CIGI QUALITA MOSIM 2023
Capitalisation de connaissances en configuration de biens et de services : vers une meilleure gestion de la communalité des modèles

ÉLISE VAREILLES[1], THIERRY COUDERT[2], Michel ALDANONDO[3] , MARYAM MOHAMMAD AMINI[2,1]

[1] Université de Toulouse - ISAE SUPAERO
10 avenue Édouard Belin, 31000 Toulouse, France
prenom.nom@isae-supaero.fr

[2] Université de Toulouse - ENI TARBES
47, avenue d'Azereix - BP 1629 - 65016 Tarbes CEDEX, France
prenom.nom@enit.fr

[2] Université de Toulouse - IMT Mines Albi
All. des sciences, 81000 Albi, France
prenom.nom@Mines-albi.fr

Résumé **– Depuis les années 2000, nous assistons à une explosion des biens et/ou services personnalisés. Anciennement réservée au secteur du luxe, la personnalisation s'étend désormais aux biens de consommation et aux services du quotidien. Dans ce contexte grandissant de personnalisation, les entreprises se doivent de proposer à leurs clients des biens et des services adaptés à leurs besoins spécifiques. Pour se faire, cette personnalisation nécessite la formalisation de la diversité proposée par une entreprise sous la forme de modèle générique de configuration. Chaque famille de systèmes et/ou services voit ainsi sa diversité extraite, validée et formalisée sous forme d'un modèle générique que l'entreprise doit ainsi gérer et maintenir. Notons que l'obtention et la mise à jour de ces modèles génériques requièrent une implication forte de plusieurs experts de l'entreprise (Marketing, Bureau d'études, Production, Maintenance), ainsi que plusieurs itérations. Plusieurs modèles génériques peuvent alors cohabiter au sein d'une même entreprise, multipliant les opérations nécessaires à leurs mises à jour et les erreurs potentielles. Pour éviter cela, nous proposons dans cet article, d'utiliser conjointement les concepts d'ontologies, de problèmes de satisfaction de contraintes, d'héritage et de communalité pour améliorer la génération, la mise à jour et la gestion de ces modèles génériques.**

Mots clés **- Capitalisation de connaissances, Ontologie, CSP, Héritage, Communalité, Configuration**

1 INTRODUCTION

Depuis les années 2000, nous assistons à une explosion des biens et/ou services personnalisés. Anciennement réservée au secteur du luxe, la personnalisation s'étend désormais aux biens de consommation et aux services du quotidien. Dans ce contexte grandissant de personnalisation, les entreprises se doivent maintenant de proposer à leurs clients des biens et des services adaptés à leurs besoins spécifiques. Pour se faire, cette personnalisation nécessite la formalisation de la diversité (système et/ou service) proposée par une entreprise sous la forme de modèles génériques de configuration. Chaque famille de systèmes et/ou services voit ainsi sa diversité extraite, validée et formalisée sous forme d'un modèle générique que l'entreprise doit ainsi élaborer, gérer et maintenir. Notons que l'obtention et la mise à jour de ces modèles génériques requièrent une implication forte de plusieurs experts de l'entreprise (Marketing, Bureau d'études, Bureau des méthodes, Production, Maintenance, etc.), ainsi que plusieurs itérations. Plusieurs modèles génériques peuvent alors cohabiter au sein d'une même entreprise, multipliant les opérations nécessaires à leurs mises à jour et les erreurs potentielles. Pour éviter cela, nous proposons dans cet article, d'utiliser conjointement les concepts d'ontologies, de problèmes de satisfaction de contraintes, d'héritage et de communalité pour améliorer la génération, la mise à jour et la gestion de ces modèles génériques. Dans la suite de cet article, nous faisons référence, sous le vocable *artefact,* indifféremment aux systèmes, sous-systèmes ou composants ainsi qu'aux services, sous-modules et modules [Guillon et al., 2021]. Pour détailler un peu plus nos propositions, les ontologies sont utilisées pour structurer la connaissance sous forme d'*atomes* de connaissances et d'*architectures* de

connaissances basées sur ces atomes. Les problèmes de satisfaction de contraintes ou CSP permettent de modéliser les relations reliant ces connaissances et de garantir leur cohérence au sein d'un atome ou d'une architecture. Le concept d'héritage permet de renforcer les relations de spécialisation / généralisation déjà présentes dans les ontologies, et d'ajouter des propriétés importantes, telles que la transitivité ou le polymorphisme. Le concept de communalité entre artefacts permet d'identifier les atomes et architectures communs à plusieurs modèles génériques. Dans notre proposition, nous définissons donc les éléments nécessaires à la construction d'une ontologie de modèles génériques qui, à notre connaissance, ne fait l'objet d'aucun travail dans la littérature. Le reste de ce document est organisé comme suit. Dans la section 2, nous présentons un état de l'art des travaux relatifs à nos propositions. Ensuite, dans la section 3, nous posons les différents éléments nécessaires à la définition d'une ontologie de modèles génériques pour la configuration. Enfin, en section 4, nous présentons nos conclusions et nos perspectives de recherche.

2 ÉTAT DE L'ART

Dans cette section, nous balayons tour à tour (1) la configuration de produits / systèmes / services, (2) la formalisation / modélisation des connaissances à l'aide d'approches telles que les ontologies et les CSP, (3) le concept d'héritage et de spécialisation / généralisation puis (4) la communalité entre systèmes / services. Nous concluons par notre question de recherche.

2.1 Configuration de produits / systèmes / services

[Soininen et al., 1998] définissent la configuration de produits comme "un problème de conception utilisant un ensemble de composants prédéfinis, ainsi qu'un ensemble de restrictions sur la façon dont les composants peuvent être combinés". [Oddsson et Ladeby, 2014] définissent un modèle générique comme "une représentation ou une description abstraite, décrivant la structure du produit, les entités dont le produit est composé et les règles sur la façon dont les entités et leurs propriétés peuvent être combinées".

