# Des situations de modélisation pour décrire un processus de modélisation

Antoine Beugnard, Fabien Dagnat, Sylvain Guerin, Christophe Guychard

▶ **To cite this version:**

Antoine Beugnard, Fabien Dagnat, Sylvain Guerin, Christophe Guychard. Des situations de modélisation pour décrire un processus de modélisation. Revue des Sciences et Technologies de l'Information - Série ISI : Ingénierie des Systèmes d'Information, 2015, 20 (2), pp.41 - 66. 10.3166/ISI.20.2.41-66 . hal-01164480

**HAL Id: hal-01164480**
**https://hal.science/hal-01164480**

Submitted on 14 Mar 2022

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# Des situations de modélisation pour décrire un processus de modélisation

**Antoine Beugnard** [1], **Fabien Dagnat** [1], **Sylvain Guérin** [2], **Christophe Guychard** [2]

1. *Télécom Bretagne, IRISA,*
*Technopole Brest-Iroise, CS 83818*
*29283 Brest cedex 3, France*
*prenom.nom@telecom-bretagne.eu*

2. *Openflexo*
*Technopole Brest-Iroise, 135 rue Claude Chappe*
*29280 Plouzané, France*
*prenom.nom@openflexo.org*

RÉSUMÉ. *Nous proposons d'identifier des situations de modélisation en mettant en évidence des actions élémentaires sur les artéfacts de modélisation que sont les modèles et les méta-modèles. Nous illustrons la démarche de modélisation par une étude de cas détaillée. Nous pensons que l'identification de ces situations élémentaires permet de mieux comprendre la modélisation et par conséquent les besoins des outils de modélisation. Nous présentons Openflexo, un outil de* modélisation libre *qui nous permet de mettre en œuvre cette approche.*

ABSTRACT. *We propose to identify modeling situations highlighting elementary actions on modeling artifacts such as models and meta-models. We apply a modeling process to a detailled use case. We believe that identification of these basic situations helps better understand the modeling and therefore modeling tools requirements. We present Openflexo a* free modeling *tool that helps us implementing this approach.*

MOTS-CLÉS : *Modélisation, processus, notation.*

KEYWORDS: *Modeling, process, notation.*

## 1. Introduction

La modélisation est à la base de la plupart des stratégies de résolution de problèmes, en particulier de la démarche scientifique et de l'ingénierie (Morris, 1967 ; Lachapelle, Cunningham, 2007). On modélise en s'appuyant sur des écrits [1] sous formes de phrases dans le cas de modèles textuels ou de dessins pour les diagrammes. Ces phrases et ces dessins, artéfacts résultats de la modélisation, doivent respecter des règles de construction plus ou moins explicites ou formalisées.

Dans « On the art of modeling » (Morris, 1967), W. Morris propose de passer d'un processus de modélisation intuitif à une approche explicite. Il illustre l'article par un problème de planification de transport. Au-delà des étapes de réflexion, il fait apparaître deux étapes qu'on retrouve dans tout processus de modélisation :

1. Identifier une instance spécifique du problème ;
2. Déterminer leur généralisation ou abstraction, et en définir des représentations, à travers des variables mathématiques par exemple.

Il note que la production de ces artéfacts (les exemples et les symboles) suit un processus d'élaboration par enrichissement :

> « Le processus de modélisation peut être utilement vu comme un processus d'enrichissement ou d'élaboration. On commence avec des modèles très simples, assez différents de la réalité, et on essaie de façon évolutive d'aller vers des modèles plus élaborés qui reflètent plus précisément la complexité de la situation considérée. »

Cette vision, par enrichissement uniquement, est simplificatrice, dans la plupart des cas, les enrichissements sont suivis de réductions (filtrage, sélection) des modèles. Enfin, W. Morris met également en évidence le besoin de liens (implicites ou explicites) entre ces artéfacts :

> « L'*analogie* ou l'*association* avec des structures logiques bien développées précédemment jouent un rôle important dans le choix du point de départ de ce processus d'élaboration ou d'enrichissement. »

Plus récemment, pour l'apprentissage de résolution de problème, les auteurs de (Lachapelle, Cunningham, 2007) proposent le processus cyclique explicite suivant :

« 1. QUESTIONNE – Quel est le problème ? Qu'ont fait les autres ? Quelles sont les contraintes ?

2. IMAGINE – Quelles sont les solutions possibles ? Remue-méninge d'idées. Choisir la meilleure.

3. PLANIFIE – Quel diagramme peut nous aider maintenant ? Faire une liste des choses nécessaires.

---

1. Même si les premiers philosophes discutaient sur l'Agora sans laisser de traces écrites et que nos ancêtres ont dû résoudre bien des problèmes avant l'invention de l'écriture…

4. CRÉE – Suivre le plan et créer. Tester et confronter.

5. AMÉLIORE – Discuter ce qui marche, ce qui ne marche pas, et ce qui pourrait mieux marcher. Modifier la conception pour la rendre meilleure. La tester. »

Dans cet article, nous nous concentrons sur la partie du processus qui est outillée ce qui concerne les parties PLANIFIE, CRÉE et AMÉLIORE.

Dans la suite de cet article, nous appelons *modèle* une représentation explicite d'un système ou d'un problème et *méta-modèle*, l'ensemble des règles explicites - quel que soit leur mode d'explicitation - qu'un modèle doit respecter. Nous verrons que les liens entre modèle et méta-modèle peuvent être connu *a priori*, mais sont parfois élaborés *a posteriori*, comme le résultat d'un choix de modélisation.

La formalisation de la gestion de l'écriture des outils de modélisation s'est concrétisée, pour les phrases, par la théorie des langages et des grammaires, pour les dessins, par les approches dites dirigées par les modèles (IDM). Dans les deux cas, la représentation d'un modèle produit (phrase ou dessin) est *conforme* aux règles exprimées respectivement dans la grammaire ou le méta-modèle (au sens IDM). Cette relation de conformité est l'une des relations possibles entre modèles et méta-modèles. Nous considérerons donc deux niveaux de modélisation : le modèle (exemple, instance, concrétisation) et le méta-modèle (généralisation, abstraction). Dans le sens où nous l'utilisons, le méta-modèle (Kleppe, 2007) est l'expression des concepts (lexique, ou vocabulaire) utilisables pour élaborer un modèle, de toutes les règles et contraintes (grammaire) d'assemblage de ces concepts et de l'ensemble des règles d'écriture (syntaxe concrète) qui permettent de les représenter. Notre approche est abstraite et peut être considérée comme « catégorique » au sens mathématique du terme. Nous nous intéressons aux relations entre modèles et méta-modèles sans prendre en considération leur représentation interne et la façon dont ils sont réalisés ou stockés.

Nous notons que dans la plupart des outils et démarches de modélisation existants, la relation de conformité est considérée comme stricte i.e. le modèle conserve exactement et uniquement les propriétés définies dans son méta-modèle (qui doit préexister), mais nous ne retenons pas cette hypothèse dans notre démarche. En effet, elle se révèle parfois être une contrainte prescriptive qui empêche un concepteur de pleinement décrire son modèle car le méta-modèle ne l'a pas prévu. Nous montrerons que la relation liant modèle et méta-modèle s'élabore également pendant la modélisation. La conformité n'est pas nécessairement une donnée initiale, elle se construit avec le modèle. Dans (Kelly, Pohjonen, 2009), Kelly et Pohjonen notent plusieurs mauvaises pratiques liées à la conformité imposée a priori (*Predetermined Paradigm, If You Have a Hammer…, The Library Is the Language, Sacred at Birth*, etc). Ainsi, dans notre approche, la conformité peut être choisie comme prescriptive ou non et ce sur tout ou partie du modèle suivant les besoins du concepteur. Des outils peuvent être imaginés pour vérifier si une conformité non-prescriptive spécifiée entre un (élément de) modèle et un (élément de) méta-modèle est vraie.

Enfin, on rencontre aussi souvent la relation d'instanciation (*instanceof*) qui relie un modèle créé depuis un méta-modèle à ce méta-modèle. Cette relation admet

dans la plupart des outils de modélisation une définition qui exprime la conformité au méta-modèle au sens déjà vu, mais y ajoute que la réalisation de l'instance est obtenue directement suivant la forme définie par le méta-modèle (qui devient une sorte de moule à instances). On notera que cette vision de la relation d'instanciation n'est pas aussi restrictive dans les approches à base d'ontologies. Cela donne lieu dans (Kühne, 2006) à une séparation de la relation d'instanciation en deux sous-concepts : instanciation linguistique et ontologique. Dans le cadre de cet article et suivant la vision catégorique déjà exposée, nous ne nous intéressons pas à la réalisation des artéfacts de modélisation. Nous ne restreignons donc pas la définition de la relation d'instanciation à celle (linguistique) inspirée des langages de programmation objet.

