sélectionne l'entrée « Exporter le scénario » du menu contextuel. Cette action déclenche l'exécution de la transformation de modèles permettant l'export du scénario (voir 2.4.1.2) et produit un fichier XML portant le même nom que le modèle d'origine (scenario.xml dans notre exemple).

7.2.5 *Autres fonctionnalités*

Ce scénario d'utilisation illustre les fonctionnalités principales de notre prototype d'éditeur de scénarios. Néanmoins, certaines fonctionnalités que nous avons développées n'ont pas été présentées :

— il est possible de repositionner les flèches *next element* depuis les deux extrémités ;
— il est possible de créer manuellement ces flèches, via l'outil de la palette *next element* ;
— il est possible de copier-coller les éléments d'un diagramme, y compris dans une structure d'activités ;
— il est possible de glisser-déposer des éléments dans ou en dehors d'une structure d'activités ;
— il est possible de supprimer des éléments du diagramme : cela supprime également l'élément sémantique et dans le cas des structures les éléments contenus.

D'autres fonctionnalités, non spécifiques à notre éditeur, sont proposées par défaut avec Sirius comme la possibilité de ré-organiser automatiquement les éléments d'un diagramme, de redimensionner automatiquement un élément, de le masquer...

7.3 CONSIDÉRATIONS SUR L'IMPLÉMENTATION

Sans décrire exhaustivement le développement de l'éditeur, nous souhaitons, dans cette section, revenir sur l'implémentation de certaines fonctionnalités. L'objectif est d'illustrer le développement avec Sirius et d'expliquer quelques problématiques d'implémentation et les solutions que nous avons proposées.

7.3.1 *Exemple d'implémentation d'un outil de création*

Tous les outils de la palette dans l'éditeur permettent d'ajouter un élément sur le diagramme. Ces outils sont implémentés selon le même schéma :

1. instanciation de l'élément sémantique,

2. initialisation des valeurs des propriétés (si nécessaire) de l'élément nouvellement créé,
3. « chainage » avec l'élément précédent (création du lien *next element*).

La création d'un nouvel élément sémantique dans le modèle provoque automatiquement l'affichage d'un élément visuel correspondant. Cet aspect est géré par le champ *Semantic candidate* dans le VSM.

Voici en particulier les principales opérations constituant l'implémentation de l'outil de création d'une session (*LearningSession*) :

— le contexte est défini à *self*, soit la *racine* du diagramme : ici l'élément *PedagogicalScenario* (opération *Change context*),
— l'élément sémantique *Learning Session* est instancié (opération *Create Instance*) dans la relation *structuralComponents* du contexte (le scénario),
— l'attribut *type* de l'élément nouvellement créé est défini à *lecture* (opération *Set*),
— la relation *prevE*, indiquant l'élément précédent, référence le premier élément du diagramme n'ayant pas de suivant.

Dans notre éditeur, la plupart des outils permettant de créer des noeuds sur le diagramme sont implémentés de cette façon. La classe instanciée, la relation de composition contenant le nouvel élément ainsi que les attributs initialisés changent en fonction de l'élément créé.

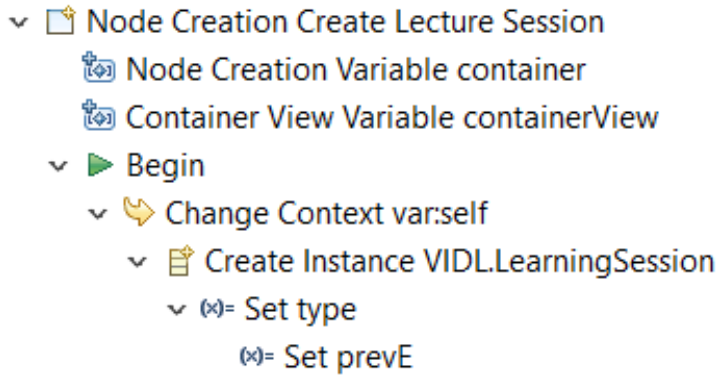

FIGURE 50 – Spécification de l'outil de création d'un noeud *Lecture Session*

7.3.2 *Contexte d'instanciation*

La spécification de l'outil de création d'une activité pédagogique est légèrement différente du schéma présenté précédemment. Une activité pédagogique peut être contenue visuellement soit par le diagramme, soit par une structure d'activités. Sémantiquement, elle correspond soit à la relation *mapedL3Elements* d'une *LearningSession*, soit

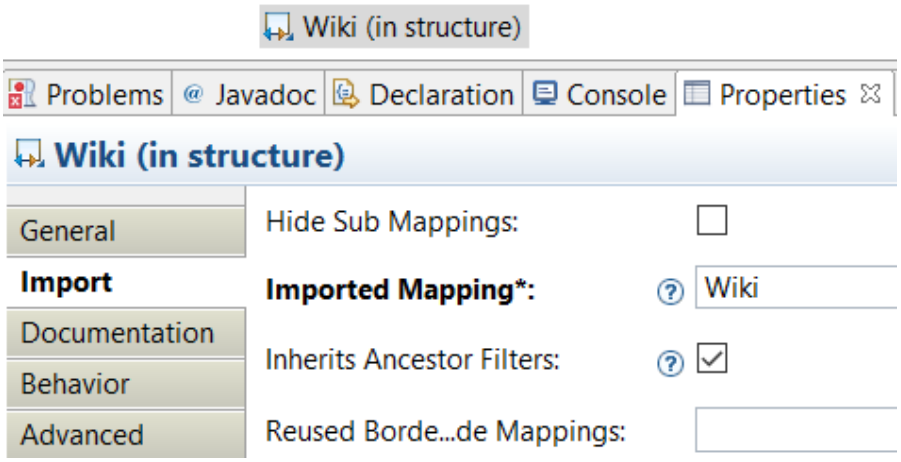

FIGURE 51 – Import de la spécification d'un noeud

à la relation *subL3Elements* d'une *ActivityStructure* (qui peut être *Seqence*, *ChoiceStructure* ou *AnyOrder*). L'opération *CreateInstance* utilisée dans la définition des outils de création nécessite de spécifier la relation de composition contenant l'élément nouvellement créé. Nous utilisons donc une structure conditionnelle dont la condition porte sur le type de l'élément conteneur : selon s'il hérite de *ActivityStructure* nous appliquons une opération *CreateInstance* différente (voir figure 43).