La configuration est donc un problème combinatoire qui repose sur un modèle générique. En configuration, deux étapes sont nécessaires : (1) l'étape de formalisation des connaissances, qui conduit à un modèle générique et (2) l'étape de réutilisation des connaissances qui, partant d'un modèle générique et de besoins clients, aboutit à un produit / système / service personnalisé.

Dans cet article, nous nous concentrons uniquement sur l'étape de formalisation des connaissances et l'obtention de modèles génériques.

2.2 Formalisation des connaissances

L'étape de formalisation des connaissances consiste à identifier, extraire, structurer et formaliser des connaissances de manière à faciliter leur interprétation [Felfernig A., Friedrich G., & Jannach D., 2001)]. Dans l'étape de formalisation des connaissances, différents experts de l'entreprise (Marketing, Bureau d'études, Bureau des méthodes, Production, Maintenance, etc.) sont mobilisés afin d'aboutir à des modèles génériques pertinents, cohérents et représentatifs, reprenant l'ensemble des connaissances générées au cours du cycle de vie. Il faut ainsi formaliser les connaissances atomiques relatives aux composants / modules des artefacts, leurs alternatives et variantes, puis leur architecture, i.e. la nomenclature du système / service. [Sabin et Weigel, 1998] mentionnent que la plus grande difficulté dans la formalisation des connaissances est de modéliser différents types de relations entre les connaissances telles que la classification (*is-a*), l'agrégation (*part-of*), et les relations liées à la cardinalité, la géométrie, etc.

Puisque la formalisation des connaissances pour définir des modèles génériques est un point critique, de nombreux efforts ont été réalisés pour supporter cette étape. [Hotz et al., 2014] font la synthèse des différentes approches telles que la représentation des connaissances basée sur les contraintes (CSP, satisfaction dynamique des contraintes, satisfaction générative des contraintes), basée sur des représentations graphiques (modèles de fonctionnalités et modèles en langage unifié de modélisation - UML), ou basée sur la logique (logique du premier ordre, programmation par ensembles de réponses) conduisant à la définition d'un modèle générique. [Soininen et al., 1998] propose une ontologie générale contenant des concepts de modélisation afin de représenter les connaissances dans le domaine de la configuration. [Yang, Dong et Miao, 2008] proposent, quant à eux, une approche basée sur des ontologies pour formaliser la connaissance de la configuration des produits en utilisant le Web Ontology Language et le Semantic Web Rule Language.

Dans le cadre de nos travaux, nous basons nos propositions sur (1) les ontologies afin de structurer la connaissance sous forme d'*atomes* de connaissance et d'*architectures* de connaissances basées sur ces atomes, associées (2) aux problèmes de satisfaction de contraintes ou CSP afin de modéliser les relations reliant ces connaissances et de garantir leur cohérence au sein d'un atome et/ou d'une architecture.

2.2.1 Définition des ontologies

[Studer, Benjamins et Fensel, 1998] définissent une ontologie comme une "*spécification formelle et explicite d'une conceptualisation partagée*". Une ontologie capture les connaissances du domaine afin d'en fournir une compréhension commune. Elle définit des concepts ou des classes, des attributs de classes, des domaines d'attributs, des relations hiérarchiques, des niveaux d'abstraction et d'héritage, des compositions, des règles, des axiomes et des assertions. Les ontologies reposent sur un mécanisme d'inférence ensembliste et permettent des raisonnements inductifs et déductifs, sur des requêtes concernant les connaissances représentées dans l'ontologie, la vérification de la cohérence de la connaissance et l'identification d'anomalies. Les ontologies séparent clairement la base de connaissance et son exploitation.

2.2.2 Définition des CSP

[Montanari, 1974] définit un CSP comme un triplet {X, D, C} où X est un ensemble de variables, D un ensemble de domaines- un pour chaque variable, et C un ensemble de contraintes reliant les variables. Les contraintes représentent des restrictions sur les combinaisons de valeurs des variables, en explicitant les combinaisons de valeurs autorisées ou interdites. Les CSP reposent sur des mécanismes de propagation de contraintes de type arc-cohérence [Mackworth, 1977] et permettent des raisonnements déductifs en retirant des domaines, les valeurs devenues incohérentes avec le modèle générique et les choix utilisateurs. Ces mécanismes de propagation permettent de construire la solution de manière

interactive. Comme pour les ontologies, les CSP séparent clairement la base de connaissance et son exploitation [Felfernig et al., 2014].

2.3 Concept d'héritage, de spécialisation et généralisation

[Taivalsaari 1996] définit le concept d'héritage comme le concept fondamental du paradigme de programmation orientée objets, permettant entre autres la réutilisabilité et l'adaptabilité des objets. L'héritage est une relation de généralisation / spécialisation que l'on retrouve dans les ontologies [Ohira, Hochin et Nomiya, 2011] : une classe-mère définie par ses caractéristiques, i.e. des attributs et des méthodes qui lui sont propres, transmet à ses classes-filles ses caractéristiques. Les classes-filles héritent donc des caractéristiques de leur classe-mère.

Le concept d'héritage possède les propriétés de transitivité, de non réflexivité, d'asymétrie et de non cycle. Il est possible de redéfinir les méthodes des classes-filles par le mécanisme de polymorphisme d'héritage afin de leur conférer un comportement spécifique. Dans ce cas, les méthodes gardent la même signature - elles portent sur la même liste d'attributs que leur classe-mère - mais produisent des résultats différents.

2.4 Communalité

De nombreuses définitions de la communalité de produits peuvent être trouvées dans la littérature, mais la définition proposée par [Ashayeri, J., Selen W., 2005] qui définit la communalité comme "le nombre de pièces/composants qui sont utilisés par plus d'un produit et qui est déterminé pour toutes les familles de produits", donne une compréhension simple du concept de communalité des produits.