Pour montrer la nécessité d'une approche de la modélisation vivante avec des interactions diverses entre modèles et méta-modèles et des méta-modèles évolutifs, nous présentons quelques exemples (partie 2). Nous enchaînons en détaillant un cas d'étude (partie 3), mené avec une collectivité territoriale. Puis, pour raisonner sur la dynamique de la modélisation et pour servir de base à la définition d'exigences pour les ateliers de modélisation, nous identifions des usages de ces deux niveaux de modélisation au travers des situations (partie 4) rencontrées lors de travaux de modélisation en les illustrant avec des exemples concrets. Nous abordons rapidement l'enchaînement de ces situations (partie 5) et les discutons. Nous présentons ensuite la démarche suivie avec l'atelier Openflexo (partie 6) qui nous permet d'expérimenter notre vision de la modélisation libre. Pour finir, nous comparons notre approche avec d'autres travaux dans la partie 7 avant de conclure.

## 2. Exemples de modélisation

Pour commencer, nous décrivons des situations vécues de modélisation avec des professionnels. Nous souhaitons, par ces exemples, montrer comment sont élaborés des modèles dont le méta-modèle n'est pas connu (ou partagé). Un tel modèle ne peut pas être conforme *a priori*. La non-conformité est une nécessité initiale et la conformité un résultat de l'activité de modélisation. On élabore des modèles dans le but de construire une *représentation* et une *interprétation partagée* (le sens) dans le but de communiquer (ou calculer) avec soi-même ou avec d'autres.

### *2.1. Modéliser l'activité d'une collectivité territoriale*

Une collectivité territoriale fait interagir des populations très différentes. D'un côté, les élus qui définissent les orientations, développent la vision stratégique et qui ont une perception de l'organisation de la collectivité propre. Ils n'ont que rarement besoin de savoir comment les choses sont réalisées. De l'autre côté, les services ont une organisation hiérarchique fonctionnelle précise qui met en œuvre le comment, et alimente en informations les élus pour aider à la prise de décision.

Cette vision est toute théorique. Des résistances existent, des informations sont cachées dans les deux sens, des enjeux de pouvoir s'exercent. Dans le contexte écono-

mique actuel, l'optimisation du fonctionnement de ces organisations devient cruciale et des demandes d'outils apparaissent.

Nous participons, en collaboration avec une collectivité territoriale, à l'élaboration d'un outil qui facilitera la mise en correspondance des deux visions du fonctionnement de cette collectivité. Chaque partie vient avec sa propre conceptualisation, parfois ses propres outils, pour décrire cette complexité. Notre travail est de construire et d'outiller des langages permettant une interprétation commune. Il est inconcevable de faire adopter par l'une ou l'autre partie le modèle de l'autre, mais il est fondamental de trouver une zone de consensus.

Le dialogue avec chacun des acteurs conduit à une émergence simultanée des représentations construites sous forme de dessins ou de feuilles de calculs. Chacune d'entre-elles répond, outre au besoin de communication avec les autres acteurs, à des préoccupations propres à chacun. Notre travail consiste dans un premier temps à identifier avec eux les éléments conceptuels permettant la compréhension commune (le consensus), et de les différencier des éléments spécifiques à chaque classe d'acteurs (concepts et éléments d'écriture). Il s'agit ensuite de travailler sur cet ensemble de concepts et de représentations pour les relier, les contraindre, les interpréter et les outiller.

Le résultat qu'on cherche à obtenir doit permettre à chacun de s'approprier l'information et de la manipuler dans son langage (avec des écritures différentes), sans la dénaturer (conformité conceptuelle), tout en s'appuyant sur les sources de données existantes : organigrammes, cartographies des missions de la collectivité, inventaire des directives stratégiques, etc. On obtient alors plusieurs langages spécialisés à leur domaine, articulés autour d'intersections (au sens d'ensemble d'éléments communs) sur lequelles l'interprétation est partagée. La correction de l'échange est alors garantie car aucun glissement d'interprétation n'est nécessaire.

Ce cas est développé dans les parties 3 (méthode) et 6 (outils) pour mettre en évidence la démarche et les situations de modélisation décrites dans la partie 4.

### *2.2. Utiliser un DSL pour décrire l'interaction de composants*

L'intégration de composants logiciels au sein du même système multi-plateforme (différents langages, différentes architectures cibles) nécessite de parfaitement maîtriser les interfaces entre ces composants. Cela suppose d'en partager la définition avec l'ensemble des acteurs chargés du développement de ces composants, tout en respectant les spécificités de chaque système cible.

Cette situation, rencontrée chez un industriel développant des systèmes embarqués complexes, nous a conduit à travailler sur la définition d'un langage dédié (DSL [2]) pour décrire ces interfaces. Une collection de générateurs, co-construits avec les spé-

2. Domain Specific Language

cialistes de chaque plate-forme, exploite ce langage pour produire du code spécifique à chaque plate-forme cible.

Le succès d'une telle démarche repose sur la capacité du langage à capturer les préoccupations de sa communauté d'utilisateurs. Bien que réunissant des ingénieurs dont les disciplines ont une certaine proximité, les communautés présentent une diversité d'usages, de pratiques et de contraintes qui rendent difficile l'établissement d'un consensus.

### *2.3. Modéliser avec UML*

Le langage de modélisation UML est considéré très largement comme un standard de modélisation de systèmes à logiciel prépondérant. Il couvre une grande part du cycle de vie du logiciel, de l'analyse aux tests en passant par la conception détaillée. Il est utilisé en enseignement comme à large échelle dans l'industrie.

Il est intéressant d'observer pourtant qu'entre sa version 1 et sa version 2 de nouveaux concepts sont apparus (composants, contraintes), de nouvelles notations également. Notons aussi que des propositions d'extension ont été faites pour, par exemple, ajouter des couleurs (Coad *et al.*, 1999) ou introduire des éléments de méthode (Cheesman, Daniels, 2000). Les utilisateurs (en particulier les étudiants et leurs enseignants) savent aussi « tordre » certaines notations ou ajouter quelques annotations pour préciser le sens de telle ou telle entité.

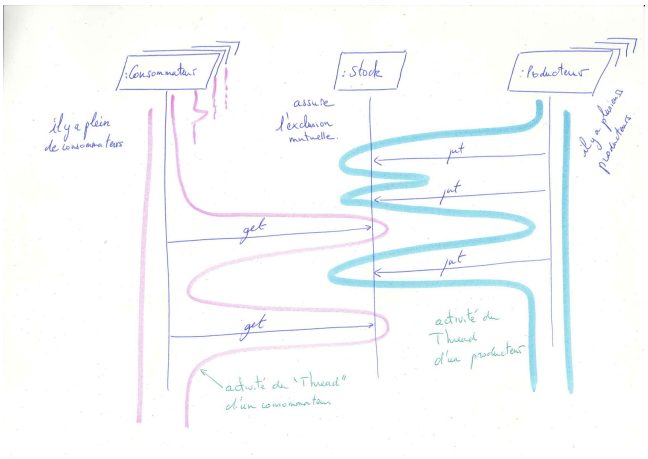


FIGURE 1. *Un diagramme de séquence UML fait à la main et enrichi*

La figure 1 illustre comment un diagramme de séquence UML peut être enrichi par des annotations manuelles, non prévues par les outils, pour montrer la notion de flux d'activité, d'instances multiples afin d'illustrer une exclusion mutuelle. Le papier et le crayon restent de très bons outils de modélisation.

### *2.4. Discussion*

Ces expériences montrent bien que quel que soit le spectre prévu pour un langage, les utilisateurs sont toujours amenés à y apporter leur point de vue et à l'adapter à leurs besoins. Se contenter de ce qu'offre un cadre de modélisation, c'est adopter une façon de penser qui si elle est parfaitement adaptée au besoin peut suffire, mais si elle n'est pas parfaitement adaptée, peut obliger à « tordre » sa façon de penser. Est-ce acceptable ?