Il en va de même pour les outils de la plate-forme qui peuvent être créés dans trois contextes : dans une *LearningSession* (directement dans le diagramme des activités), dans une *PedagogicalStructure* ou dans une *PedagogicalActivity* (dans le diagramme des outils de Moodle). Encore une fois, nous utilisons une structure conditionnelle (*Switch*) pour traiter différemment ces trois situations.

7.3.3 *Mécanisme d'import dans Sirius*

Du fait des spécifications de notre langage, de nombreux éléments sont redondants entre les diagrammes. Sirius structure le VSM selon ces différents diagrammes. Ainsi l'élément sémantique *Wiki* est à la fois représenté dans le diagramme des activités et dans le diagramme des outils, et sous la même forme. Pour éviter la redondance des définitions des noeuds, arcs et outils dans plusieurs diagrammes, nous utilisons la fonctionnalité d'import proposée par Sirius. Par exemple (voir figure 51) : le *SubNode* Wiki défini dans le conteneur *Sequence* importe la définition graphique du noeud *Wiki* spécifiée dans le diagramme des outils Moodle. Ainsi, si la couleur du noeud *Wiki* est modifiée dans le diagramme des outils Moodle, le changement sera répercuté partout où cette définition est importée.

Nous appliquons cette méthode de « factorisation » des définitions autant que possible, en particulier pour les outils, afin de faciliter les modifications du VSM.

7.3.4 *Intégration du tissage et de la transformation de modèles*

Lorsque l'utilisateur double-clique sur une activité pédagogique, l'évènement est capturé par un outil de type *Double Click*. Dans ce cas, nous déclenchons l'exécution de la transformation de modèle nécessaire au mécanisme **d'implémentation par défaut** puis ouvrons un nouveau sous-diagramme représentant cette implémentation. Pour exécuter la transformation, nous utilisons l'opération *External Java Action* qui permet d'exécuter du code Java. Ce code consiste en une classe implémentant l'interface *IExternalJavaAction* et contenant une méthode *execute(selection, parameters)*. L'implémentation de cette méthode est la suivante :

— récuperer l'élément sémantique (*EObject*) associé à l'élément graphique sélectionné ;
— charger le code de la transformation de modèle ;
— instancier le contexte d'exécution de la transformation (modèle en entrée, modèle en sortie, méta-modèles associés. . .) ;
— exécuter la transformation.

Dans notre cas, le modèle en entrée est le même qu'en sortie : il s'agit du modèle de scénario édité par l'utilisateur. L'élément sémantique à traiter est passé à la transformation via une variable de contexte. La transformation a appliquer est determinée en fonction de la classe de l'élément sélectionné : si l'utilisateur a double-cliqué sur un élément de type *ResourceConsultation* alors le code Java cherchera une transformation a appliquer dans le fichier *ResourceConsultation.eol*.

La transformation de modèle pour l'export du scénario est également exécutée par du code Java, lui même déclenché par une opération de type *External Java Action*.

7.4 BILAN

Ce chapitre a permis de présenter notre proposition d'éditeur de scénario. Du point de vue utilisation : nous avons illustré, via un scénario d'utilisation simple, ses fonctionnalités et l'approche à base de diagrammes imbriqués que nous proposons. Du point de vue développement : nous avons décrit globalement la conception d'un éditeur avec Sirius et détaillé certains aspects de notre implémentation. L'éditeur présenté ici reste un prototype dont l'objectif est de concrétiser notre approche théorique afin de pouvoir vérifier sa validité. Ainsi, même si nous souhaitons qu'il soit utilisable par des enseignants à des fins d'expérimentation, il ne s'agit pas d'un logiciel mature prêt à être déployé.

8

CONCLUSION ET PERSPECTIVES

8.1 SYNTHÈSE DES CONTRIBUTIONS

L'usage des plate-formes de formation en ligne est aujourd'hui répandu chez les enseignants du supérieur. Néanmoins, la conception de scénario pédagogique dans l'optique d'une mise en œuvre sur ce type de plate-forme, que ce soit pour un usage à distance, en enseignement mixte, ou en présentiel, est peu traité dans les travaux de recherche actuels. Il existe de nombreuses solutions permettant de modéliser un scénario pédagogique, selon divers formalismes, mais très peu proposent l'opérationnalisation vers une de ces plate-formes. Nous nous sommes intéressés à ces propositions et en avons étudiés trois dans la section 2.1.5, qui permettent un export vers Moodle. Ces langages et outils proposent une compatibilité limitée avec la plate-forme et induisent des pertes sémantiques entre le scénario tel que le concepteur l'a modélisé, et le cours mis en oeuvre. Leur expressivité pédagogique est également limitée et les concepts proposés sont souvent trop généralistes par rapport aux pratiques des enseignants.

Afin de pallier à ces limitations, nous avons proposé un langage de scénarisation pédagogique graphique. En amont de la conception du langage, nous avons étudié les besoins et les pratiques des enseignants-concepteurs, à l'aide d'une enquête en ligne et d'entretiens individuels et collectifs (voir chapitre 4). Cette étude nous a permis de proposer une abstraction, qui sera l'élément central de notre langage de scénarisation : l'activité pédagogique. Nous appelons activité pédagogique l'abstraction d'un usage récurrent d'un outil (fonctionnalité ou ressource) de la plate-forme associé à son paramétrage. Un outil peut permettre plusieurs usages et une activité pédagogique peut reposer sur plusieurs outils. Nous avons donc proposé une méthode d'identification des activités pédagogiques, afin d'identifier une activité et sélectionner son implémentation (en termes d'outils de la plate-forme) la plus adéquate. Cette méthode mène à la construction d'une matrice qui permet de mettre en correspondance, selon différents critères, une activité et son implémentation.

Dans le chapitre 5, nous avons proposé une solution basée sur le tissage de modèle afin de formaliser et d'opérer ces correspondances. Pour cela nous proposons de spécifier un modèle de tissage mettant en relation les activités pédagogiques issus du méta-modèle du langage et leurs implémentations en terme d'outils issus du méta-modèle de la plate-forme. Ce modèle de tissage est ensuite traité par une transformation de haut niveau (HOT) afin de produire des transformations de modèles qui pourront être intégrées à l'éditeur de scénario.