L'indice de degré de commonalité - Commonality Index (DCI) a été introduit dans [Thevenot, H. J., Simpson, T. W., 2007]. Son calcul se base sur les informations contenues dans la nomenclature de la famille de produits. Le DCI évalue la communalité en se basant sur le rapport entre le nombre de composants communs dans une famille de produits et le nombre total de composants dans la famille.

2.5 Question de recherche

Au regard des différentes approches citées ci-dessus, la question de recherche suivante émerge :

> "Est-il possible de définir une ontologie des modèles génériques pour améliorer la génération, la mise à jour et la gestion de modèles génériques ?"

Nous posons, dans cet article, les éléments nécessaires à cela en utilisant conjointement les concepts d'ontologies, de problèmes de satisfaction de contraintes, d'héritage et de communalité.

3 Propositions

Dans cette section, les deux éléments utilisés pour réaliser l'ontologie de modèles génériques sont définis : les atomes génériques (GA) et les modèles génériques (GM). La spécialisation de ces deux éléments est elle aussi présentée.

3.1 Atomes générique ou GA

3.1.1 Définition des GA

Un atome générique ou GA est un élément de connaissance qui est cohérent et qui n'est pas, *a priori*, décomposable. Les GA correspondent donc aux familles de composants ou de modules, éléments feuille des nomenclatures. Par exemple, une roue de vélo, nommée *GA-Wheel*, est un composant nécessaire à la nomenclature d'un vélo.

Un GA est décrit au moyen d'attributs dont les valeurs possibles sont définies par leurs domaines de validité. Les attributs peuvent être symboliques (matériau de la roue par exemple) ou numériques (diamètre de la roue en pouces) et leurs domaines peuvent être discrets (*GA-Wheel.matériau = {carbone, acier, plastique}*) ou continus (*GA-Wheel.diamètre = {[12, 31]}*). Il existe des attributs spécifiques correspondant à des indicateurs clés de performance (KPI) tels que le coût, le poids, la performance, pour évaluer chaque GA. Certaines relations entre attributs permettent ou interdisent certaines combinaisons de valeurs et définissent ainsi les solutions possibles de la famille de GA. Par exemple, il n'est pas possible d'avoir des roues de diamètre inférieur à 16 pouces en carbone.

Nous proposons que dans un GA, la connaissance soit formalisée au moyen d'un CSP. Nous entendons ici par connaissances les relations autorisées entre les différentes valeurs d'attributs au sein d'un GA. Comme mentionné précédemment, un CSP est défini par un triplet {variables, domaines, contraintes} qui peut facilement être mis en correspondance avec les attributs, domaines et relations des GA. L'utilisation de CSP au sein d'un GA, nous permet d'utiliser des mécanismes de propagation [Mackworth, 1977], pour garantir la cohérence de ces connaissances en ne gardant que les valeurs autorisées dans ses domaines, tel qu'illustré en Fig.1. Nous utilisons les termes attributs, domaines et contraintes dans la suite du document.

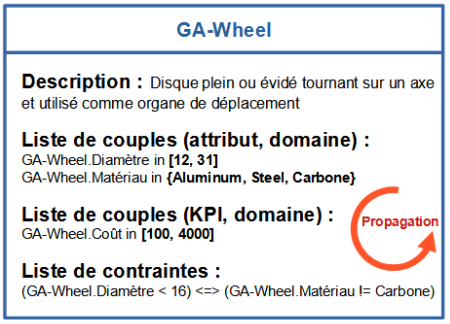

Fig. 1 : Atome de connaissance ou GA

3.1.2 Spécialisation des GA

Dans le modèle proposé, les GA peuvent être définis à différents niveaux de détail, en utilisant les concepts d'héritage et de spécialisation. Par exemple, une roue *GA-Wheel* pourrait-être spécialisée en roue de ville *GA-CityWheel*.

Un GA-Child peut être spécialisé à partir d'un GA-Parent plus général afin d'affiner ou de détailler ses connaissances. Dans ce cas, (1) le GA-Child hérite de toutes les caractéristiques du GA-Parent : attributs, domaines et contraintes. Ensuite (2), les domaines de validité du GA-Child peuvent être spécialisés (ou restreints) en :

- restreignant ses domaines hérités grâce à des modifications de contraintes héritées (ajout de combinaisons de valeurs interdites), tel que proposé par le polymorphisme d'héritage,
- définissant de nouvelles contraintes sur ses attributs. Dans le cas où le GA-Child n'a pas de contraintes spécifiques, ses domaines de validité sont égaux aux domaines de son GA-Parent.

Enfin (3), en plus du fait qu'un GA-Child hérite de son GA-Parent, des attributs, des domaines et des contraintes spécifiques qui sont dédiés à ce GA-Child et qui modélisent ses connaissances spécifiques peuvent être ajoutés. Par exemple, une roue de ville possède un certain nombre de catadioptres. Il faut ainsi ajouter un attribut numérique *GA-CityWheel.catadioptre ={[1, 6]}* au GA-CityWheel, tel qu'illustré en Fig. 2. Sa description est également complétée afin d'indiquer que l'utilisation est principalement adaptée au milieu urbain. Au sein d'un GA-Child, des connaissances héritées et des connaissances spécifiques peuvent ainsi être combinées grâce à des contraintes reliant les attributs hérités et les attributs spécifiques.