La situation est identique avec les langages de programmation et les paradigmes qu'ils traduisent. Certains problèmes se représentent très bien (et se résolvent aussi très bien) en pensant objet, d'autres en pensant fonctionnel, ou synchrone, ou déclaratif etc. Ainsi, la conception d'un langage de programmation relève de compromis et de choix qui forcément réduisent leur spectre d'application pratique (Hoare, 1973 ; Mitchell, 2003).

Rappelons enfin ce que Abraham Maslow (1908 - 1970), psychologue américain, notait : « Quand on a le marteau comme unique outil, tous les problèmes finissent par ressembler à des clous. »

Lors d'activités de modélisation, la solution simple consiste à faire adopter un méta-modèle commun et à y faire adhérer tous les utilisateurs. Cette solution n'est utilisable que dans des cas spécifiques, souvent lorsque les intervenants se connaissent et partagent déjà une culture importante. La plupart du temps, il est nécessaire de construire une interprétation commune, et ce n'est qu'en rencontrant les divergences, qu'on se rend compte des différences. Cette difficulté devient centrale lorsque l'on ne connaît pas exactement les utilisateurs qui vont travailler avec ces modèles, situation courante dans les entreprises.

Il convient également de souligner qu'un langage de modélisation a toujours des limites et n'est donc bien adapté qu'à la production d'une certaine classe de modèles. Pour choisir un langage de modélisation, il faut donc bien connaître, par avance, la forme des modèles que l'on va vouloir produire. Cela rend donc difficile le choix *a priori* d'un langage lorsque l'on ne connaît pas bien le domaine des systèmes que l'on souhaite modéliser.

Ainsi, dans le cas des modèles les plus formels, la sémantique est parfaitement définie, et les utilisateurs devront s'y plier. Par exemple un compilateur impose le sens du programme et peut, à l'exécution, provoquer la surprise d'un utilisateur si celui-ci n'en partage pas l'interprétation. Tout changement de compilateur voire de version du compilateur peut également provoquer des surprises. Dans tous les cas, le compilateur devient le garant de l'interprétation sur lequel les utilisateurs peuvent se reposer pour, éventuellement en expérimentant, vérifier l'(ou les) interprétation(s) possible(s) d'un modèle (le programme). Pour des langages de modélisation dont la sémantique est moins formalisée ou outillée, cet arbitre d'interprétation n'est pas disponible.

Lorsqu'on aborde un nouveau problème, ou lorsqu'on rencontre de nouveaux interlocuteurs, le processus de modélisation devient itératif. Partant d'un choix de repré-

sentation donné (issu de l'histoire ou des connaissances des personnes impliquées), si l'interprétation est commune [3] l'objectif est atteint, sinon, il faut faire évoluer l'interprétation. Deux cas se présentent : soit il est possible, avec la représentation choisie, de converger vers une nouvelle interprétation, soit c'est impossible. Dans le second cas, il est nécessaire de faire évoluer la représentation soit en faisant évoluer la représentation (nouvelles couleurs, nouveaux symboles, nouvelles structures grammaticales, nouveaux mots, etc), soit en créant une nouvelle représentation et en la reliant (et dans la représentation et dans l'interprétation) à la représentation initiale.

Pour un modélisateur, la situation idéale est celle où des méta-modèles existent déjà pour modéliser. Nous avons montré avec ces trois exemples que les situations sont nombreuses où les méta-modèles existants ne suffisent pas. Dans ces cas, nous proposons une identification de situations de modélisation élémentaires où le modèle et son méta-modèle (et leurs représentations) sont co-construits.

## 3. Cas d'étude : le cas d'une collectivité territoriale

Notre cas d'étude, dont le contexte est celui de l'exemple 2.1, est typiquement un de ceux où la situation de départ est une page blanche en termes de méta-modélisation.

Aucun de nos interlocuteurs au sein de la collectivité n'avait connaissance des principes de l'IDM, une minorité d'entre eux avait déjà utilisé des outils de modélisation (traitement accoustique pour l'une, et modélisation de processus pour l'autre), et une seule personne avait déjà pratiqué un moteur de base de données relationnelles.

Dans ce type de contexte, il est difficile d'imposer un langage de modélisation déjà existant. D'autant plus qu'il nous a fallu intégrer d'autres contraintes telles que : l'existence au sein de la collectivité d'un vocabulaire partagé (bien que non formalisé), et le souhait de réutiliser les chartes graphiques existantes.

Les données à notre disposition étaient hétérogènes, tant par la forme que par la provenance :

– Un tableur contenant les données de cartographie des activités ;

– Une représentation des mêmes données sous forme de carte mentale (*Mind-map*) ;

– Le document présentant les objectifs stratégiques (données non structurées) ;

– Quelques planches de diagrammes libres présentant le cadre d'utilisation ;

– Quelques exemples de documents faisant usage de ces données : rapports de cadrage, rapports d'activité, support de vulgarisation/valorisation des actions de la collectivité, ...

3. Cette situation est difficile à détecter de manière absolue. On utilise un consensus supposé, des croyances... qu'il faut parfois savoir remettre en cause.

L'équipe en charge du projet a émis le souhait qu'un maximum de cet existant soit réutilisé pour éviter d'introduire de trop grandes modifications d'usage au sein de l'organisation. Plusieurs objectifs étaient fixés pour le projet :

– Faciliter la maintenance des données et leur appropriation par les producteurs ;
– Aider à l'alignement des *objectifs stratégiques* avec les *activités* ;
– Produire des représentations graphiques à destination des différents publics ;
– Ouvrir à de nouvelles utilisations, par croisements avec d'autres données.

Nous avons donc travaillé à l'élaboration d'un ensemble d'outils en nous appuyant sur les capacités de l'infrastructure Openflexo (introduite dans la partie 6). Celle-ci a l'avantage de fournir les moyens de capturer et d'interpréter dynamiquement des (méta-)modèles conceptuels et des modèles de représentation (définitions de diagrammes ou formulaires). Cela nous a permis, au cours de nos séances de travail, de co-construire et de tester nos modèles avec l'équipe de la collectivité.

Les trois exemples qui suivent, illustrent des situations de modélisation que nous avons rencontrées au cours de ces travaux. Ils correspondent à une phase d'analyse dans laquelle nous avons fait émerger, à partir des documents fournis, les concepts constitutifs de la cartographie.

Certains concepts sont apparus assez simplement depuis leur représentation dans les documents fournis. Comme illustré en figure 2, le travail effectué dans ce cas consistait à :

1. Isoler dans les exemples une représentation qui faisait consensus ;
2. Identifier le concept ;
3. Produire des diagrammes d'exemple pour valider les choix réalisés.

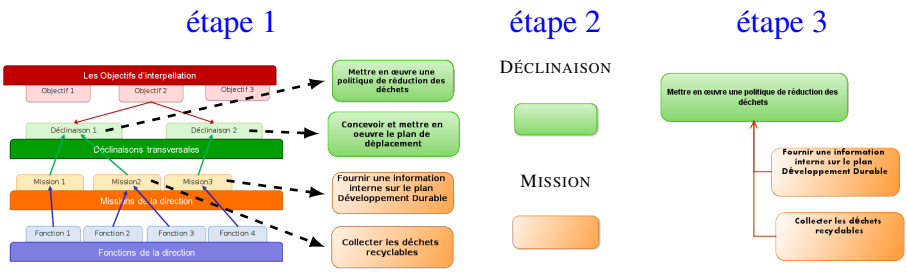


FIGURE 2. *Émergence d'un concept depuis un exemple*

Au cours des discussions, certains concepts sont aussi apparus sans représentation directe. La figure 3 illustre ce cas :

1. Identification d'un nouveau concept au cours des échanges sur un document ;
2. Définition d'une représentation pour ce nouveau concept ;
3. Création d'un diagramme d'exemple pour validation.