De notre étude des besoins, nous avons également tiré un ensemble d'exigences pour la conception de notre langage et de l'éditeur de scénario associé. Nous avons pu ainsi établir des spécifications à la fois sur la syntaxe abstraite du langage, formalisée dans un méta-modèle,

et sur la notation graphique, centrée diagramme (voir chapitre 6). Notre méta-modèle est conçu comme une extension du méta-modèle de la plate-forme et est structuré en quatre niveaux. La spécification de la syntaxe concrète (notation graphique) a été réalisée avec l'outillage DSM Sirius, et tient ainsi compte des possibilités et des limites de cet outillage.

Nous avons également proposé un prototype d'éditeur de scénario, développé avec Sirius, implémentant ces spécifications (voir chapitre 7). L'édition du scénario se fait à l'aide de diagrammes imbriqués qui représentent chacun le contenu d'un élément sémantique (respectivement : le scénario pédagogique, une session d'apprentissage, une activité pédagogique). Lorsque l'utilisateur double-clique sur une activité pédagogique, il ouvre un diagramme représentant son implémentation. Si des correspondances ont été modélisées pour ce type d'activité, une implémentation par défaut sera visible dans ce diagramme.

8.2 LIMITES ET PERSPECTIVES

Notre proposition de langage de scénarisation pédagogique graphique centré sur la plate-forme et dirigé vers les praticiens ouvre la voie à de nombreuses perspectives. Ces perspectives sur la structuration du langage, l'abstraction ou la généralisation de notre approche impliquent à la fois une réflexion sur la sémantique des éléments du langage et les pratiques de conception, et sur la conception de l'éditeur à l'aide de l'outillage DSM et des techniques de l'IDM.

8.2.1 *Limite du système de correspondances*

Nous exploitons le tissage de modèle afin de permettre une certaine flexibilité (personnalisation) des correspondances entre concepts abstraits et concepts du méta-modèle de la plate-forme. Néanmoins, il n'est pas possible de personnaliser certaines correspondances qui sont traitées par la transformation d'export du scénario et non par le tissage. La correspondance entre session d'apprentissage et section par exemple est justifiée par les règles de bonne formation d'un cours sous Moodle. Un contournement envisageable est de valider le modèle de tissage avant son traitement par la HOT, afin entre autres de vérifier qu'un concept est bien *mappé* avec le concept de section. Cela nécessiterait d'identifier et d'implémenter toutes ces règles, ce qui constitue un travail de développement non négligeable.

8.2.2 *Théorie de l'activité*

Les travaux sur la théorie de l'activité [Leo81] [Eng14] ont proposé une structuration de l'activité humaine en six points :

— l'objet (et l'objectif),
— le sujet,
— la communauté,
— l'instrument,
— les règles,
— la division du travail.

Notre définition de l'activité pédagogique ne tient pas compte de cette structuration, néanmoins, notre langage permet de modéliser une partie de ces constituants. **L'instrument** (ou l'outil) est représenté par l'implémentation d'une activité : l'ensemble des outils et ressources utiles à la mise en place de l'activité constituent l'artefact avec lequel l'apprenant va interagir. Les **objectifs** (*outcomes*) visés par l'activité peuvent être représentés via les objectifs pédagogiques qui sont modélisables dans notre langage et associés à une activité pédagogique.

La **division du travail** est définie comme la répartition du travail entre les acteurs de l'activité : notre langage ne permet pas de décrire spécifiquement la répartition du travail pour une activité donnée, par contre il est possible de différencier le travail des apprenants en utilisant les groupes. Ainsi, le concepteur peut associer une activité particulière à un groupe et une autre à un groupe différent, la division du travail peut se faire en découpant une activité en sous-activités pour chacun des groupes.

Les **règles** sont modélisées par la sémantique des activités. L'activité *Write A Report* exige de l'apprenant qu'il rédige un rapport, l'implémentation de cette activité peut inclure un outil (label par exemple) pour préciser les modalités de réalisation de l'activité et donc les règles qui la régissent. Certaines propriétés des activités pédagogiques, qui influent potentiellement sur leurs implémentations, permettent de spécifier des conditions particulières à sa réalisation. Les structures d'activités constituent également un moyen de spécifier les règles de réalisation d'un ensemble d'activités : une séquence oblige, en utilisant le suivi d'achèvement et la restriction d'accès sur Moodle, les apprenants à exécuter les différentes activités dans un ordre donné.

La **communauté**, l'ensemble des acteurs intervenant dans l'activité, n'est pas représentée dans notre langage. Il est possible de spécifier des groupes, mais leur constitution en termes d'individus, de nombre, de rôles etc. n'est pas modélisée. Le **sujet**, n'est pas non plus traité dans notre langage de scénarisation.

Une perspective envisageable pour notre proposition est de mettre plus en avant cette structuration de l'activité pédagogique. Actuellement, ces différents éléments sont modélisés de manière incomplète et via des mécanismes différents (propriétés, diagramme d'implémentation etc.). Il semble possible d'orienter le langage et l'éditeur vers une modélisation plus homogène d'une activité pédagogique selon

la théorie de l'activité, en proposant par exemple de modéliser dans un même diagramme : les acteurs de l'activité (la communauté), son implémentation (l'instrument), les consignes pour cette activité (les règles), la répartition du travail selon les rôles (division du travail) et les objectifs pédagogiques qu'elle vise. Le verrou principal est l'implémentation de ces éléments sur la plate-forme. En effet, notre approche vise à éviter toute perte sémantique lors de l'opérationnalisation du scénario.

Moodle propose le concept de rôles, utiles à la division du travail et à la communauté, mais il s'agit plus d'un moyen de définir les droits quant à la gestion de l'espace cours. Il reste la possibilité d'exploiter les groupes pour associer les participants à leur rôle, mais il n'est pas possible de proposer des interactions différentes en fonction du groupe à l'échelle d'une activité.

La modélisation de l'apprenant (le sujet) via le langage de scénarisation, pourrait consister en une modélisation des connaissances et compétences pré-requises à l'activité et de celles visées une fois l'activité réalisée. Moodle propose le concept d'objectifs (*outcome*) qu'il est possible d'associer à une activité et de valider l'acquisition une fois l'activité réalisée. Les pré-requis pourraient être implémentés à l'aide de simples labels indicatifs avant chaque activité.

