Une méthode, un langage et un environnement pour la modélisation et la résolution de problème de conception de système

Pierre-Alain Yvars, Laurent Zimmer

S.mart
Systems.Manufacturing.Academics.Resources.Technologies
Générations S.mart
Collectif-Partage-Diversité
18ème colloque national
4-6 avril 2023, Carry le Rouet

Une méthode, un langage et un environnement pour la modélisation et la résolution de problème de conception de système

Pierre-Alain. Yvars [a,*], Laurent. Zimmer [a]

[a] ISAE-Supméca, QUARTZ, 3 rue Fernand Hainaut, 93407 Saint Ouen Cedex, France
[b] Dassault Aviation, Direction de la Prospective, 78 quai Marcel Dassault, 92552 Saint Cloud, France

* e-mail : pierre-alain.yvars@isae-supmeca.fr

1. INTRODUCTION

Le principe général usuel en matière de conception de systèmes consiste à proposer un système candidat, puis à évaluer ses performances et à les comparer aux exigences attendues. Si les exigences sont vérifiées alors le candidat est une solution sinon un nouveau candidat doit être proposé et le processus répété (cf. Figure 1). La méthode est itérative, dite de de conception point à point, et l'on passe d'un candidat à l'autre jusqu'à trouver une solution valide, c'est-à-dire une solution qui vérifie toutes les exigences du cahier des charges. Les avantages et limites de cette vision de la conception ont été examinés dans [1, 2].

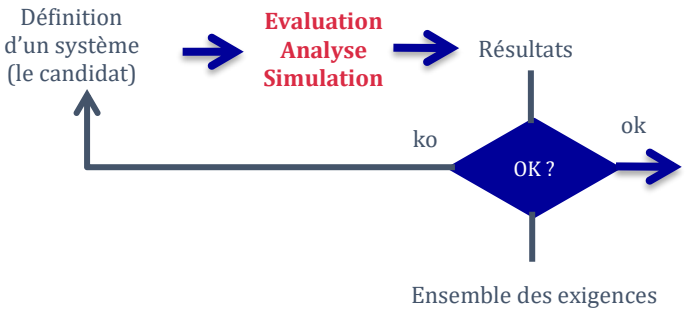

Figure 1 – Démarche usuelle de conception de système

A contrario, et face à un état de l'art dont les limites ont été détaillées dans [3], nous avons démarré en 2013 un travail de recherche visant à proposer une vision de la conception de système sous l'angle de la synthèse et basée sur des modèles. Les problèmes adressés relèvent du dimensionnement, de la configuration, de l'allocation/déploiement et de la génération d'architecture. Les types de systèmes considérés peuvent être à dominante technologique (mécanique, électronique, génie électrique…), à dominante logiciel (système embarqué) ou bien encore des systèmes cyber physiques (CPS).

Nous présentons ici les résultats de notre démarche selon trois aspects : une vision méthodologique de la synthèse de système appelée MBSS (Model Based System Synthesis), un langage de représentation du problème (DEPS) ainsi qu'un environnement de modélisation et de résolution (DEPS Studio). Enfin nous éliciterons un ensemble de cas d'étude traités à l'aide du MBSS et destinés à alimenter et valider nos travaux. Ce papier est volontairement dédié à une vision synthétique des travaux réalisés. Pour plus de détails, le lecteur pourra se référer à la bibliographie.

2. LE MBSS

2.1. Principes

Plusieurs principes de base caractérisent l'approche MBSS [2]:
- une approche "set based design" dans laquelle l'espace de conception est réduit au fur et à mesure que les exigences sont placées dans l'espace de conception
- la modélisation du problème de conception plutôt que la description d'un système candidat
- la prise en compte simultanée de plusieurs exigences hétérogènes
- l'utilisation des exigences pour réduire l'espace des solutions basées sur les exigences.

L'approche pour résoudre le problème est dite "par satisfaction des exigences" et non par "validation et vérification" puisqu'à tout moment nous essayons de

trouver des solutions dans les limites de l'espace de conception contraint par les exigences. Les solutions trouvées sont alors nécessairement correctes par construction (cf Figure 2). On tend alors vers un cycle en I de conception plutôt que vers un traditionnel cycle en V.