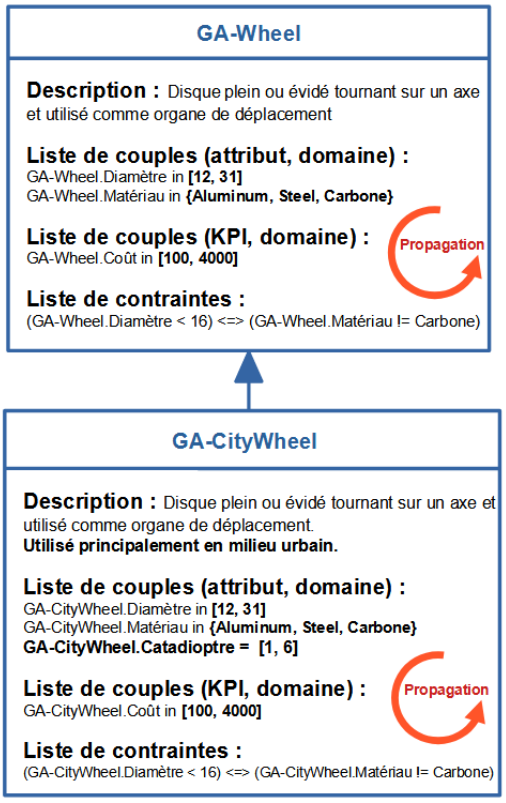

Fig. 2 : Spécialisation de connaissance

Le modèle proposé permet de définir une ontologie de GA avec différents niveaux de spécialisation. Un GA-Parent universel (*GA-Universal*) rassemble toutes les caractéristiques communes à l'ensemble des GA-Children, tels que les KPI par exemple. La modification des caractéristiques de tout GA-Parent (attributs, domaines, contraintes) est héritée par tous ses GA-Children et leurs descendants, grâce à la propriété de transitivité de l'héritage. Un exemple d'ontologie simple est présenté en Fig. 3. Nous retrouvons le *GA-Wheel* et le *GA-CityWheel* ainsi que le *GA-Rim* et le *GA-Tyre* qui sont tous des descendants du *GA-Universal*. L'indicateur Coût, présent dans le *GA-Universal*, est hérité par tous les GA descendants. La propagation des contraintes présentes dans chaque CSP permet de garantir la cohérence de chaque GA dans l'ontologie et, par conséquent, sa validité.

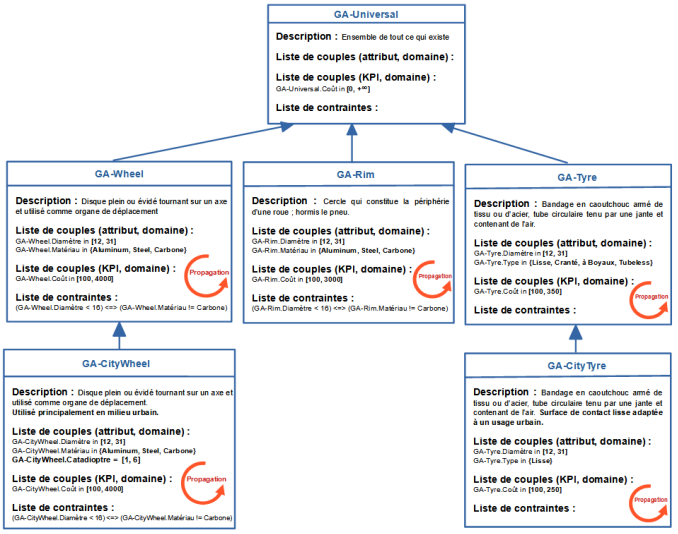

Fig. 3 : Ontologie de GA

3.2 Modèle générique ou GM

3.2.1 Définition des GM

Les Modèles Génériques (GM) correspondent à l'architecture des connaissances formalisant une famille d'artefacts. Ils correspondent donc aux familles de systèmes et/ou sous-systèmes et aux familles de services et/ou sous-services [Guillon et al., 2021]. Les GM sont hiérarchiques et basés sur l'ontologie de GA prédéfinie et celles des GM existants. Un GM intègre à la fois des GA et des GM pour créer la structure de la famille d'artefacts sur plusieurs niveaux de décomposition. Par exemple, un sous-ensemble roue peut-être décomposé en une jante et un pneu si les deux GA *GA-Rim et GA-Tyre* existent dans l'ontologie. Dans ce cas, *le GA-Wheel* précédent ne pouvant être décomposé, il faut créer un *GM-Wheel,* dans l'ontologie des GM, composé d'un *GA-Rim* et d'un *GA-Tyre,* comme illustré en Fig. 4. Cela permet ainsi de compléter l'ontologie en ajoutant d'autres types de relations entre les différents GA (ici, des relations de composition).

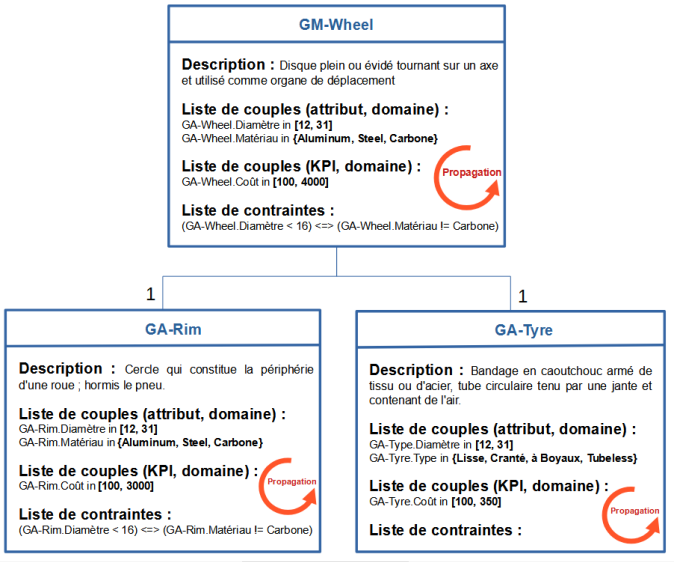

Fig. 4 : Construction *Bottom-Up* d'un modèle Générique GM

La construction des GM est donc *bottom-up* (des feuilles vers la racine de la nomenclature), récursive (incluant des GM) et structurelle (relation de composition). Un GM représente à la fois :

1. l'architecture générique, c'est-à-dire la nomenclature générique, de la famille d'artefacts. Une nomenclature est une

liste d'éléments (composants / modules, ou GA et sous-systèmes / sous-modules ou GM) avec leurs quantités nécessaires à la réalisation du système / service spécifique. Cette décomposition est réalisée au moyen du type d'association "est composé de".
2. les relations entre les différents GA et GM composant le GM, quel que soit leur niveau de décomposition. Ces relations peuvent être définies :

- directement entre les GA et GM, comme "obligatoire", "interdit", "exclu",
- ou entre les attributs des GA et GM, comme des combinaisons autorisées ou interdites de valeurs d'attributs.