8.2.3 *Mise à l'essai avec des enseignants-concepteurs*

De par la méthode de recherche orientée *Design-Based Research* (DBR) que nous avons suivie, la spécification et l'outillage de nos propositions relève de l'opérationnalisation de notre proposition théorique et approche générale (extension par abstraction d'un méta-modèle existant pour enrichir son expressivité). La spécification de nos propositions s'est également appuyée sur des besoins et des exigences identifiés auprès du public-cible (cf. chapitre 4). Par construction, le langage de modélisation visuel (VIDL) et son environnement outillé (éditeur et intégrations des différentes transformations) résultent d'un processus de conception et de développement prouvant que nous avons proposé *une* forme d'abstraction possible répondant aux besoins identifiés et réalisée selon les méthodes et techniques IDM et DSM de notre approche.

Les propositions étayées dans les chapitres précédents n'ont pas cherché à démontrer qu'elles étaient l'unique approche théorique d'extension, ni que les techniques et outillage utilisés étaient les seules à pouvoir mettre en oeuvre cette extension. Nous avons cherché uniquement à démontrer qu'il était *possible* de réaliser (spécifier et mettre en oeuvre) une extension de la sémantique d'un méta-modèle de manière à atteindre des exigences précises collectées par un travail exploratoire préliminaire. Nous avons également élaboré un VIDL spécifique (en couches, notation graphique et correspondance textuelle, à

niveau spécification et implémentation) dont les caractéristiques, positionnées suite à l'état de l'art des approches existantes similaires, ont été concrètement mises en oeuvres.

Nous considérons que la validation scientifique de notre proposition est donc, selon la méthode de recherche DBR, intrinsèquement liée à la conception et au développement du VIDL outillé que nous avons proposé.

8.2.3.1 *Vérification versus validation au sens du Génie Logiciel*

Nous pouvons toutefois considérer la validation de nos propositions *logicielles* selon deux perspectives empruntées au Génie Logiciel :

— vérification logicielle : il s'agit de s'assurer que l'outillage développé fonctionne correctement au regard des fonctionnalités conçues et plus particulièrement celles en réponse à une exigence fonctionnelle relevée par les recherches exploratoires ;
— la validation logicielle : il s'agit de s'assurer que la contribution logicielle réponde correctement aux besoins initiaux des utilisateurs finaux (les enseignants-concepteurs dans notre cas).

La vérification logicielle peut être réalisée via différentes mises à l'essai par nous même (processus interne). Le chapitre précédent sur l'outillage intégrait une présentation scénarisée d'un usage type de l'environnement logiciel. Cela correspond à cette vérification interne.

La validation logicielle requiert pour sa part l'intervention de participants externes au projet et correspondant au public-cible (processus externe). L'objectif de cette validation ne serait pas de valider ou invalider les besoins et exigences initiales, surtout qu'il n'est pas envisageable de faire intervenir à nouveau des personnes ayant déjà participé à l'enquête ou aux entretiens. L'objectif consisterait plutôt à s'assurer que l'environnement outillé que nous proposons répond correctement, selon le point de vue d'enseignants-concepteurs, aux besoins et exigences initiaux.

8.2.3.2 *Périmètre logiciel à considérer pour une mise à l'essai*

La contribution logicielle incluse dans le périmètre de cette thèse correspond à l'éditeur graphique réifiant le langage de modélisation visuel ainsi qu'à l'intégration des différents services de transformations internes et externes. Ces contributions participent concrètement au projet GraphiT qui inclut également l'API développée pour Moodle dont le but est d'assurer l'opérationnalisation du scénario (modèle conforme au méta-modèle de Moodle) en espace-cours correspondant. Bien que cette API n'entre pas directement dans le cadre des travaux de cette thèse il nous parait nécessaire de l'exploiter afin de démontrer aux participants que le fichier obtenu par le service d'export, à la fin de leur activité de scénarisation avec l'éditeur gra-

phique, spécifie bien le contenu d'un espace-cours Moodle correspondant à leur scénario.

8.2.3.3 *Vue générale de l'expérimentation*

L'expérimentation consisterait à demander à plusieurs enseignants-concepteurs de spécifier un scénario, donné initialement sous une forme textuelle, à l'aide de notre éditeur graphique, durant une durée maximale commune pré-déterminée. Chaque participant devrait alors proposer une scénarisation pédagogique subjective mais cadrée de manière à assurer la possibilité qu'il ait d'utiliser les principaux concepts propriétés et relations de notre VIDL, ainsi que les fonctionnalités principales de l'éditeur (i.e. celles répondant à une exigence fonctionnelle identifiée). L'utilisation post-scénarisation, par nos soins, de la fonctionnalité d'export puis de l'utilisation de l'API d'import développée pour Moodle, nous permettrait ensuite de montrer à chaque participant le résultat de sa conception pédagogique sous forme d'espace-cours équivalent. Ceci nous permettrait alors de recueillir auprès des participants leurs opinions (adéquation entre les intentions initiales implicites dans leur scénario et la correspondance Moodle). Pour collecter ses données *qualitatives* nous pourrions utiliser, dans un premier temps, un questionnaire individuel, puis dans un second temps, proposer aux participants d'échanger durant une discussion collective qui serait guidée par nos soins en exploitant le fil conducteur du questionnaire.

En imposant la situation sous une forme proche d'un résultat de conception préliminaire (public, contenu didactique, objectifs visés, etc.), cela nous permettrait de proposer aux enseignants de spécifier uniquement un scénario, subjectif, mais cadré par une situation commune à concevoir et mettre en œuvre. Les différences de compétences en conception, scénarisation, mise en œuvre, en général ou pour une plate-forme spécifique, pourraient grandement influencer l'expérimentation. Il parait alors important de fixer la situation d'apprentissage commune à tous les participants. Il sera important d'éviter deux écueils : ne pas assez cadrer la situation pédagogique et trop la cadrer. Dans le premier cas l'expérience et les compétences personnelles de chaque enseignant peut influencer leur activité dans l'expérimentation et rendre très difficile l'interprétation des résultats. Le deuxième cas pourrait entraîner des résultats trop similaires pour l'ensemble des participants et ne permettrait pas de tirer des conclusions sur la valeur ajoutée de l'outillage et de l'approche du projet.