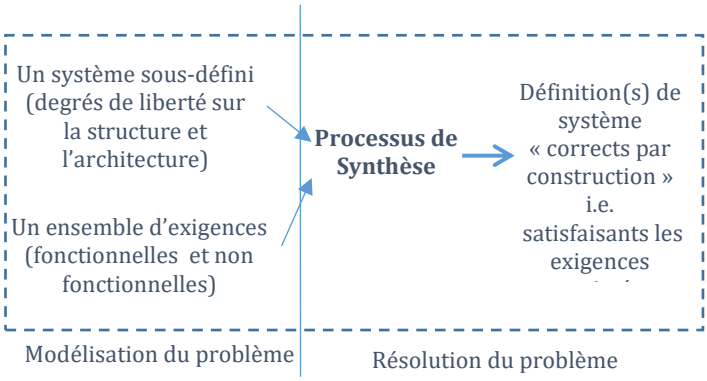

Figure 2 – Processus MBSS

2.1. Modéliser le problème au lieu de décrire le système

Un objet dont les valeurs des inconnues ne sont pas fixées a priori sera appelé un objet sous-défini.

La modélisation du problème nécessite un formalisme ayant les capacités suivantes de :
- modélisation d'une architecture sous-définie : les inconnues de l'architecture, leur domaine de valeurs possibles, les propriétés structurelles devant être satisfaites nécessairement par toute architecture de solution et la décomposition hiérarchique de l'architecture sous-définie
- modélisation d'un ensemble de comportements sous-définis : les inconnues comportementales, leur domaine de valeurs possibles, les propriétés comportementales et la décomposition hiérarchique des comportements
- partage de l'architecture sous-définie entre plusieurs comportements. En effet, une solution du problème sera une instanciation complète du modèle d'architecture satisfaisant l'union de tous les comportements exprimés
- définition des exigences : sous forme de propriétés structurelles et comportementales sur l'architecture sous-définie et les modèles de comportement.

De plus en raison de la diversité des problèmes de conception abordés :
- les inconnues doivent pouvoir prendre leurs valeurs dans des domaines mixtes : intervalles de nombres réels, intervalles d'entiers, énumérations de valeurs réelles, énumérations de valeurs entières
- les propriétés exprimées doivent pouvoir l'être sous forme d'équations et d'inégalités algébriques linéaires et non linéaires mais aussi sous forme de propriétés spécialisées telles que les valeurs à prendre dans les catalogues.

Les concepts énoncés ci-dessus apparaissent dans le méta-modèle MBSS de la figure 3.

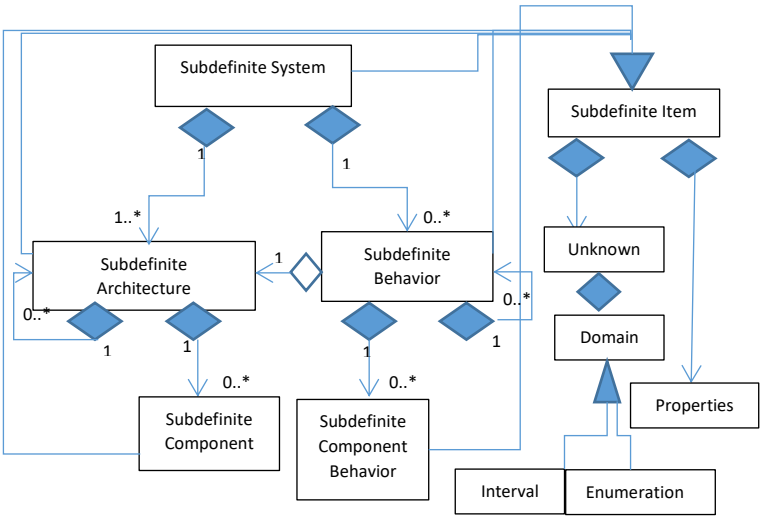

Figure 3 – Extrait du méta modèle MBSS [2]

3. LE LANGAGE DEPS

DEPS est un langage de modélisation textuel, déclaratif, structuré à base de modèle et de propriétés, offrant la possibilité de décrire une ontologie des quantités physiques et permettant la représentation des variables et des propriétés dans des domaines discrets et continus. En tant que langage déclaratif, DEPS offre à l'ingénieur la possibilité de décrire son problème de conception sans décrire la manière de le résoudre. A charge ensuite à un outillage dédié équipé d'un solveur générique de résoudre le problème [3-5].