Les relations entre les GA et GM permettent de formaliser tout l'espace des solutions de la famille d'artefacts, c'est-à-dire tous les systèmes / services qui peuvent être fabriqués à partir des connaissances du GM.
3. la méthode d'évaluation des KPI de chaque GM. Comme chaque GA et chaque GM sont évalués sur des indicateurs communs, il faut définir la manière de les agréger au niveau du GM. Il peut s'agir d'une simple somme ou d'une formule plus complexe (MAX, MIN, MOYENNE, etc.).

Au sein d'un GM et comme pour les GA, la connaissance est formalisée sous la forme d'un CSP pour modéliser i) les contraintes entre GA et GM ; ii) les contraintes entre les valeurs des attributs des différents GA et GM composants un GM ; iii) les contraintes liant les KPI. L'utilisation de CSP au sein d'un GM permet une propagation à la fois sur les GA et sur les GM, via les différentes contraintes exprimées. Le CSP garantit ainsi la cohérence du GM complet par rapport aux différentes connaissances de ses GA et GM.

Nous reprenons l'exemple de la figure 4, en ajoutant des contraintes entre le diamètre de la roue, celui de la jante et celui du pneu (Fig. 5). Pour qu'une composition de roue (i.e. un assemblage) soit cohérente, ces trois diamètres doivent être égaux.

Évaluation des KPI : Nous avons vu que les GM et GA sont évalués sur des KPI (par exemple, le poids, les performances, le prix, le coût, etc.) La manière de les agréger d'un niveau *n* au niveau supérieur direct *n+1* doit être définie au niveau du GM *n+1*, quel que soit le nombre de GA et GM composants le GM *n+1*. Une telle relation est donc formalisée sous d'une contrainte globale [Rossi, F., van Beek, P., et Walsh, T., 2006] pour éviter un inventaire exhaustif de tous les GA et GM impliqués. Cette contrainte globale est appliquée à chaque niveau de décomposition pour chaque GM. Dans l'exemple de la figure 5, nous ajoutons une contrainte globale permettant de calculer le coût de la roue en faisant la somme des coûts des composants (nommés "items" dans la contrainte). La propagation des contraintes des différents CSP entraîne la restriction du domaine du coût de la roue (qui passe de [100, 4000] à [200, 3350]).

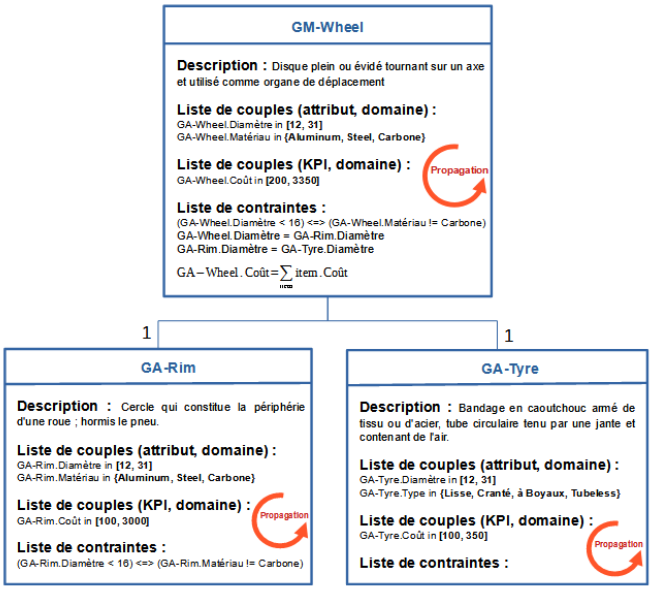

Fig. 5 : Exemple de relations entre différents GA dans un GM

3.2.2 *Spécialisation des modèles génériques*

Afin de modéliser efficacement la connaissance des modèles génériques, et de manière identique aux GA, les GM peuvent être définis à différents niveaux d'abstraction.

Un GM ou GM-Parent peut être spécialisé en un ou plusieurs GMs ou GM-Child spécifiques afin de raffiner ou de détailler la connaissance du GM-Parent. Cette spécialisation permet de définir des modèles plus spécifiques correspondant à des systèmes ou services plus détaillés. Les avantages de la spécialisation des GM, suivant le concept d'héritage, sont d'abord la propagation automatique des connaissances d'un GM-Parent suite à sa mise à jour vers tous ses GM-Child, grâce à la propriété de transitivité. Ensuite, la spécialisation évite aux experts en connaissances de partir d'une page blanche pour formaliser les connaissances d'une famille d'artefacts ayant une commonalité proche.

Plusieurs niveaux de connaissances sur une même famille d'artefacts peuvent ainsi être formulés et réutilisés. Premièrement, lors de la spécialisation, un GM-Child hérite de toutes les caractéristiques de son GM-Parent, c'est-à-dire de l'architecture du GM-Parent, de tous les GA et GM qui le composent avec leur CSP embarqué, de toutes les contraintes liant les GA et les GM, et de toutes les méthodes d'agrégation des KPI. Les contraintes héritées peuvent être redéfinies grâce au polymorphisme d'héritage et de nouvelles contraintes peuvent aussi être exprimées.