Il sera également important, même s'il s'agit d'une expérimentation quantitative, de s'assurer que le groupe de participants soit représentatif de différents niveaux d'expertise d'utilisation de Moodle et de compétences en scénarisation pédagogique. Afin de constituer de tels groupes nous allons procéder par une sélection de participants volontaires, des enseignants-concepteurs auxquels nous attribuons au

préalable un profil déterminé (*convenience sampling*). Une pré-enquête à remplir permettra d'expliciter, du point de vue du participant, ses niveaux d'expertises. Afin de permettre éventuellement de mettre en place d'autres sessions de mises à l'essai, à des fins de précisions des résultats selon les profils, nous envisageons de demander aux participants, en fin d'évaluation, de proposer les contacts d'autres enseignants-concepteurs qu'ils estiment de niveau d'expertise similaire (*snowball sampling*).

8.2.4 *Autres niveaux d'abstraction*

Les premières réflexions sur l'abstraction, dans le cadre du projet GraphiT, nous avaient menés à envisager plusieurs niveaux d'abstraction dans le langage de scénarisation. La proposition d'abstraction par les usages d'outils de la plate-forme (les activités pédagogiques) est apparue pertinente et importante lors des discussions avec les enseignants-concepteurs et ingénieurs pédagogiques que nous avons rencontrés. Nous avons donc centré notre proposition autour de cette abstraction. Néanmoins, il serait pertinent d'approfondir l'étude d'autres niveaux d'abstraction ou de granularité des éléments de conception du scénario.

Nous avions envisagé l'ajout d'activités composites, ou *templates* : des combinaisons prédéfinies d'un ensemble d'activités pédagogiques pour un objectif donné. L'étude de cas, ou la situation-problème correspondraient à des éléments de cette granularité, il ne s'agit pas d'une unique activité mais d'un ensemble d'activités à réaliser dans un ordre précis pour atteindre un objectif pédagogique.

Un élément de ce type serait considéré comme de niveau 3, selon la structure de notre méta-modèle, mais ne serait pas une structure d'activités. D'un point de vue de la notation, il s'agirait d'un bloc similaire à l'activité pédagogique mais distinct par une icône supplémentaire ou une couleur différente. L'utilisateur manipulerait ces *templates* dans le diagramme des activités, mais en double-cliquant sur un élément de ce type il ouvrirait un sous-diagramme, lui permettant de modéliser le contenu du *template*. Ce sous-diagramme serait semblable à un diagramme des activités (possibilité de mixer outils Moodle, activités pédagogiques et structures) sauf qu'il ne permettrait pas l'ajout de *templates* afin de limiter le niveau d'imbrication.

Le mécanisme de tissage utilisé pour les correspondances entre activités pédagogiques et outils de la plate-forme pourrait également être exploité. Le modèle de tissage capturerait les correspondances entre le *template* et les activités pédagogiques le constituant. Suivant le même processus que pour les activités et les outils, ce modèle permettrait de générer des transformations de modèles qui seraient exécutées pendant l'édition du scénario afin d'y ajouter l'implémentation par défaut d'un *template*.

Une étude de la littérature, sur les patrons pédagogiques [Ber+12], sur les patrons de conception de façon générale [JSP99] [NN98] et sur les templates UML [Dri10], et des pratiques via l'examen d'espaces cours existants et des discussions avec des concepteurs, permettrait d'identifier des *templates* pertinents à intégrer à notre langage.

8.2.5 *Application à d'autres plate-formes*

Nos contributions ciblent spécifiquement le LMS Moodle. Néanmoins, la scénarisation pédagogique outillée est pertinente quelle que soit la plate-forme d'opérationnalisation. Il serait pertinent d'étudier sous quelles conditions notre approche serait généralisable. En effet, nos propositions sont, par construction, spécifiques à la plate-forme d'opérationnalisation. La méthodologie globale, reste cependant identique quelque soit la plate-forme.

Cibler une nouvelle plate-forme nécessite d'avoir accès au méta-modèle du métier de conception de celle-ci. La méthode d'identification proposée par [EOL15] a été conçue pour les plate-formes de formation en ligne mais repose sur des analyses généralisables à tout EIAH : analyse du modèle de données, analyse fonctionnelle, analyse IHM, etc.

En supposant que le méta-modèle de la plate-forme est formalisé, il faut ensuite étudier son extension. Si les structures d'activités semblent être suffisamment universelles pour être conservées, les activités pédagogiques que nous avons proposées sont spécifiques aux pratiques de conception sur un LMS. L'abstraction basée sur l'usage d'un outil avec des paramètres spécifiques est elle toujours pertinente ? Dans le cas des *serious games*, on peut imaginer une abstraction basée sur les mécaniques de jeux. Ainsi, le « moteur » du jeu propose différents ressorts ludiques qui sont capturés dans le méta-modèle et l'abstraction consiste en une activité, issue des pratiques récurrentes des concepteurs, exploitant ce ressort. L'aspect pédagogique de l'activité peut également être considéré : tout comme les activités pédagogiques pour Moodle sont implémentées à l'aide d'outils, les activités pour les *serious games* seraient implémentées à l'aide d'une combinaison de composants ludiques et pédagogiques. Une enquête telle que celle qui a été menée par le projet GraphiT reste coûteuse, mais de simples entretiens avec un groupe d'utilisateurs de la plate-forme permet d'identifier des pratiques récurrentes et des besoins spécifiques qui seront utiles à la conception du langage.

Notre approche par tissage pour les correspondances reste applicable pour relier les différents niveaux d'abstractions. Le méta-modèle de tissage que nous proposons est générique, mais la transformation de haut-niveau a été développée spécifiquement pour nos besoins en transformations et nécessiterait d'être modifiée pour l'application à une autre plate-forme.

L'éditeur graphique nécessite d'être re-développé pour une nouvelle plate-forme. L'approche « interprétée » de l'outillage DSM Sirius permet un développement itératif rapide : la conception d'un prototype d'éditeur ne nécessiterait pas une refonte complète du code de l'application. Une fois le méta-modèle du langage formalisé, la spécification de la notation graphique peut être réalisée en parallèle de son implémentation dans Sirius, afin de la concevoir en accord avec les capacités et les limites de l'outillage.

L'opérationnalisation des scénarios reste très dépendante de la plate-forme cible. Si celle-ci propose une fonctionnalité d'import avec un format de fichier basé sur XML, il est possible d'utiliser une transformation de modèles d'export, comme nous l'avons fait, et/ou une transformation XSLT [1]. Dans le cas d'un format texte autre qu'XML, une approche à base de transformation *Model-to-Text* est envisageable. Si la plate-forme ne propose pas directement l'import, il faut étudier les possibilités d'extension par un *plug-in* ou, en dernier ressort, des modification du code source de la plate-forme.