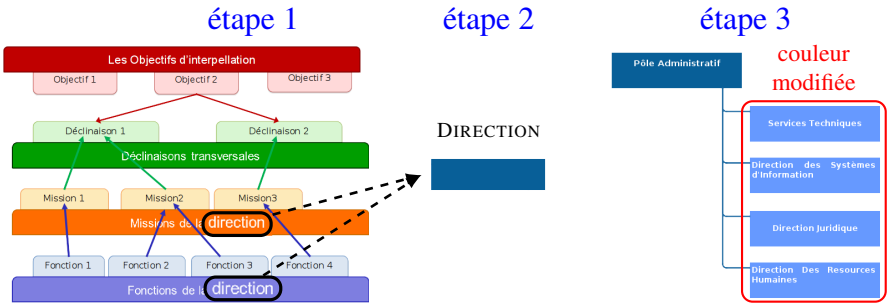


FIGURE 3. *Émergence d'un concept sans représentation (Direction)*

Les diagrammes d'exemples produits par les utilisateurs nous ont parfois servis à identifier des concepts qui n'étaient pas apparus en premier lieu. Comme sur la figure 3, il est arrivé qu'ils changent la couleur d'une forme particulière, distinguant par là même un nouveau concept et invalidant le modèle conceptuel.

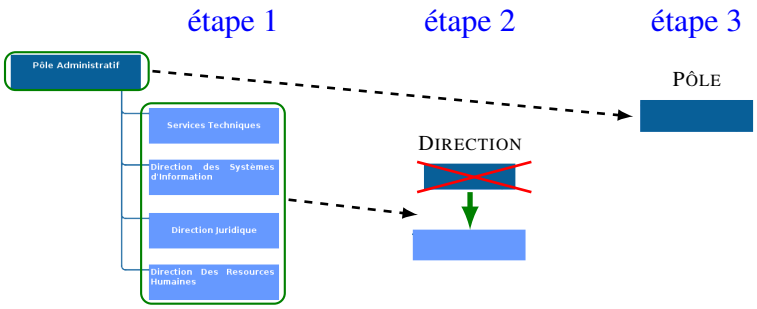


FIGURE 4. *Correction du modèle conceptuel depuis un diagramme d'exemple (introduction de Pôle)*

Dans ce type de cas, la méthode que nous avons appliquée pour revoir les modèles est illustrée sur la figure 4 :

1. Différencier les deux représentations ayant du sens ;
2. Associer une nouvelle représentation au concept pré-existant (ici [*direction*]) ;
3. Construire un nouveau concept à partir de l'autre représentation (ici [*pôle*]).

En complément du travail sur le modèle conceptuel et ses représentations graphiques, nous avons également dû projeter les concepts identifiés sur les bases de données (tableurs) utilisées par la collectivité. Cette phase nous a ainsi permis de préciser les règles d'interprétation et de production des données. L'interprétation de ces modèles de règles par l'outillage permet ensuite la manipulation des données des tableurs par l'intermédiaire de la représentation graphique, ainsi que la génération de diagrammes à partir des tableurs.

Au terme de l'expérimentation avec la collectivité, nous sommes parvenus à construire un environnement et une méthode de travail qui permettent aux membres de l'équipe d'être plus autonomes pour faire évoluer leurs outils. Ce processus de modélisation collaborative a été bien appuyé par l'outillage que nous développons.

Lors de cette étude nous n'avons pas utilisé les termes de *modèle* ou de *méta-modèle*, mais plutôt d'*exemples* et de *concepts*. Dans la partie suivante nous nous appuyons sur ce cas pour identifier des situations de modélisation élémentaires.

## 4. Situations de modélisation

Pour décrire les situations de modélisation, nous avons deux types d'artéfacts à considérer : les modèles et les méta-modèles. Les situations varient selon l'ordre dans lequel ces artéfacts apparaissent ou sont reliés dans la démarche. Nous simplifions la description en ne considérant qu'un seul acteur dans chaque situation. Une analyse plus fine de ces situations pourrait être intéressante en prenant en compte différents acteurs et donc différentes intentions dans le processus. Les figures 5, 6 et 7 présentent les 12 situations considérées, une flèche indique le ou les éléments qui apparaissent. Nous utiliserons aussi les mots *instance* pour décrire ce qui se trouve au niveau modèle et *concept* pour ce qui est au niveau méta-modèle. Chacune des situations est décrite en détail et illustrée dans une des sous-sections suivantes par un exemple concret.

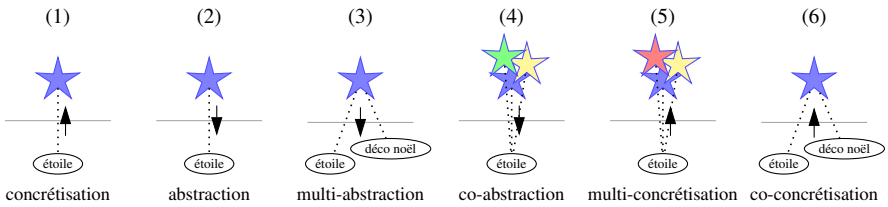


FIGURE 5. *Des situations de modélisation (haut : niveau modèle, et bas : niveau méta-modèle)*

(1) Un méta-modèle existe, le travail consiste à produire un modèle [*concrétisation*].

(2) Un modèle existe, le travail consiste à trouver un méta-modèle [*abstraction*].

(3) Un modèle existe, il faut trouver plusieurs méta-modèles [*multi-abstraction*].

(4) Des modèles existent, le travail consiste à trouver un méta-modèle [*co-abstraction*].

(5) Un méta-modèle existe, le travail consiste à construire plusieurs modèles [*multi-concrétisation*].

(6) Des méta-modèles existent, le travail consiste à construire un modèle [*co-concrétisation*].

(7) Un modèle et un méta-modèle existent, il faut les relier [*interprétation*].

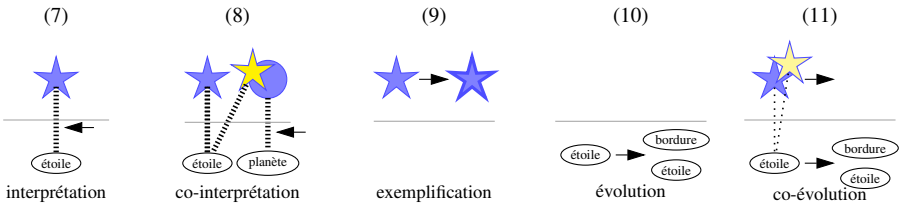


FIGURE 6. *Des situations de modélisation (haut : niveau modèle, et bas : niveau méta-modèle)*

(8) Des modèles existent, des méta-modèles existent, le travail consiste à les relier [*co-interprétation*].

(9) Un modèle existe, le travail consiste à construire un autre modèle (sans aucun méta-modèle) [*exemplification/extension*].

(10) Un méta-modèle existe, le travail consiste à construire un autre méta-modèle (sans aucun modèle) [*évolution/extension*].

(11) Un méta-modèle existe avec plusieurs de ses modèles conformes, le travail consiste à faire évoluer le méta-modèle (cas précédent) en adaptant (ou non) ses modèles [*co-évolution*].

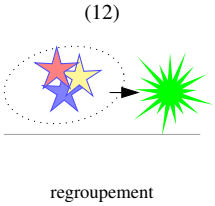


FIGURE 7. *Une situation de regroupement (niveau modèle)*

(12) Un ensemble de modèles est regroupé pour former un tout [*regroupement*].

Les cas (9) et (10) sont probablement équivalents dans le cadre d'une interprétation multi-niveaux de la modélisation. En effet, à un niveau donné d'abstraction, l'absence de référence à un autre niveau, plus concret ou plus abstrait, rend le travail équivalent. Nous les différencions tout de même car la plupart des outils proposent des moyens de manipulations de ces deux niveaux très différents, liés à leur mode de représentation.

Le cas (12) permet de définir des niveaux de détail dans le modèle indépendamment du méta-modèle. Cette situation permet par exemple de mettre en relation ; on regroupe un concept (qui joue le rôle de relation) avec d'autres concepts qui jouent d'autres rôles dans cette relation. Nous ne considérons cette situation qu'au niveau modèle, par glissement [4] elle serait applicable au niveau méta-modèle.

4. Voir le 5.2 pour une définition.

Nous illustrons et commentons ces différentes situations dans les sections suivantes.

### *4.1. Concrétisation*

La concrétisation est la forme d'utilisation la plus classique d'un outil de modélisation. Les concepteurs de l'outil ont préparé des éditeurs adaptés à un ensemble de concepts (méta-modèle). Les utilisateurs élaborent des modèles en respectant les règles prévues. Cette approche est très efficace lorsque le modèle que l'on souhaite construire est adapté au méta-modèle utilisé, i.e. les concepts nécessaires sont présents. Dans le cas où les concepts ne s'alignent pas exactement, les utilisateurs sont parfois amenés à tordre les interprétations ou imaginer des usages des concepts non prévus.