Deuxièmement, un GA ou GM hérité peut-être spécialisé dans le GM-Child par rapport à son GA-Parent ou GM-Parent. Afin de détailler les connaissances, il est parfois nécessaire de spécialiser un GA ou un GM en utilisant l'un de ses enfants GA-Child ou GM-Child (s'ils existent dans les ontologies de GA et de GM). Par exemple, dans le cas d'une roue de ville *GM-CityWheel*, il peut-être intéressant de remplacer le pneu *GA-Tyre* par un pneu plus spécialisé de type *GA-CityTyre*, s'il existe. La connaissance du GM-Child sera plus précise et exacte que celle de son GM-Parent. Ce mécanisme de spécialisation n'est autorisé que pour les GA appartenant à la même lignée familiale : le GA-Child plus spécialisé doit être un descendant du GA-Parent hérité. En effet, l'ensemble des attributs et contraintes hérités doit rester le même, entre le

GA-Child et le GA-Parent pour la cohérence globale du GM-Child. Dans l'exemple de la figure 6, un modèle générique de roue *GM-Wheel* composé d'un *GA-Rim* et d'un *GA-Tyre* peut être spécialisé en roue de ville *GM-CityWheel* composé d'un *GA-Rim* et d'un *GA-CityTyre*. Dans ce cas, nous voyons que le modèle générique *GM-CityWheel* est plus spécialisé que le *GM-Wheel*. La connaissance exprime clairement le fait qu'une roue de ville *GM-CityWheel* doit être composée d'un pneu de ville *GA-CityTyre* et une jante *GA-Rim*. La modification du modèle de plus haut niveau *GM-Wheel* (suite à une mise à jour par exemple) sera propagée sur tous les GM descendants (*GM-CityWheel* dans cet exemples). Nous voyons également que la propagation des contraintes entraîne une restriction du domaine de l'attribut coût dans le *GM-CityWheel* qui passe de [200, 3350] à [200, 3250].

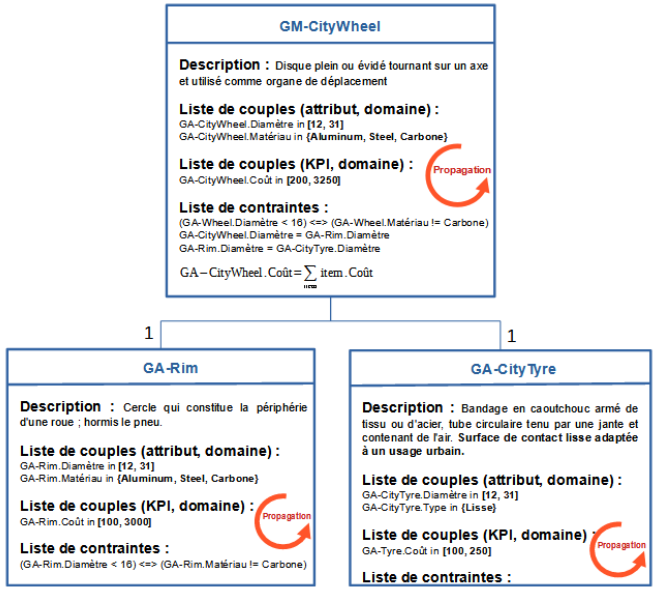

Fig. 6 : Exemple de spécialisation d'un GM

Troisièmement, un ou plusieurs GA ou GM spécifiques qui sont uniquement dédiés à un GM-Child et qui modélisent ses connaissances spécifiques, peuvent être inclus dans l'architecture du GM-Child, à n'importe quel niveau de décomposition. Dans l'exemple de la roue de ville *GM-CityWheel*, une dynamo (i.e. un *GA-Dynamo*) peut-être ajoutée si celle-ci existe dans l'ontologie des GA. Les GA et GM hérités ainsi que les GA et GM spécifiques peuvent être combinés grâce à de nouvelles contraintes reliant leurs attributs. Les KPIs du niveau supérieur direct prendront en compte ces nouveaux GA et GM (de par leur formalisation en contraintes globales) dans leurs évaluations et permettront l'évaluation de l'ensemble du GM-Child. Afin d'illustrer cette possibilité, nous créons un nouveau GA correspondant à une dynamo de moyeu (*GA-HubDynamo*) que nous ajoutons à l'ontologie de la figure 3. Le *GA-HubDynamo* est une spécialisation du *GA-Universal* (Fig. 7). Il possède deux attributs correspondants à la tension (comprise entre 6 et 12 volts) et la puissance (comprise entre 3 et 6 watts) de la dynamo. Son coût est compris entre 20 et 50.

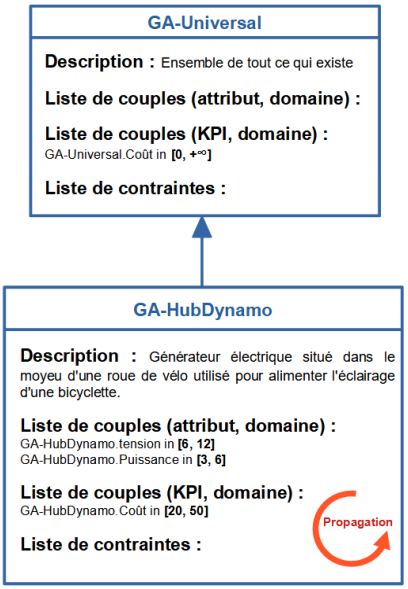

Fig. 7 : GA d'une dynamo de moyeu : *GA-HubDynamo*

Nous formalisons enfin un nouveau GM (*GM-CityWheelDynamo*) en spécialisant le *GM-CityWheel*. Le diamètre de ce type de roue doit être supérieur à 16 pouces afin de pouvoir accueillir le moyeu. La propagation des contraintes donne un coût de [220, 3300] et un diamètre supérieur à 16 pouces (pour la roue, la jante et le pneu). Il s'agit donc d'une connaissance sur une roue de ville spéciale avec un moyeu servant de générateur d'électricité (Fig. 8). La seule différence par rapport à une roue de ville étant le moyeu générateur, la commonalité est importante entre ces deux modèles génériques et notre approche de formalisation d'une nouvelle connaissance par spécialisation d'une connaissance existante prend tout son sens.