1. Extensible Stylesheet Language Transformations

BIBLIOGRAPHIE

[Abd09] Firas Abdallah. "Méta-modélisation pour décrire et instrumenter une situation d'apprentissage de pédagogie par projet collectif". 2009LEMA1003. Thèse de doct. 2009, 1 vol. (208 p.) URL : http://www.theses.fr/2009LEMA1003/document.

[Abe13a] Aymen Abedmouleh. "Approche Domain-Specific Modeling pour l'opérationnalisation des scénarios pédagogiques sur les plateformes de formation à distance". Thèse de doctorat dirigée par Choquet, ChristopheOubahssi, Lahcen et Laforcade, Pierre Informatique Le Mans 2013. Thèse de doct. 2013. URL : http://www.theses.fr/2013LEMA1028/document.

[Abe13b] Aymen Abedmouleh. "Approche Domain-Specific Modeling pour l'opérationnalisation des scénarios pédagogiques sur les plateformes de formation à distance". Thèse de doctorat dirigée par Choquet, ChristopheOubahssi, Lahcen et Laforcade, Pierre Informatique Le Mans 2013. Thèse de doct. 2013, p. 134–144. URL : http://www.theses.fr/2013LEMA1028/document.

[Bal+07] Daniel Balasubramanian et al. "The graph rewriting and transformation language : GReAT". In : *Electronic Communications of the EASST* 1 (2007).

[Ber+12] Joseph Bergin et al. *Pedagogical patterns : Advice for educators*. Joseph Bergin Software Tools, 2012.

[Bol+12] Michail Boloudakis et al. "CADMOS : A learning design tool for Moodle courses". In : (2012).

[Bot+06] Luca Botturi et al. "A classification framework for educational modeling languages in instructional design". In : *6th IEEE International Conference on Advanced Learning Technologies (ICALT 2006)*. 2006.

[Bou+15] Erwan Bousse et al. "Supporting Efficient and Advanced Omniscient Debugging for xDSMLs". In : *8th International Conference on Software Language Engineering (SLE)*. Pittsburg, United States, oct. 2015. URL : https://hal.inria.fr/hal-01182517.

[Com08] Benoit Combemale. "Ingénierie Dirigée par les Modèles (IDM) État de l'art". In : (2008).

[Con16] IMS Global Learning Consortium. *Spécification du langage IMS-LD (IMS Learning Design)*. 2016. URL : http://www.imsglobal.org/learningdesign/index.html.

[CTC13] Jean-Pierre Clayer, Claudine Toffolon et Christophe Choquet. "Towards a Pattern-based adaptive approach for Instructional Design Based on Teacher's Pedagogical Design Schemes". In : *15th Int. Conf. on Enterprise Information Systems (ICEIS'13)*. Angers (FRANCE), avr. 2013, p. 532–538.

[Der15] Michael Derntl. "Openglm". In : *The Art & Science of Learning Design*. Sous la dir. de Marcelo Maina, Brock Craft et Yishay Mor. Rotterdam : SensePublishers, 2015, p. 157–168. ISBN : 978-94-6300-103-8. DOI : 10.1007/978-94-6300-103-8_11.

[Dri10] Rim Drira. "Assistance à la modélisation et à la contextualisation de dispositifs pédagogiques complexes". Thèse de doctorat Informatique École Nationale des Sciences de l'Informatique (Tunis) 2010. Thèse de doct. 2010. URL : http://www.theses.fr/2010LIL10141/document.

[DV09] Marcos Didonet Del Fabro et Patrick Valduriez. "Towards the efficient development of model transformations using model weaving and matching transformations". In : *Software & Systems Modeling* 8.3 (2009), p. 305–324. ISSN : 1619-1374. DOI : 10.1007/s10270-008-0094-z.

[DVT10] Juan Manuel Dodero, Álvaro Martınez del Val et Jorge Torres. "An extensible approach to visually editing adaptive learning activities and designs based on services". In : *Journal of Visual Languages & Computing* 21.6 (2010), p. 332–346.

[ElK08] Hassina El-Kechaï. "Conception collective de scénarios pédagogiques dans un contexte de réingénierie : une approche par la métamodélisation située". Thèse de doctorat dirigée par Tchounikine, Pierre Informatique Le Mans 2008. Thèse de doct. 2008, 1 vol. (277 p.) URL : http://www.theses.fr/2008LEMA1019/document.

[Eng14] Yrjö Engeström. *Learning by expanding*. Cambridge University Press, 2014.

[EOL15] Nour El Mawas, Lahcen Oubahssi et Pierre Laforcade. "Identification et formalisation du langage de conception pédagogique des plateformes de formation". In : *EIAH (Environnements Informatiques pour l'Apprentissage Humain)*. Agadir (Maroc), fév. 2015.

[FD06] Kathrin Figl et Michael Derntl. "A Comparison of Visual Instructional Design Languages for Blended Learning". In : *Proceedings of EdMedia : World Conference on Educational Media and Technology 2006*. Sous la dir. d'Elaine Pearson et Paul Bohman. Orlando, FL USA : Association for the Advancement of Computing in Education (AACE), juin 2006, p. 941–948. URL : https://www.learntechlib.org/p/23118.

[Fle+08] Franck Fleurey et al. "A Generic Approach for Automatic Model Composition". In : *Models in Software Engineering : Workshops and Symposia at MoDELS 2007, Nashville, TN, USA, September 30 - October 5, 2007, Reports and Revised Selected Papers*. Sous la dir. d'Holger Giese. Berlin, Heidelberg : Springer Berlin Heidelberg, 2008, p. 7–15. ISBN : 978-3-540-69073-3. DOI : 10.1007/978-3-540-69073-3_2.

[Fou16a] The Eclipse Foundation. *Acceleo Eclipse Project webpage*. 2016. URL : http://www.eclipse.org/acceleo/.

[Fou16b] The Eclipse Foundation. *Eclipse Modeling Framework*. 2016. URL : http://www.eclipse.org/modeling/emf/.

[Fou16c] The Eclipse Foundation. *Generic Eclipse Modeling System (GEMS)*. 2016. URL : http://www.eclipse.org/gmt/gems/.

[Fou16d] The Eclipse Foundation. *Graphical Modeling Framework*. 2016. URL : http://www.eclipse.org/gmf-runtime/.