3.1. Ontologie pour ingénieur

Les données manipulées dans DEPS peuvent être :
- des valeurs entières ou réelle
- des intervalles entiers, des intervalles réels, des domaines énumérés entiers ou des domaines énumérés réels.

Si l'on s'en tient à ces types simples, on ne peut pas représenter les quantités manipulées par les concepteurs de systèmes techniques. C'est pourquoi nous avons proposé deux concepts pour représenter ces quantités : *QuantitKind* et *Quantity.*

QuantityKind Volume **Type** : real ; **Min** : 0 ; **Max** : +maxreal ; **Dim** : L3 ; **End**	**Quantity** Volume **Kind** : Voltage ; **Min** : 0; **Max** : 1000; **Unit** : m3 ; **End**

Figure 4 - Exemple de Quantity et QuantityKind

Un *QuantityKind* porte un type de base (entier ou réel), une limite min, une limite max ainsi que la dimension au sens de l'analyse dimensionnelle de la quantité. Les valeurs infinies négatives et positives pour les bornes

du domaine sont notées -maxreal et +maxreal pour les réels et -maxint, +maxint pour les entiers. Une quantité est définie à partir d'un *QuantityKind*. La *QuantityKind* porte la dimension, la *Quantity* porte l'unité (cf Figure 4).

3.2. Représentation structurée via des modèles

De manière générale, un problème de conception est représenté par un ensemble de modèles DEPS, de sorte que la caractéristique fondamentale du langage DEPS est le "modèle" (*Model*). Tout modèle encapsule dans l'ordre : un ensemble d'arguments, un ensemble de constantes (*Constants*), un ensemble de variables (*Variables*), un ensemble d'éléments (*Elements*) et un ensemble de propriétés (*Properties*). Les arguments peuvent être des constantes ou des identifiants d'éléments. Les éléments sont des instances d'autres modèles.

DEPS offre des capacités de structuration des modèles et permet ainsi de capturer la structure sous-définie d'un problème et d'une classe de système. Il est possible d'exprimer des relations d'héritage, de composition et d'agrégation entre modèles et instances de modèles. Tout modèle possède une signature ce qui permet d'exprimer plusieurs modèles ayant le même nom avec des arguments et des contenus différents (cf Figure 5).

```
Model GasModel (MM)
Constants
 MM : MolarMass ;
Variables
 M : mass ;
Elements
Properties
End
```

```
Model Tank(p, t, Gas)
Constants
 R : Real = 8.314 ;
 p : Pressure ;
 t : Temperature;
Variables
 V : Volume ;
Elements
 Gas : GasModel[MolarMass] ;
Properties
 p*V= (Gas.M/Gas.MM)*R*t;
End
```

Figure 5 - Exemple de modèle de Gaz et de réservoir en DEPS

3.3. Encapsulation des propriétés

Plusieurs types de propriétés peuvent être encapsulées dans un modèle DEPS :

- des équations ou inéquations algébriques linéaires ou non linéaires
- des propriétés en extension, par morceaux
- des méta propriétés : contraintes conditionnelles.

```
Problem Problem()
 Constants
 Variables
 Elements
  O2:GasModel(0.032);
  H2:GasModel(0.002);
  T1:Tank(2.56e+7,300,O2);
  T2:Tank(7.00e+7,300,H2);
 Properties
  O2.Mass = 10;
  T2.V = 500;
End
```

Figure 6 - Exemple de problème en DEPS

Finalement, les instances de modèles doivent satisfaire les propriétés exprimées dans le modèle ou héritées. Il est également possible de définir un critère à optimiser. Une fois les modèles créés, il est nécessaire d'exprimer le problème (*Problem*) particulier à résoudre (cf Figure 6).