La spécialisation de GM permet de compléter l'ontologie initiale. En effet, chaque GM pouvant être spécialisé et chaque GM étant associé à son GA (cf. l'ontologie initiale Fig. 3), nous obtenons une ontologie de GM (exemple Fig. 9). La connaissance en configuration de biens et de services ainsi formalisée sera aisément maintenable et réutilisable. En effet, le modèle proposé, combiné aux mécanismes d'héritage, nous permet, de manière très simple mais efficace, de définir des GM avec plusieurs niveaux de spécialisation. Les GM sont basés sur les GA et GM existants, en les architecturant et en les reliant pour former un modèle générique cohérent d'une famille d'artefacts. Chaque GM peut être spécialisé (1) en ajoutant des attributs et/ou des contraintes, (2) en spécialisant certains GA ou GM (dans la même lignée familiale et les descendants uniquement), (3) en ajoutant de nouveaux GA ou GM à n'importe quel niveau de la nomenclature et à n'importe quel niveau d'abstraction, et (4) en ajoutant des contraintes entre les GA hérités et ajoutés. La modification des caractéristiques de tout GM-Parent (architecture, GA, contraintes, etc.) est ainsi héritée par tous ses GM-Child et leurs descendants. Notre proposition, couplant ontologie, CSP, héritage et commonalité, permet aux experts de concevoir et mettre à jour des GM d'une manière très simple et efficace. Le modèle proposé permet donc de définir une ontologie de GM avec différents niveaux de spécialisation, tel qu'illustré en Fig.9.

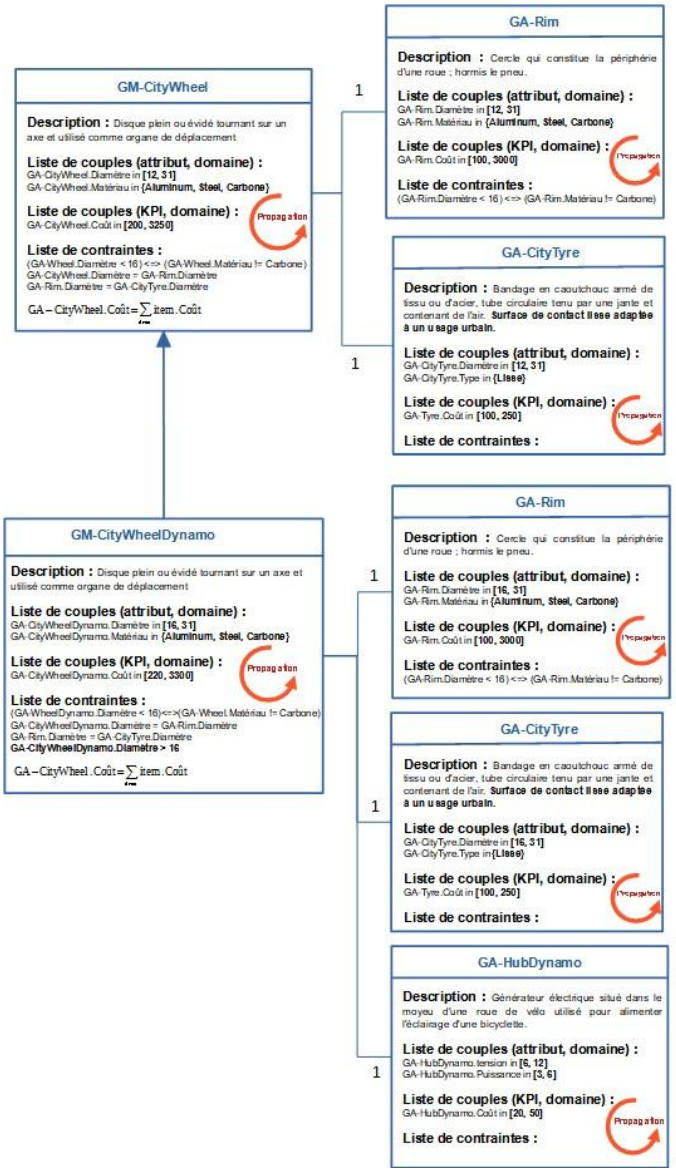

Fig. 8 : Spécialisation d'un GM par ajout d'un GA

4 Conclusion

Dans cet article, nous avons étudié la formalisation des connaissances en configuration qui a pour but de créer des modèles génériques. À notre connaissance, aucun travail scientifique n'a formalisé la spécialisation des modèles génériques et leur structuration sous forme d'ontologie.

L'objectif de cet article est donc de proposer les éléments nécessaires à la formalisation des connaissances et la définition des modèles génériques, et à leur spécialisation pour raffiner les connaissances et rendre les modèles de plus en plus spécifiques. Dans ce but, nous avons d'abord défini la notion d'*atome générique de connaissances* (AG), utile pour modéliser la connaissance générique des composants ou modules. Deuxièmement, nous avons défini la spécialisation des GA pour affiner ou détailler ses connaissances. Troisièmement, en utilisant les GA, nous avons défini la notion de *modèle générique* (GM) pour formaliser la connaissance d'une famille d'artefacts avec toutes ses options et variantes. Nous formalisons ces connaissances au sein des GA et des GM sous forme de CSP, et utilisons la propagation de contraintes pour garantir la cohérence des modèles.

Les contributions de cet article comprennent :

1. la formalisation de la connaissance en configuration en utilisant l'association de l'ontologie et de CSP, et en exploitant la notion de commonalité,
2. la définition des GM à différents niveaux de détails en utilisant les notions d'héritage et de spécialisation,
3. l'utilisation de la propagation des contraintes pour maintenir la cohérence des GA et GM.

Comme perspectives futures, nous avons l'intention de travailler d'abord sur (1) le développement de notre proposition pour prendre en compte la généralisation, (2) le codage de notre proposition dans une maquette réelle, (3) la prise en compte de la configuration Engineer-to-Order (ETO). La mise en œuvre de notre proposition sur des cas industriels réels est également dans notre esprit.