[Fou16e] The Eclipse Foundation. *Graphiti - a Graphical Tooling Infrastructure*. 2016. URL : http://www.eclipse.org/graphiti/.

[Fou16f] The Eclipse Foundation. *Sirius Specifier Manual*. 2016. URL : https://www.eclipse.org/sirius/doc/specifier/Sirius%5C%20Specifier%5C%20Manual.html.

[Fou16g] The Eclipse Foundation. *Sirius website*. 2016. URL : http://www.eclipse.org/sirius/.

[Gri+09] David Griffiths et al. "From Reload to ReCourse : learning from IMS Learning Design implementations". In : *Distance Education* 30.2 (2009), p. 201–222. DOI : 10.1080/01587910903023199.

[Gro16a] Object Management Group. *OMG's MetaObject Facility*. 2016. URL : http://www.omg.org/mof/.

[Gro16b] Object Management Group. *Spécification de OCL par l'OMG*. 2016. URL : http://www.omg.org/spec/OCL/.

[Gro16c] Object Management Group. *Spécification de QVT par l'OMG*. 2016. URL : http://www.omg.org/spec/QVT/.

[Gro16d] Object Management Group. *Spécification du langage UML (Unified Modeling Language)*. 2016. URL : http://www.omg.org/uml/.

[JBT06] Frédéric Jouault, Jean Bézivin et Atlas Team. "Km3 : a dsl for metamodel specification". In : *In proc. of 8th FMOODS, LNCS 4037*. Springer, 2006, p. 171–185.

[JCV12] Jean-Marc Jézéquel, Benoit Combemale et Didier Vojtisek. *Ingénierie Dirigée par les Modèles : des concepts à la pratique...* Sous la dir. d'Ellipses. Références sciences. Ellipses, fév. 2012, p. 144. URL : https://hal.inria.fr/hal-00648489.

[Jou+06] Frédéric Jouault et al. "ATL : A QVT-like Transformation Language". In : *Companion to the 21st ACM SIGPLAN Symposium on Object-oriented Programming Systems, Languages, and Applications*. OOPSLA '06. Portland, Oregon, USA : ACM, 2006, p. 719–720. ISBN : 1-59593-491-X. DOI : 10.1145/1176617.1176691.

[JSP99] David Jones, Sharonn Stewart et Leonie Power. "Patterns : using proven experience to develop online learning". In : *Proceedings of ASCILITE*. T. 99. 1999, p. 155–162.

[KRB12] Maria Katsamani, Symeon Retalis et Michail Boloudakis. "Designing a Moodle course with the CADMOS learning design tool". In : *Educational Media International* 49.4 (2012), p. 317–331. DOI : 10.1080/09523987.2012.745771.

[Laf10] Pierre Laforcade. "A Domain-Specific Modeling approach for supporting the specification of Visual Instructional Design Languages and the building of dedicated editors". In : *Journal of Visual Languages & Computing* 21.6 (2010). Special Issue on Visual Instructional Design Languages, p. 347–358. ISSN : 1045-926X. DOI : http://dx.doi.org/10.1016/j.jvlc.2010.08.008.

[Laf15] Pierre Laforcade. *Compte rendu de fin de projet - Projet Graphit*. 2015. URL : http://www-lium.univ-lemans.fr/~laforcad/graphit/wp-content/uploads/2014/08/GraphiT-CR-FIN-V2.2.pdf.

[Laf16] Pierre Laforcade. *Site web du projet GraphiT*. 2016. URL : http://www-lium.univ-lemans.fr/~laforcad/graphit/.

[Led+01] Akos Ledeczi et al. "The generic modeling environment". In : *Workshop on Intelligent Signal Processing, Budapest, Hungary*. T. 17. 2001, p. 1.

[Leo81] Alexej N Leontjev. "Problems of the development of the mind". In : (1981).

[LL13] Esteban Loiseau et Pierre Laforcade. "Spécification de langages de scénarisation graphiques centrés sur les plateformes de formation à distance - Etude et expérimentation d'approches DSM pour Moodle". In : *Conférence nationale sur les EIAH*. Toulouse (France), 2013, p. 167–177. URL : http://www.irit.fr/EIAH2013/.

[Loi+15] Esteban Loiseau et al. "Abstraction par méta-modélisation du métier de conception des plate-formes de formation". In : *7ème Conférence sur les EIAH (EIAH 2015)*. Agadir (Maroc), fév. 2015.

[LP04] Anne Lejeune et Jean-Philippe Pernin. "A taxonomy for scenario-based engineering". In : *Inxight Software Inc*. 2004, p. 249–256.

[MFJ05] Pierre-Alain Muller, Franck Fleurey et Jean-Marc Jézéquel. "Weaving executability into object-oriented metalanguages". In : *Proceedings of MODELS/UML'2005*. Montego Bay, Jamaica, oct. 2005. URL : https://hal.inria.fr/hal-00795095.

[Moo09] Daniel Moody. "The Physics of Notations : Toward a Scientific Basis for Constructing Visual Notations in Software Engineering". In : *IEEE Transactions on Software Engineering* 35.6 (nov. 2009), p. 756–779. ISSN : 0098-5589. DOI : 10.1109/TSE.2009.67.

[Ngu08] Thi Thanh Tam Nguyen. "Codèle : An Model Composition Approach for Large-Scale System Engineering". Theses. Université Joseph-Fourier - Grenoble I, déc. 2008. URL : https://tel.archives-ouvertes.fr/tel-00399655.

[NN98] Marc Nanard et Jocelyne Nanard. "Pushing Reuse in Hypermedia Design : Golden Rules, Design Patterns and Constructive Templates". In : *In HYPERTEXT '98. Proceedings of the ninth ACM conference on Hypertext and hypermedia : links, objects, time and space—structure in hypermedia systems*. Press, 1998, p. 11–20.

[Nod07] Thierry Nodenot. "Scénarisation pédagogique et modèles conceptuels d'un EIAH : Que peuvent apporter les langages visuels ?" In : *Revue Internationale des Technologies en Pédagogie Universitaire (RITPU)/International Journal of Technologies in Higher Education (IJTHE)* 4.2 (2007), p. 85–102.

[Our12] El Amine Ouraiba. "Scénarisation pédagogique pour des EIAH ouverts : une approche dirigée par les modèles et spécifique au domaine métier". Thèse de doctorat dirigée par Choquet, Christophe Informatique Le Mans 2012.