4. L'ENVIRONNEMENT DE MODELISATION ET DE RESOLUTION DEPS STUDIO

L'environnement intégré de modélisation et de résolution DEPS Studio [5-7] associé au langage DEPS comprend un éditeur de modèle, un gestionnaire de projet, un compilateur et un solveur. L'expérience montre que la spécification d'un problème de conception de système n'est jamais correcte du premier coup et que de nombreuses erreurs de modélisation ne sont détectables que par le calcul. Nous avons donc décidé de développer et d'intégrer notre propre solveur dans l'environnement de développement afin que la recherche de solutions contribue efficacement au processus de modélisation du problème. Il s'agit d'une approche de développement rapide de modèles (analogue à une approche RAD) qui, contrairement à une approche de transformation de modèles, réduit le temps d'exécution de la boucle de développement de modèles. Elle permet également de remonter les erreurs jusqu'au bon niveau d'abstraction.

La résolution d'un problème de conception nécessite la capacité de prendre en compte :

- les problèmes sous contraintes
- les équations et inégalités algébriques non linéaires sur des domaines mixtes
- d'autres types de relations telles que les tables de valeurs.

Pour ce faire, nous avons développé un solveur mixte, à base de contraintes, dédié au calcul sur des modèles DEPS structurés.

Les méthodes de calcul que nous utilisons sont issues des techniques de programmation par contraintes. La structure des modèles DEPS est préservée tout au long de la chaîne de compilation jusqu'aux modèles de calcul.

Le solveur implémente une méthode de propagation HC4 révisée sur les équations et les inégalités. Initialement prévue pour les domaines continus, nous avons étendu la méthode aux quatre types de domaines : les intervalles réels ouverts, les intervalles entiers, les ensembles énumérés de valeurs flottantes et les ensembles énumérés de valeurs entières signées. Il traite naturellement les boucles algébriques.

Dans le cas d'un problème sur-contraint, un échec peut survenir lors de la première propagation ou après avoir exploré les parties restantes de l'arbre de recherche. L'échec est interprété comme la preuve qu'il n'existe pas de solution au problème et non comme un échec de l'algorithme de résolution. A contrario, l'algorithme de résolution garantit la correction des solutions générées.

5. CAS D'ETUDE ACADEMIQUES ET INDUSTRIELS

Notre approche a été validées sur les cas d'étude académiques et industriels suivants :

a) dimensionnement de robot sous exigences fonctionnelles [3]

b) dimensionnement de système de transmission par adhérence sous exigences fonctionnelles et environnementales [8]

c) configuration et positionnement de caméra embarquée sous exigence de coût et de fiabilité [2]

d) synthèse d'architecture avionique modulaire embarquée sous exigences de sûreté de fonctionnement, sécurité et capacité [9-10]

e) synthèse d'architecture de commande de système de gestion et distribution électrique embarquée avion sous exigences de sûreté de fonctionnement [11]

f) génération d'architecture de système de stockage d'énergie embarqué sous exigences fonctionnelles [12-15].

Le tableau 1 synthétise pour chaque cas d'étude :

- le type de problème adressé : dimensionnement (d), configuration (c), allocation/déploiement (a), génération d'architecture (ga)
- le type d'inconnue : Discret, continu, Mixte
- le type de système : mécanique (m), génie électrique (ge), système embarqué (se), cyber-physique (cps)
- le type d'exigence : fonctionnelle (f), sûreté de fonctionnement (sur), sécurité(sec), environnemental (env), coût (co), fiabilité (fia).

Tableau 1 – Caractéristiques des cas d'étude

Cas d'étude	Type de problème	Calcul	Système	Exigence
a	d	continu	m	f
b	d	mixte	m	f, env,co
c	d,c	mixte	cps	f, co, fia
d	d, a, ga	discret	se	f, sec, sur
e	d, a, ga	discret	cps	f, sur
f	d, ga	mixte	ge	f

6. CONCLUSION

Dans cet article, nous avons présenté une première version de travaux visant à concevoir des systèmes techniques par le biais de la synthèse. La démarche que nous proposons est particulièrement indiquée en conception préliminaire et en intégration de systèmes. Elle est composée de trois volets : La méthode MBSS, manière de voir la synthèse de systèmes, le langage DEPS pour la représentation des problèmes de conception et l'environnement intégrer de modélisation et de résolution DEPS Studio. La démarche a été validée sur plusieurs cas d'étude académiques et industriels.