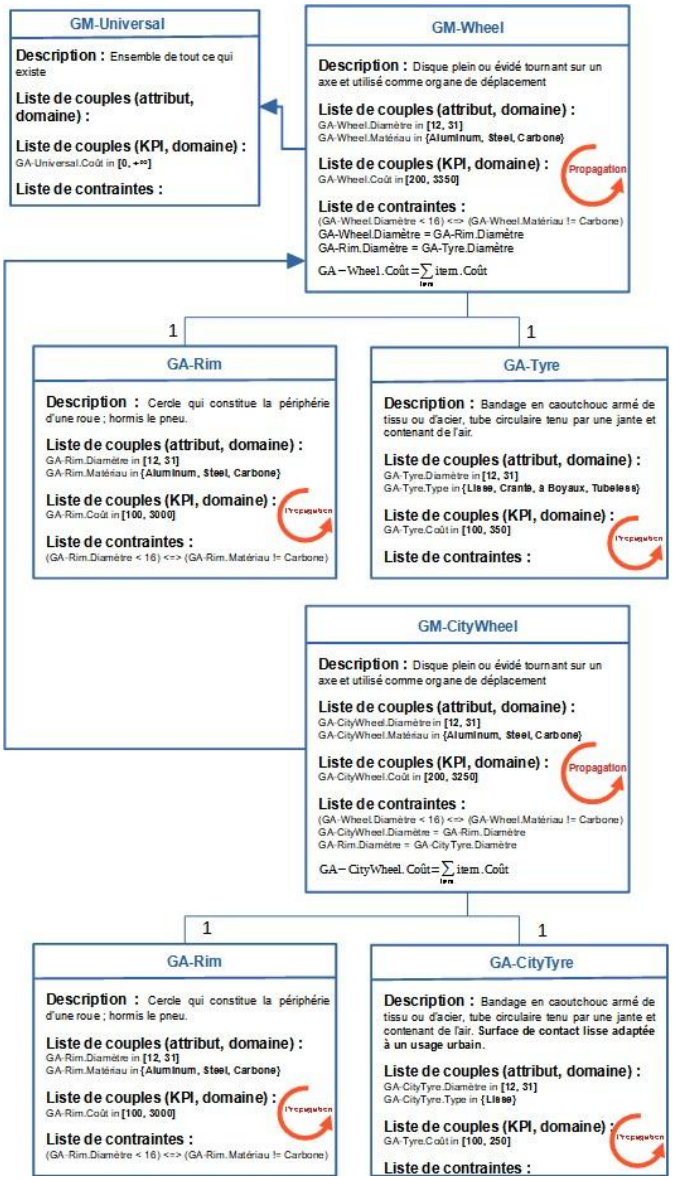

Fig. 9 : Ontologie de GM

5 Référence

Tan, K. C. (2001). A framework of supply chain management literature. *European Journal of Purchasing & Supply Management*, *7*(1), 39-48.

Terzi, S., & Cavalieri, S. (2004). Simulation in the supply chain context: a survey. *Computers in industry*, *53*(1), 3-16.

Chan, F. T., Chung, S. H., & Wadhwa, S. (2004). A heuristic methodology for order distribution in a demand driven collaborative supply chain. *International Journal of Production Research*, *42*(1), 1-19.

Guillon D., Ayachi R., Vareilles E., Aldanondo M., Villeneuve E., & Merlo C. (2021). Product˅service system configuration: a generic knowledge-based model for commercial offers. International Journal of Production Research,, 59(4), 1021-1040.

Soininen, T., Tiihonen, J., Männistö, T., & Sulonen, R. (1998). Towards a general ontology of configuration, Artificial Intelligence for Engineering Design, Analysis and Manufacturing: AIEDAM, 12(4), pp. 357–372.

Oddsson, G. & Ladeby, K. R. (2014). From a literature review of product configuration definitions to a reference framework, Artificial Intelligence for Engineering Design, Analysis and Manufacturing: AIEDAM, 28(4), pp. 413–428.

Felfernig A., Friedrich G., & Jannach D. (2001) Conceptual modeling for configuration of mass-customizable products Artificial Intelligence in Engineering, 15(2) 165-176.

Sabin, D. and Weigel, R. (1998). Product configuration frameworks, IEEE Intelligent Systems and their Applications, 13(4), 42-49.

Hotz, L., Felfernig, A., Stumptner, M., Ryabokon, A., Bagley, C., and Wolter, K. (2014). Configuration Knowledge Representation and Reasoning, Knowledge-Based Configuration. Elsevier, pp. 41–72.

Yang, D., Dong, M. and Miao, R. (2008). Development of a product configuration system with an ontology-based approach, CAD Computer Aided Design, 40(8), 863–878.

Studer, R., Benjamins, V. R. and Fensel, D. (1998). Knowledge Engineering: Principles and methods, Data and Knowledge Engineering, 25(1–2), 161–197.

Montanari U. (1974) Networks of constraints: Fundamental properties and applications to picture processing, Information Sciences - V 7, 95-132.

Mackworth A. K. (1977) Consistency in networks of relations, Artificial Intelligence, 8(1), 99-118.

Felfernig, A., Hotz, L., Bagley, C., and Tiihonen, J. (2014). Knowledge-Based Configuration From Research to Business Cases. Newnes.

Taivalsaari,A.(1996), On the Notion of Inheritance, ACM Computing Surveys , 28(3), 438-479.

Ohira, Y., Hochin, T. and Nomiya, H. (2011). Introducing specialization and generalization to a graph-based data model, Lecture Notes in Computer Science, 6884(4)1–13.

Ashayeri, J., Selen, W. (2005) An application of a unified capacity planning system. International Journal of Operations & Production Management, 25, 917-937.

Thevenot, H. J., Simpson, T. W. (2007). A comprehensive metric for evaluating component commonality in a product family. Journal of Engineering Design, 18(6), 577-598.

Rossi, F., van Beek, P., and Walsh, T. (2006). Handbook of Constraint Programming. Elsevier Science Inc., USA.