Thèse de doct. 2012. URL : http://www.theses.fr/2012LEMA1020/document.

[Pai+09] Richard F. Paige et al. "The Design of a Conceptual Framework and Technical Infrastructure for Model Management Language Engineering". In : *Proceedings of the 2009 14th IEEE International Conference on Engineering of Complex Computer Systems*. ICECCS '09. Washington, DC, USA : IEEE Computer Society, 2009, p. 162–171. ISBN : 978-0-7695-3702-3. DOI : 10.1109/ICECCS.2009.14. URL : http://dx.doi.org/10.1109/ICECCS.2009.14.

[PCC14] Claudine Piau-Toffolon, Jean-Pierre Clayer et Christophe Choquet. *Méthode orientée-patrons pour guider des praticiens lors de la définition et de la contextualisation de leurs besoins et pratiques*. 2014. URL : http://www-lium.univ-lemans.fr/~laforcad/graphit/wp-content/uploads/2014/08/D3.1-graphiT-V1.0.pdf.

[Pod14] Hélène Podvin. *Analyse des pratiques et des besoins de conception pédagogique centrée plateformes de formation*. 2014. URL : http://www-lium.univ-lemans.fr/~laforcad/graphit/wp-content/uploads/2014/08/D3-4_Analyse_-pratiques_-centrees_-PF_1.1.pdf.

[Pri+11] Luis Pablo Prieto et al. "Towards Ubiquitous Learning : 6th European Conference of Technology Enhanced Learning, EC-TEL 2011, Palermo, Italy, September 20-23, 2011. Proceedings". In : sous la dir. de Carlos Delgado Kloos et al. Berlin, Heidelberg : Springer Berlin Heidelberg, 2011. Chap. GLUE !-PS : A Multi-language Architecture and Data Model to Deploy TEL Designs to Multiple Learning Environments, p. 285–298. ISBN : 978-3-642-23985-4. DOI : 10.1007/978-3-642-23985-4_23.

[Raw+02] Adrian Rawlings et al. *Survey of Educational Modelling Languages (EMLs)*. CEN/ISSS WS/LT Learning Technologies Workshop. 2002. URL : https://hal.archives-ouvertes.fr/hal-00703018.

[Tcho2] Pierre Tchounikine. "Quelques éléments sur la conception et l'ingénierie des EIAH". In : *Actes des 2ème assises nationales du GdR I3 - Groupe de Recherche Information Interaction Intelligence, décembre 2002*. Nancy, France, 2002, 13 pages. URL : https://telearn.archives-ouvertes.fr/hal-00190110.

[Tcho9] Pierre Tchounikine. *Précis de recherche en ingénierie des EIAH*. 2009. URL : http://lig-membres.imag.fr/tchounikine/Precis.html.

[TDP11] Nikolaos K Tselios, Stelios Daskalakis et Maria Papadopoulou. "Assessing the Acceptance of a Blended Learning University Course." In : *Educational Technology & Society* 14.2 (2011), p. 224–235.

[Van03] Thomas Vantroys. "Du langage métier au langage technique, une plate-forme flexible d'exécutions de scénarios pédagogiques". Thèse de doctorat dirigée par Derycke, Alain et Peter, Yvan Informatique Lille 1 2003. Thèse de doct. 2003. URL : http://www.theses.fr/2003LIL10131/document.

UNIVERSITE BRETAGNE LOIRE

Université du Maine

Thèse de Doctorat

Esteban LOISEAU

Composition et transformation de modèles pour la spécification de langages de scénarisation pédagogique graphiques centrés sur la plate-forme de formation Moodle

Résumé

Cette thèse s'inscrit à la fois dans le domaine scientifique de l'ingénierie des Environnements Informatiques pour l'Apprentissage Humain (EIAH) et de l'Ingénierie Dirigée par les Modèles (IDM). Elle s'est déroulée dans le cadre du projet de recherche ANR GraphiT. Ces travaux sont fondés sur le constat que les enseignants sous-exploitent les plate-formes de formation en ligne à leur disposition. Par méconnaissance de leurs possibilités, la mise en œuvre de scénario pédagogique reste difficile. Les solutions actuelles pour la scénarisation pédagogique ne tiennent pas compte de la plate-forme. Si elles proposent une compatibilité avec ces dispositifs, elle entraine souvent une perte sémantique. Notre approche consiste à spécifier un langage outillé de scénarisation pédagogique qui 1/ soit centré sur une plate-forme de formation donnée et 2/ permette d'élargir l'expressivité des scénarios interprétables par cette plate-forme. Cette thèse s'intéresse à la plate-forme Moodle. Un travail préalable a permis l'identification du métier de conception de cette plate-forme et sa formalisation dans un méta-modèle. Nous l'avons étendu avec des activités pédagogiques ainsi que d'autres éléments de scénarisation encapsulant les paramétrages de la plate-forme. Nous avons étudié l'identification de ces activités et proposé une approche à base de tissage de modèle pour la spécification et la mise en oeuvre des correspondances entre activités pédagogiques et fonctionnalités de la plate-forme. Nous avons proposé une notation graphique et élaboré un langage de scénarisation centré Moodle. Ce langage a pu être réifié dans un éditeur exploitant les outillages du *Domain Specific Modeling*.

Mots clés
Scénarisation pédagogique, Ingénierie Dirigée par les Modèles, plate-forme de formation en ligne

Abstract

This thesis is part of the GraphiT project, drawing from both TEL-system Engineering and Model Driven Engineering research domains. This research work is based on the finding that teachers do not use the full potential of Learning Management Systems at their disposal. Because of their partial knowledge of LMS features, teachers often struggle to implement their pedagogical scenarios. Current instructional design tools and languages do not take the LMS into account, and rarely provide compatibility features with it. In this context, we aim at specifying a platform-specific visual instructional design language that matches teachers practices and needs. We focus on the Moodle platform, as previous works lead to the formalization of its instructional design semantics in a metamodel. We extended this metamodel with pedagogical activities and other high-level pedagogical elements to hide the complexity of Moodle features settings. We proposed an approach based on model weaving and model transformations to specify and execute mappings between pedagogical activities and their Moodle implementations. We specified a graphical notation for our instructional design language and developed a learning scenario editor using a Domain Specific Modeling tooling.

Key Words
Instructional Design, Model Driven Engineering, Learning Management System