EXEMPLE 1. — Dans le cadre de notre cas d'étude, cette situation correspond à la production de diagrammes d'exemple, après qu'une représentation a été choisie pour chaque concept : la concrétisation produit alors une forme graphique. □

### *4.2. Abstraction*

Ce cas, très classique également dans une démarche de modélisation, consiste, à partir d'un exemple, à identifier les concepts et les règles pour construire un méta-modèle auquel le modèle initial est conforme. Le résultat est exploité par des développeurs d'outil pour construire des éditeurs de modèles.

EXEMPLE 2. — Dans notre cas d'étude, cette situation est illustrée sur la figure 2. □

### *4.3. Multi-abstraction*

Ce cas, moins classique, généralise la situation d'abstraction, où le but est, à la fois d'abstraire, mais également d'organiser l'abstraction en faisant apparaître des points de vue complémentaires sur le modèle (2 dans la situation 3 de la figure 5). Nous ne différencions pas les méta-modèles, l'un peut pré-exister ou ils peuvent être conçus tous en même temps. L'important est qu'il existe plusieurs méta-modèles.

EXEMPLE 3. — Dans le cas d'étude, la situation illustrée par la figure 4 a produit une multi-abstraction. En effet, les concepts de [*Pôle*] et de [*Direction*] se recouvrent partiellement. Un [*Pôle*] étant essentiellement une [*Direction*] qui en regroupe plusieurs. □

### *4.4. Co-abstraction*

Cette pratique est une généralisation de l'abstraction présentée en 4.2. Plusieurs modèles servent d'exemples pour construire un méta-modèle auquel tous les exemples seront conformes. Les démarches taxinomiques ou de classification sont des situations

de co-abstraction. L'intérêt d'utiliser plusieurs modèles exemples est de confronter les concepts et de faire apparaître des consensus ou, au contraire, des différences d'interprétation et des conflits conceptuels.

Les situations de co-abstraction et de multi-abstraction sont combinables en une multi-co-abstraction non représentée. Nous discuterons d'« enchaînement » des situations dans la section 5.

EXEMPLE 4. — Dans notre cas d'étude, cette situation s'est présentée à chaque analyse d'une série d'exemples. Par exemple, chaque fois que l'identification d'un concept a nécessité de croiser à la fois les données du tableur et les documents de cadrage fournis. □

### *4.5. Multi-concrétisation*

Ce cas est une généralisation du premier. Le même méta-modèle est utilisé à plusieurs reprises pour produire différents modèles. Le risque est que lorsque les interprétations ne sont pas les mêmes pour produire les modèles, des incohérences peuvent apparaître sans être détectables. Le risque de cette approche est d'autant plus important que les interprétations du méta-modèle sont nombreuses comme c'est le cas avec UML et ses multiples points de variation sémantique (OMG, 2007). Par exemple, la sémantique des diagrammes d'états n'est pas définie précisément en UML et peut donner lieu à diverses interprétations (Chauvel, Jézéquel, 2005).

EXEMPLE 5. — Lors du cas d'étude, la tentative de validation du concept [Direction] (figure 3) a utilisé de la multi-concrétisation pour mettre en évidence un défaut du modèle. □

### *4.6. Co-concrétisation*

Cette situation se rencontre lorsque plusieurs experts, chacun disposant de son propre méta-modèle, se réunissent pour décrire un système dans lequel leurs points de vue doivent être représentés. Dans cette situation on *souhaite* laisser les abstractions séparées et ne pas construire un méta-modèle commun.

EXEMPLE 6. — Nous n'avons pas rencontré cette situation dans le cas d'étude. □

### *4.7. Interprétation*

Ce cas, s'il est moins habituel, participe sans aucun doute à la démarche de modélisation. Un modèle exemple existe ainsi qu'un méta-modèle. Cette situation est courante et s'appuie sur la connaissance des concepts et de leur organisation par un expert. Le travail consiste à relier un ou des éléments du modèle exemple à un concept présent dans le méta-modèle dans le but :

– soit de s'assurer que l'exemple respecte les règles du méta-modèle ;

– soit de faire apparaître un contre-exemple (le modèle exemple) au méta-modèle afin de le faire évoluer.

EXEMPLE 7. — Dans l'étude de cas, après l'identification du concept de [Direction] sans représentation dans un document fourni, ce concept aurait pu être reconnu dans un autre document ; on aurait alors interprété ce nouvel exemple comme le concept [Direction]. □

Il faut noter que la relation d'interprétation n'est pas la relation d'instanciation. L'instanciation repose sur un mécanisme de construction d'une représentation particulière. Un même concept peut être instancié de multiples manières. L'interprétation évoque l'intention de mettre en relation une représentation et son sens, défini et décrit dans un méta-modèle.

### *4.8. Co-interprétation*

Ce cas est une généralisation du cas précédent. Plusieurs modèles exemples existent ainsi que plusieurs méta-modèles. Cette situation s'appuie sur la connaissance des concepts et de leurs organisations par un expert. Le travail consiste à relier les éléments des modèles à un (ou plusieurs) concept(s) présent(s) dans les méta-modèles dans le but :

– soit de s'assurer que les exemples respectent les règles des méta-modèles ;

– soit de faire apparaître un contre-exemple aux méta-modèles afin de les faire évoluer.

EXEMPLE 8. — Lors de l'étude de cas, le croisement des points de vue entre *élus* et *services* a donné lieu à plusieurs co-interprétations. □

### *4.9. Exemplification/Extension*

Cette situation est souvent la première rencontrée. Il s'agit de produire des exemples de représentation qui servent de base à la réflexion. Il est possible de partir d'une page blanche ou de précédents exemples. Pour des dessins ou des diagrammes, on choisira des couleurs, des formes, des positions relatives des formes pour représenter le problème. L'excellent article de D. Moody (Moody, 2009) passe en revue de nombreux exemples de représentations graphiques. Une histoire des représentations est présentée par M. Friendly dans (Friendly, 2005 ; Friendly, Denis, 2001). Pour les langages, une variabilité extraordinaire des langues et grammaires est observable dans le dictionnaire des langues (Peyraube *et al.*, 2010). Il faut noter que l'exemplification est réalisée en l'absence de l'identification d'un concept associé, sans interprétation donc.

EXEMPLE 9. — Lors de notre étude de cas, le point de départ est un ensemble de documents décrivant des exemples. □

### *4.10. Évolution/Extension*

Cette situation, parallèle à la précédente, ne peut être mise en place qu'une fois la conceptualisation réalisée, c'est-à-dire lorsqu'un méta-modèle est disponible. Il s'agit de faire évoluer les concepts ou les règles les gouvernant. Ce travail se réalise souvent - explicitement ou non - en référence à des évolutions des modèles.

EXEMPLE 10. — Plusieurs fois lors de l'étude de cas, nous avons fait évoluer les modèles et les méta-modèles. □

### *4.11. Co-évolution*

C'est une situation liée à la précédente, où le méta-modèle évolue et est relié à des modèles existants ; qu'advient-il des modèles existants ? L'article (Sprinkle, Karsai, 2004) est l'un des premiers à identifier le problème avec le vocabulaire de l'ingénierie dirigée par les modèles.

Ce cas, complexe, peut prendre de nombreuses formes. Par exemple, si un nouveau concept est introduit dans le méta-modèle sans qu'aucune instance de ce concept n'ait été précédemment créée, la solution est triviale. Par contre, si la structure d'un concept est remise en cause et que des instances avaient été créées, la solution pourrait ne pas être automatisable et demander l'intervention d'un humain pour guider l'évolution.

EXEMPLE 11. — Lors de notre étude, nous ne sommes pas allés jusqu'à faire évoluer les bases de données de la collectivité suite aux évolutions de la (méta-)modélisation. □

### *4.12. Regroupement*

Cette situation est très fréquente et fondamentale. C'est une sorte d'exemplification [5] (situation 9) qui permet de montrer une nouvelle instance par regroupement. L'exemple ainsi identifié pourra, si besoin être interprété (situation 7) si le concept du groupe est reconnu ou abstrait (situation 2) si le concept du groupe est à inventer. Cette opération reste au niveau instance.

