Méthodologie et outils pour le dimensionnement

Eric Atienza

INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE

N° attribué par la bibliothèque
/__/__/__/__/__/__/__/__/__/__/

THESE

pour obtenir le grade de

DOCTEUR DE L'INPG

Spécialité : « Génie électrique »

Préparée au sein du **Laboratoire d'Electrotechnique de Grenoble**

dans le cadre de l'Ecole Doctorale
« Electronique, Electrotechnique, Automatique, Télécommunication, Signal »

présentée et soutenue publiquement par

Eric ATIENZA

Ingénieur ENSIEG

Le 4 juillet 2003

Méthodologie et outils pour le dimensionnement

Directeur de thèse : Jean BIGEON

Monsieur	Jean-Claude SABONNADIERE	Président
Messieurs	Marcel JUFER	Rapporteur
	Claude MARCHAND	Rapporteur
	Jean BIGEON	Directeur de thèse
	Robert PERRET	Examinateur
	Vincent MAZAURIC	Examinateur

INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE

N° attribué par la bibliothèque
/__/__/__/__/__/__/__/__/__/__/

THESE

pour obtenir le grade de

DOCTEUR DE L'INPG

Spécialité : « Génie électrique »

Préparée au sein du **Laboratoire d'Electrotechnique de Grenoble**

dans le cadre de l'Ecole Doctorale
« Electronique, Electrotechnique, Automatique, Télécommunication, Signal »

présentée et soutenue publiquement par

Eric ATIENZA

Ingénieur ENSIEG

Le 4 juillet 2003

Méthodologie et outils pour le dimensionnement

Directeur de thèse : Jean BIGEON

Monsieur	Jean-Claude SABONNADIERE	Président
Messieurs	Marcel JUFER	Rapporteur
	Claude MARCHAND	Rapporteur
	Jean BIGEON	Directeur de thèse
	Robert PERRET	Examinateur
	Vincent MAZAURIC	Examinateur

à Caroline&Ninon…

REMERCIEMENTS

« Avoir beaucoup d'amis, c'est n'avoir pas d'amis. »
Aristote

Je tiens à remercier Jean-Claude Sabonnadière, en sa qualité de président du Jury, en sa qualité de directeur du LEG quand j'ai intégré cette précieuse institution, en sa qualité de président de GRAIN. Je le remercie également pour la chaleur de ses encouragements lors de mes soutenances (pour le diplôme d'ingénieur déjà …)

Je tiens à remercier Marcel Jufer, qui n'est pas qu'un livre qui fait référence, mais aussi quelqu'un qui m'a appris qu'il n'y avait pas d'accent suisse, et qu'en Suisse les aiguilles tournaient toutes dans le même sens !

Je remercie chaleureusement Robert Perret qui en comprenant mon travail, m'a montré que l'électronique de puissance permettait de tout comprendre.

Je remercie Claude Marchand d'avoir accepté d'être dans mon jury, et d'avoir fait l'effort de compréhension de ma prose, souvent pas très claire. Je le remercie également pour l'extrême minutie avec laquelle il a lu et annoté mon rapport, j'en suis flatté, merci.

Je remercie Vincent Mazauric de la passion avec laquelle il défend l'enthalpie et la relaxation. Et moi aussi je préfère que l'enthalpie soit libre. Je le remercie également car sa présence en tant qu'examinateur, et en tant que membre de Schneider Electric me conforte dans l'intérêt qu'une telle société porte à mes travaux.

Je tiens à remercier Jean Bigeon. Je rends d'abord hommage à ses qualités humaines. J'ai choisi mon sujet de thèse pour travailler avec lui, car j'ai senti une capacité de fonceur et de catalyseur d'idées. Mais j'ai découvert bien plus que cela. J'ai découvert quelqu'un d'intelligent et généreux. Des qualités qui ne lui sont pas assez reconnues à mon avis.

Je le remercie pour l'opportunité qu'il m'a offerte de m'exprimer dans son équipe de recherche, de l'esprit de liberté qui y a régné grâce à lui, et des valeurs qu'il y a diffusé.

Je tiens à remercier Alain Bolopion. Comme il fuit les honneurs comme la peste, je respecterai cette pudeur en n'étalant pas trop la liste de ses qualités.

Je le remercie de l'aide précieuse qu'il m'a apportée lors de la rédaction du manuel, mais aussi lors de la préparation de l'exposé.

C'est une personne exceptionnelle qui est bien trop souvent sous estimée. J'ai vu un jour une chose incroyable : un exposé d'Alain sur JNI ! C'était grandiose ! Si vous avez l'occasion, essayez donc d'assister à un de ses exposés sur un sujet qu'il maîtrise. On en sort plus intelligent.