Dans sa version actuelle, DEPS permet d'exprimer naturellement des problèmes de dimensionnement. Il dispose également de fonctionnalités permettant la représentation de certains problèmes de configuration, d'allocation et d'architecture. Nous travaillons au sein de l'association DEPS Link [16] à enrichir le langage DEPS afin qu'il puisse exprimer d'autres types de problèmes de conception. Nous nous concentrons sur la construction de collections d'éléments et l'expression de propriétés sur celles-ci. Nous travaillons également sur l'implémentation de contraintes spécialisées. L'objectif à long terme est d'adresser des problèmes complexes de synthèse d'architecture combinant de nombreuses exigences fonctionnelles (performance...) et non fonctionnelles (sûreté, sécurité...). Les résultats seront intégrés dans la prochaine version de DEPS Studio.

RÉFÉRENCES

[1] Yvars P.A., Zimmer L.. System Sizing with a Model-Based Approach: Application to the Optimization of a Power Transmission System, Mathematical Problems in Engineering, Vol 18, July 2018.

[2] Yvars P.A., Zimmer L.. Toward a correct by construction design of complex systems : The MBSS approach, Procedia CIRP, Vol 109, Elsevier, 2022.

[3] Yvars P.A., Zimmer L.. DEPS Un langage pour la spécification de problèmes de conception de Systèmes, proc of the 10th International Conference on Modeling, Optimization & SIMulation (MOSIM 2014), France.

[4] Yvars P.A., Zimmer L.. Integration of Constraint Programming and Model-Based Approach for System Synthesis, proc of the IEEE International Systems Conference, SYSCON 2021, Vancouver, Canada.

[5] Yvars P.A., Zimmer L.. Contraintes, objets et ontologies pour la conception de systèmes complexes, Conférence Nationale sur les Applications pratiques de l'Intelligence Artificielle, dans le cadre de la plateforme IA, PFIA, Bordeaux, 2021.

[6] Yvars P.A., Zimmer L.. DEPS Studio : Un environnement intégré de modélisation et de résolution de problèmes de conception de systèmes, 8ème Conférence en ingénierie du logiciel (CIEL 2019), Toulouse, 2019.

[7] Yvars P.A., Zimmer L.. DEPS Studio : Un environnement intégré de modélisation et de résolution pour la synthèse de système, 20ème journées AFADL, Approche Formelle dans l'Assistance au Développement de Logiciel, Montpellier, 2021.

[8] Yvars P.A., Zimmer L.. A Model-based Synthesis approach to system design correct by construction under environmental impact requirements, Procedia CIRP, Vol 103, Elsevier, 2021.

[9] Zimmer L., Lafaye M., Yvars P.A.. Modélisation d'exigences pour la synthèse d'architecture avionique : Application à la sûreté de fonctionnement, 16eme journées AFADL, Approche Formelle dans l'Assistance au Développement de Logiciel, Montpellier, 2017.

[10] Zimmer L., Yvars P.A., Lafaye M.. Models of requirements for avionics architecture synthesis: safety, capacity and security, proc of the Complex System Design and Management conference – CSD&M 2020, December 2020, Paris, France.

[11] Zimmer L., Yvars P.A.. Synthesis of software architecture for the control of embedded electrical generation and distribution system for aircraft under safety constraints: The case of simple failures, proc of the 14th International Conference of Industrial Engineering, CIGI-QUALITA 2021, Grenoble, France.

[12] Diampovesa S., Hubert A., and Yvars P.A., "Modélisation d'un problème de conception en vue de réutilisabilité. Exemple d'une batterie Li-ion", Symposium de Génie Electrique (SGE 2020), 2020.

[13] Diampovesa S., Hubert A., Yvars P.A.. Designing Physical Systems through a Model-Based Synthesis Approach. Example of a Li-ion Battery for Electrical Vehicles, Computers In Industry, Vol 129, 2021.

[14] Diampovesa S, Yvars P.A., Hubert A.. Using Synthesis and Analysis for Design in Systems Engineering: an Integrated Approach, INSIGHT, Vol 24, Issue 4, Wiley, 2021.

[15] A. Hubert A., Forgez C., Yvars P.A.. Designing the architecture of electrochemical energy storage systems. A model-based system synthesis approach, Journal of Energy Storage, Vol 54, 2022

[16] site de l'association depslink. https://www.depslink.com