Par le mécanisme de regroupement, il est possible d'englober toutes les mises en relation (au sens entité-relation). Il suffit que, lors de chaque regroupement, soient identifiés les *rôles* que doivent (ou devraient s'il n'existent pas encore) jouer les instances dans la relation.

EXEMPLE 12. — Dans l'étude de cas, les directions ont des missions et sont décomposés en pôles. La mise en évidence de ces relations sont des regroupements. □

5. Mais sans la dimension notation.

### *4.13. Synthèse*

Les douze situations présentées se regroupent en cinq grandes catégories :

**Du concept à l'instance.** C'est l'approche la plus fréquemment *outillée*. La boîte à outil apporte son lot de concepts qu'il s'agit de concrétiser pour modéliser. Le méta-modèle pré-existe, le modélisateur s'y réfère. Certains outils permettent quand la concrétisation est impossible d'agrandir l'ensemble des concepts pour ensuite concrétiser. L'instance n'existe pas sans le concept.

**De l'instance au concept.** C'est l'approche la plus souvent *utilisée*, avant le passage à la catégorie précédente. On dessine sur un tableau, une feuille de papier ou dans un outil de dessin. On identifie les concepts, puis on change d'outils et on passe des concepts aux instances.

**Reconnaissance d'une interprétation.** L'approche intermédiaire permet l'existence indépendante de concepts et d'instances, pour finalement reconnaître leur lien d'interprétation.

**Composition.** Indépendante du lien d'interprétation, la composition s'applique aussi bien aux instances qu'aux concepts. Elle permet d'introduire la notion de composition qui permet de représenter des associations, des relations, du contenu, etc.

**Évolution.** Les évolutions mettent en évidence la dynamique de construction des deux niveaux impliqués : instances et concepts. Elles peuvent être indépendantes du lien d'interprétation ou pas.

## 5. Diverses façons d'enchaîner

Une fois des situations de modélisation identifiées, la question de leur enchaînement [6] se pose naturellement. Nous ne ferons qu'illustrer quelques exemples relevant de ce large problème. En particulier, nous ne considérons pas les cas de modélisation coopérative où plusieurs personnes pourraient enchaîner des situations et où il faudrait gérer la synchronisation de leurs actions.

### *5.1. Enchaînement comme séquence de situations*

La situation de multi-concrétisation (respectivement multi-abstraction) peut être vue comme une composition de plusieurs concrétisation (resp. abstraction). Ce n'est pas nécessairement le cas, car le fait de regrouper plusieurs actions dans la même situation n'est pas exactement la même situation que de reproduire *indépendamment* plusieurs fois l'action.

6. Nous n'utilisons pas le terme composition, mais le terme enchaînement qui met plus en évidence la dimension temporelle.

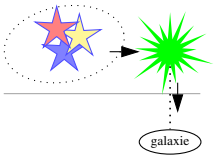


FIGURE 8. *On enchaîne un regroupement et une identification de concept*

On peut également enchaîner des situations différentes. La figure 8 montre un enchaînement classique de deux situations où l'on commence par regrouper (12) un ensemble de modèles pour l'abstraire (2) en un concept.

### *5.2. Glissement comme empilement d'abstractions*

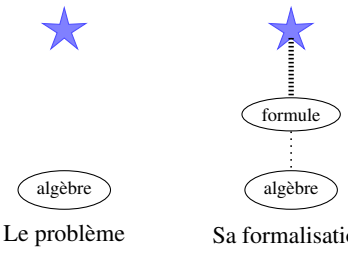


FIGURE 9. *Un enchaînement pour représenter une approche algébrique*

Un modèle peut parfois être interprété comme un méta-modèle et conduire à la création de modèles de « niveaux » différents. Inversement, un méta-modèle peut aussi être considéré comme la concrétisation d'un méta-méta-modèle. Nous appelons ce changement de point de vue un glissement. Il est donc possible d'empiler les situations de concrétisation, abstraction ou interprétation qui sont des suites de glissements.

La figure 9 montre une formulation de la démarche d'algébrisation en 2 étapes. La première (à gauche) consiste à identifier un problème (le modèle) et une théorie (le méta-méta-modèle). La seconde étape (partie droite de la figure 9) consiste à produire une formule qui représente le problème qui est une concrétisation de la théorie et une interprétation du modèle pour le problème considéré. On voit ici qu'on combine (et empile) deux situations élémentaires : concrétisation et interprétation.

Les situations 12 de regroupement, 9 d'exemplification et 10 d'évolution s'appliquent à un seul niveau. Il est naturel de pouvoir les considérer sur un autre niveau après avoir changé de point de vue par glissement.

### *5.3. Regroupement multi-niveaux*

On pourrait chercher à interpréter les interprétations comme des regroupements entre des concepts de niveaux différents. Certes, mais leur nature profondément an-

crée dans la sémantique de la modélisation nous les fait considérer à part ; elles participent à la définition du sens de ce qu'est la modélisation. C'est pourquoi nous ne considérons que les liens de concrétisation (1), d'abstraction (2) et d'interprétation (7) entre ces deux niveaux sans chercher à plus abstraire (ou généraliser) par une situation de regroupement entre concepts plus abstrait encore.

### *5.4. Enchaînement de regroupements*

Nous ne présentons qu'un seul axe de regroupement. C'est un cas particulier. Il existe dans le cas général de nombreux axes possibles de regroupement où le même élément de modèle pourrait appartenir à plusieurs regroupements. Des outils qui mettraient en pratique cette vision de la modélisation devraient aussi gérer des informations concernant les « logiques » de regroupement afin d'aider l'utilisateur. Parmi les informations de regroupement, les représentations (langages) et règles métiers (contraintes) sont également essentielles.

## 6. Outillage

Afin de valider la pertinence de nos situations de modélisation, et supporter notre cas d'étude, nous avons conçu un outillage qui s'appuie sur une infrastructure logicielle développée par la société Openflexo [7]. Les situations présentées précédemment sont, sur le principe, toutes réalisables en utilisant cette infrastructure. Dans les faits, nous n'avons exploité que 10 des 12 situations décrites. Les situations 11 et 12 (coévolution, regroupement) nécessitent des évolutions du logiciel pour pouvoir être totalement abordées.

Dans la suite de cette partie, nous présentons l'environnement de *modélisation libre* qui a été utilisé. Ensuite nous décrirons comment il a été mis en œuvre sur le cas d'étude, illustrant concrètement ainsi quelques unes des situations de modélisation.

### *6.1. L'outil de « modélisation libre »*

Le *« Free Modeling Editor »* (FME) est un prototype destiné à l'expérimentation de nouveaux modes d'interaction avec les modèles. Il est utilisé pour faciliter l'émergence d'une syntaxe graphique simultanément au modèle conceptuel associé.

Son interface, illustrée par la figure 10, est découpée en une partie *outils de dessin* sur la droite et une partie *modèle conceptuel* et *instances* sur la gauche. L'utilisateur peut librement dessiner dans l'espace au centre en plaçant sur l'espace de travail des formes graphiques sélectionnées depuis une des palettes présentées à droite.

Deux navigateurs de concepts sont proposés sur la gauche de l'interface. Les instances apparaissent catégorisées sous le nom du concept dans le premier navigateur.

7. openflexo.org

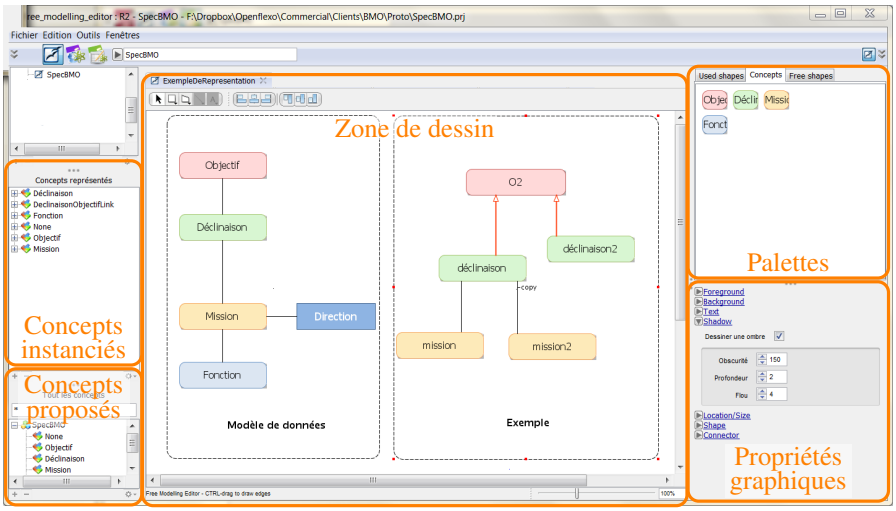


FIGURE 10. *L'organisation du «Free Modeling Editor»*

Les formes graphiques (instances) non encore associées à un concept apparaissent sous le terme *None*. L'ensemble des concepts définis apparaît dans le second navigateur.

Il est possible de créer un concept à l'aide d'un menu contextuel proposé dans le second navigateur. Dans ce cas, le nouveau concept n'est associé à aucune forme graphique. L'association des formes graphiques avec les concepts se fait à partir de la zone de dessin. Un menu contextuel accessible à partir d'une forme sélectionnée permet cette association. Si le concept correspondant n'existe pas encore, l'utilisateur peut en définir un nouveau.

Pour la création des formes graphiques, trois types de palettes sont proposés à l'utilisateur :

1. Une palette permettant la ré-utilisation des formes déjà définies dans le dessin en cours d'édition (« Used shapes ») ;
2. Une palette proposant les formes déjà associées à des concepts (« Concepts ») ;
3. Des formes simples prédéfinies (« Free shapes »).

Le comportement de l'outil diffère en fonction de la palette utilisée :

– Le choix d'une forme depuis la palette « Concepts » déclenche la création d'une forme graphique sur le dessin, et d'une nouvelle instance du concept concerné, reliée à la forme graphique ;

– Le choix d'une forme prédéfinie ou la réutilisation d'une forme existante, ne déclenche que la création d'une forme graphique.

Chaque création de forme dont les propriétés graphiques varient par rapport à celles déjà présentes sur le dessin provoque l'apparition d'une nouvelle entrée lui correspondant dans la palette contextuelle « Used shapes ». Les propriétés graphiques des formes utilisées sont toutes ajustables. Il est également possible d'utiliser des images fournies par les utilisateurs.

Il faut noter que l'association d'une forme graphique avec une instance n'est que partiellement prescriptive dans le *FME*. En effet, lorsqu'une instance est créée avec une forme prescrite (celle qui est associée au concept), l'utilisateur peut toujours la modifier. Dans ce cas, la forme associée à l'instance diffère de celle associée au concept. Cette situation a été rencontrée sur le cas d'étude et est illustrée sur la figure 4. Des outils sont également mis à disposition pour restaurer les formes prescrites, ou, au contraire, remplacer la forme associée à un concept et l'imposer à l'ensemble des autres instances.

Etant donné la souplesse de l'outil, il est facile d'expérimenter les différentes situations de modélisation décrites ci-avant avec le *FME*. La séparation des espaces de travail et des responsabilités conduit à un fonctionnement de l'atelier qui permet de pratiquer les 10 premières situations décrites. Dans la section suivante, nous décrivons la façon dont il a été mis en œuvre sur le cas d'étude, en illustrant les situations de modélisation rencontrées.

### *6.2. Mise en œuvre sur les situations de modélisation*

L'outil de *modélisation libre* a été au cœur de la méthode de modélisation présentée en 3. Les premiers diagrammes utilisés ont été initialisés grâce à la fonctionnalité d'import de planches *Powerpoint* proposée par l'outil. Dans un premier temps il a donc été utilisé comme un simple outil de dessin, pour :

– ré-utiliser les exemples fournis par la collectivité,

– faire émerger l'ensemble des représentations permettant d'illustrer le problème.

Dans ce cas, ce sont essentiellement les situations d'exemplification (4.9) et de regroupement (4.12) qui ont été activées. Les diagrammes ont été retravaillés par les utilisateurs jusqu'à ce que les représentations conviennent, situation d'évolution décrite en (4.10) . Cela a également conduit à la suppression de certains éléments, situation non abordée dans notre travail d'identification des situations.

Dans une seconde étape, partant des diagrammes obtenus, nous avons pu faire émerger l'ensemble des concepts en utilisant le support du *FME* pour toutes les situations d'abstraction (abstraction 4.2, multi-abstraction 4.3 et co-abstraction 4.4).

La phase de validation de cette seconde étape a nécessité l'utilisation d'un autre outil de l'infrastructure Openflexo: l'*Éditeur de vues*. Celui-ci permet la création de nouveaux diagrammes d'exemple à partir des modèles conceptuels et de représentation produits par le *FME*. Les nouveaux diagrammes produits sont donc conformes

aux spécifications établies (concepts et représentations). Les situations de modélisation invoquées ici sont la concrétisation (4.1) ainsi que la multi-concrétisation (4.5).

Comme illustré sur la figure 4 du cas d'étude, il a parfois fallu faire évoluer le modèle conceptuel à la suite de cette première phase de validation. Les situations d'évolution (4.10) et d'interprétation (4.7) ont été utilisées dans ce cas :

– Évolution de la représentation : changement de la couleur d'une forme graphique ;

– Évolution du modèle conceptuel : identification d'un nouveau concept ;

– Interprétation de l'instance dont la représentation a été changée comme une instance du nouveau concept.

Il a également parfois été nécessaire d'aller au-delà des fonctionnalités du *FME* [8] pour répondre aux demandes des utilisateurs sur des situations telles que la co-évolution (4.11). Cependant, il a été possible d'aller jusqu'au bout de la démarche, et de disposer d'un modèle conceptuel et de différentes spécifications de diagrammes que nous avons pu ré-utiliser dans la suite du processus.

## 7. Travaux connexes

Nous abordons les travaux connexes selon deux points de vue. Le premier concerne des travaux sur des situations de modélisation qui sont étudiées par quelques études empiriques. Le second est rattaché à des réflexions sur la nature même de la modélisation.

Dans (Hermans *et al.*, 2009), les auteurs étudient les facteurs de succès de l'utilisation de DSL. Dans un cadre très limité – les services de communication WCF (Windows Communication Foundation) – les auteurs ont étudié 30 projets utilisant ACA.NET [9]. L'étude porte sur l'intérêt d'utiliser un DSL et non pas sur la façon d'en définir, même si les facteurs étudiés comme l'utilisabilité, l'expressivité ou la réutilisabilité y sont liés. L'article conclut à un intérêt des DSL pour la productivité et la fiabilité et des suggestions pour améliorer la réutilisabilité.

À l'inverse, dans (Kelly, Pohjonen, 2009), Kelly et Pohjonen présentent les écueils principaux de la modélisation spécifique à un domaine (DSM) à l'aide de 75 projets dans des domaines très différents (de l'avionique, du médical, de la configuration de système, etc.). Les difficultés rencontrées comme « Seuls les gourous peuvent », « Outil : Si vous avez un marteau… » ou « Intouchable dès la création » nous renforcent dans l'idée de l'importance des situations de modélisation originales mises en évidence (exemplification, interprétation, multi-abastraction) et que la (méta)-modélisation est un processus vivant et itératif.

8. En intervenant sur les modèles manipulés par le *FME* par l'intermédiaire des autres outils disponibles au sein de l'infrastructure.

9. http://www.avanade.com/delivery/acanet/

Un autre écueil, présenté dans (Wouters, 2013), concerne la difficulté à partager avec l'ensemble des acteurs la définition du langage. Il est souvent plus efficace de passer par des exemples et des éléments de représentation (graphique, texte ou données semi-structurées selon les habitudes des interlocuteurs) pour établir une compréhension commune des concepts sous-jacents. Ce constat est d'ailleurs partagé par des approches telles que (Canovas Izquierdo, Cabot, 2012), qui proposent à la communauté d'utilisateurs d'un langage de le concevoir et le faire évoluer par l'intermédiaire de représentations stéréotypiques.

Concernant les réflexions sur la modélisation, les travaux de J.-M. Favre (Favre, 2004 ; Favre, Nguyen, 2005) posent les bases des relations entre modèles et méta-modèle en introduisant les relations $\delta$ de liens de composition, $\mu$ de modélisation, $\epsilon$ de liens entre une instance et l'ensemble des instances possibles, $\chi$ de conformité, et $\tau$ de trace. On peut voir dans les situations que nous présentons des raffinements de $\tau$, certaines liés à la conformité $\chi$ (situation 1 à 5, 7, 9 et 10), d'autres à la composition $\delta$ (situation 12). Nous mettons hors du champs les relations de cet article $\mu$ et $\epsilon$. Les situations 6 et 8 ne sont pas envisagées. La situation 11 de co-évolution est présentée comme une composition de $\mu$ (ou de $\chi$) et de $\tau$.

Dans (Muller *et al.*, 2010) les auteurs étudient d'autres relations entre modèles en formalisant différentes intentions dans le but de « faciliter le raisonnement et la discussion sur les méta-modèles et leurs relations » et de servir de base au développement d'ateliers de méta-modélisation « conscients » des intentions.

C. Atkinson *et al.* (Atkinson *et al.*, 2009) met en évidence la différence de sens entre l'abstraction conceptuelle et l'abstraction syntaxique (des langages) pour proposer une approche de modélisation multi-niveaux. Nous retrouvons cette distinction dans notre découplage entre le modèle et sa représentation puis entre le modèle et son méta-modèle. Notre description autorise une modélisation multi-niveaux selon les deux interprétations de l'abstraction.

Dans (Thalheim, 2010 ; 2012), B. Thalheim étudie la notion de modèle et propose une formalisation. Son approche considère un modèle comme un artefact fini dont il décrit les propriétés et les relations avec ce qu'il représente et celui qui l'interprète. Notre vision de la notion de modèle est plus liée à la dynamique de construction de la modélisation.

## 8. Conclusion

La modélisation est une démarche complexe, avant tout intellectuelle, mais qui peut être outillée. Le papier et le crayon (ou leur équivalent) restent des outils de base grâce à leur flexibilité et à la liberté qu'ils offrent. L'outillage informatique apporte des fonctionnalités supplémentaires qui permettent de vérifier des propriétés telles que la conformité. Pourtant, cet apport peut se faire au détriment de la liberté. Comment concilier vérification et liberté ? L'identification de situations de référence de modélisation peut servir de spécification d'usage pour des ateliers de modélisation. Nous

avons observé :

– qu'il ne faut pas réduire la modélisation au dessin ; la notion de méta-modèle apporte des outils de vérification et peut guider lors de la concrétisation ;

– qu'un méta-modèle se construit et doit pouvoir évoluer ; on doit pouvoir aller au-delà de ce qui est prévu ;

– que méta-modèle et modèle ne sont pas nécessairement liés a priori, mais que c'est une activité de modélisation que d'identifier ces liens.

Modéliser requiert toutes les situations, combinées dans des ordres variés. Par exemple, on peut commencer par *exemplifier*, puis *co-abstraire* et *interpréter*, puis *généraliser*, pour *exemplifier* à nouveau afin de valider le méta-modèle. Un atelier de modélisation doit permettre un travail à tous ces niveaux : intra-niveau et inter-niveaux.

La plupart des outils actuels sont trop contraignants – ils obligent à adopter un méta-modèle (ou un ensemble de méta-modèles) figé – ou trop complexe – et le travail sur le méta-modèle est une activité de programmation qui ramène souvent les concepts aux paradigmes des informaticiens.

L'infrastructure Openflexo propose une approche multi-niveau par assemblage ou fédération de modèles (Koudri *et al.*, 2013). Elle peut être utilisée comme simple outil de dessin, comme outil de modélisation classique, mais aussi comme outil de méta-modélisation pour élaborer un méta-modèle sans programmation et enfin pour mettre en relation – interpréter – des dessins et des méta-modèles.

Pour prolonger ce travail, nous envisageons une analyse plus fine des situations, en prenant en compte par exemple les différents acteurs, mais aussi des situations non pas constructives mais destructives comme la suppression d'une interprétation ou la suppression d'un exemple. D'autres situations sont certainement intéressantes comme le changement de niveau évoqué dans la partie 5.2, quand un modèle devient un méta-modèle ou réciproquement. Enfin, une étude précise des outils de modélisation devrait être entamée au regard des différentes situations de modélisation identifiées.

## Bibliographie


Atkinson C., Gutheil M., Kennel B. (2009, novembre). A Flexible Infrastructure for Multilevel Language Engineering. *IEEE Transactions on Software Engineering*, vol. 35, n° 6, p. 742–755.

Canovas Izquierdo J., Cabot J. (2012, June). Community-driven language development. In *Modeling in software engineering (mise), 2012 icse workshop on*, p. 29-35.

Chauvel F., Jézéquel J.-M. (2005). Code generation from uml models with semantic variation points. In L. Briand, C. Williams (Eds.), *Model driven engineering languages and systems*, vol. 3713, p. 54-68. Springer Berlin Heidelberg.

Cheesman J., Daniels J. (2000). *UML components: A simple process for specifying component-based software*. Addison-Wesley.

Coad P., Lefebvre E., Luca J. D. (1999). *Java modeling in color with uml*. Prentice Hall.

Favre J.-M. (2004). Towards a Basic Theory to Model Model Driven Engineering. *3rd Workshop in Software Model Engineering*.

Favre J.-M., Nguyen T. (2005, avril). Towards a Megamodel to Model Software Evolution Through Transformations. *Electronic Notes in Theoretical Computer Science*, vol. 127, n° 3, p. 59–74.

Friendly M. (2005). Milestones in the history of data visualization: A case study in statistical historiography. In C. Weihs, W. Gaul (Eds.), *Classification: The ubiquitous challenge*, p. 34–52. New York, Springer.

Friendly M., Denis D. J. (2001). *Milestones in the history of thematic cartography, statistical graphics, and data visualization.* Web document, http://www.datavis.ca/milestones/.

Hermans F., Pinzger M., Deursen A. van. (2009). Domain-Specific Languages in Practice: A User Study on the Success Factors. In A. Schürr, B. Selic (Eds.), *Model 2009*, p. 423–437. Berlin, Heidelberg, Springer Verlag.

Hoare C. A. R. (1973). *Hints on programming language design.*. Rapport technique. Stanford, CA, USA.

Kelly S., Pohjonen R. (2009). Worst Practices for Domain-Specific Modeling. *Software, IEEE*, vol. 26, n° 4, p. 22–29.

Kleppe A. (2007, October). A language description is more than a metamodel. In *Fourth international workshop on software language engineering.* Grenoble, France, megaplanet.org.

Koudri A., Guychard C., Guerin S., Beugnard A., Champeau J., Dagnat F. (2013, juin). De la nécessité de fédérer des modèles dans une chaîne d'outils. *Génie logiciel*, n° 105, p. 18 – 23.

Kühne T. (2006). Matters of (Meta-) Modeling. *Software and Systems Modeling*, vol. 5, n° 4, p. 369–385.

Lachapelle C., Cunningham C. (2007, June). Engineering is elementary: Children's changing understandings of engineering and science. In *American society for engineering education annual conference & exposition.* Honolulu, HI.

Mitchell J. C. (2003). *Concepts in programming languages*. Cambridge University Press.

Moody D. (2009). The "physics" of notations: toward a scientific basis for constructing visual notations in software engineering. *IEEE Transactions on Software Engineering*, vol. 35, n° 6, p. 756–779.

Morris W. T. (1967). On the art of modeling. *Management Science*, vol. 13, n° 12, p. B707–B717.

Muller P.-A., Fondement F., Baudry B. (2010). Modeling modeling modeling. *Software and Systems Modeling*.

Peyraube A., Bonvini E., Busuttil J. (2010). *Dictionnaire des langues*. Presses Universitaires de France.

Sprinkle J., Karsai G. (2004). A domain-specific visual language for domain model evolution. *Journal of Visual Languages and Computing*, vol. 15, n° 3, p. 291–307.

Thalheim B. (2010). Towards a Theory of Conceptual Modelling. *Journal of Universal Computer Science*, vol. 16, n° 20, p. 3102–3137.

Thalheim B. (2012). The science and art of conceptual modelling. In *Transactions on large-scale data-and knowledge-centered systems vi*, p. 76–105. Springer.

*UML 2.0 superstructure specification.* (2007). http://www.omg.org/cgi-bin/doc?ptc/2003-08-02.

Wouters L. (2013). Towards the notation-driven development of dsmls. In A. Moreira, B. Schätz, J. Gray, A. Vallecillo, P. Clarke (Eds.), *Model-driven engineering languages and systems*, vol. 8107, p. 522-537. Springer Berlin Heidelberg. Consulté sur http://dx.doi.org/10.1007/978-3-642-41533-3_32