Je tiens à remercier Jean-Michel Guichon à plus d'un titre. D'abord pour son travail, puisqu'il a activement participé au développement de ma plateforme, en essayant d'y faire atterrir son logiciel précolombien. Mais surtout pour tous les cafés que je lui ai offerts et qui ne m'ont pas coûté un rond !

Je tiens à remercier Armando Fonseca de s'être déplacé de Paris pour ma soutenance, déjà. Je ne compte pas non plus les cafés, les cités d'or…

Je remercie ensemble Jean-Michel et Armando, car ils m'ont appris le sens de mots comme ténacité, courage, intelligence … Et ce n'est pas facile !

Je tiens à remercier Christophe Lechevalier qui m'a initié aux ficelles du LEG. C'est assez compliqué, et même ainsi guidé on peut commettre des erreurs (aider des gens malgré eux, considérer certaines choses comme acquises …).

Je remercie Néo, P'tit Ben, Big ben, DEDEEEE, Nuts, Bibi, Pichu, Disciplus Simplex, laule, nioniotte : Cherchez l'intrus. On se marre bien aussi des fois souvent quand même en thèse.

Normalement cette thèse signe la fin définitive de mes études. J'en profite donc pour remercier trois personnes dans tout mon cursus (26 années, une centaine de profs …). Ces trois personnes m'ont un jour rattrapé dans mes errances, ou bien sont parvenues à me canaliser.

- Merci à Madame Montand, instit de c.p. sans qui j'aurais redoublé ma maternelle, qui a su empêcher que mon trop plein d'énergie serve à taper les copains.
- Merci à Monsieur Douillet, prof de Math-Sup, qui m'a permis de comprendre que l'on n'est jamais sûr d'avoir été assez intelligent.
- Merci à Jean Bigeon, mon directeur de thèse, qui le premier a réussi à focaliser presque toute mon énergie sur mon travail.
- Merci également à MM Dufait, Potier, Rossignol, Piguet, Hatané tous aussi stimulants.

Je remercie la jeune équipe de Design Processing Technologies pour sa confiance, et sa motivation.

Enfin, mon dernier remerciement va directement à Caroline. Cette thèse est aussi la sienne. Pour faire une thèse il faut certes faire travailler sa matière grise, mais il faut aussi du cœur à l'ouvrage.

TABLE DES MATIERES

INTRODUCTION

René Taton [Taton-95] dans son histoire des sciences présente pour les mathématiques l'évolution des concepts, et un d'entre eux est particulièrement remarquable : les équations. Même s'il semble évident à tous qu'un jour les équations n'ont pas existé, il est difficile de comprendre comment nos ancêtres faisaient pour s'en passer.

Dans « la science antique et médiévale », René Taton [Taton-95-1] cite l'exemple d'un problème mathématique et décrit la façon dont un scribe égyptien le résolvait :

« Une quantité dont la quatrième partie lorsqu'on la lui ajoute devient quinze. Quelle est cette quantité ? »

En bon scientifique contemporain, je pose x cette quantité et j'écris :

$x+x/4=15$

$x*(1+1/4)=15$

$x=15*(4/5)=12$

Mais comment procédait un scribe en Egypte ancienne ?

« Compte avec 4.
Calcule le quart, à savoir 1, Total 5.
Compte avec 5 pour trouver 15 : 3 (15/5).
Multiplie 3 par 4 : 12. C'est le résultat »

On voit bien que le raisonnement est peu construit, sachant qu'il part de 4 comme étant un nombre facilement divisible par 4. Pour R. Taton, il est peu probable que les Egyptiens aient disposé de la notion d'équation, même si le sujet est un peu controversé. On imagine bien la difficulté qu'ils pouvaient avoir pour résoudre des problèmes, même aussi simples.

Par ailleurs il faut ajouter que la méthode de calcul chez les Egyptiens est tout aussi étonnante. Ainsi pour multiplier 13 par 7 un scribe procédait comme suit :

* 1	7
2	14
* 4	28
* 8	56
	—
résultat 91	

Le scribe écrit dans la colonne de droite le multiplicateur 7, dans celle de gauche 1. Il double les nombres des deux colonnes en les mettant dans une nouvelle ligne, jusqu'à ce qu'il puisse en additionnant des nombres de la première colonne obtenir le multiplicande (les lignes sont marqués d'une étoile). Le résultat est alors la somme des nombres sélectionnés de la seconde colonne.

Ainsi, semble-t-il, a commencé le calcul. Après le calcul viennent naturellement les équations, qui sont une représentation à la fois répétable et généralisée d'un calcul.

On trouve chez les Babyloniens des énoncés de problèmes que nous résolvons aujourd'hui par des paires d'équations à deux inconnues, et même des équations du second

degré. Sous certaines réserves il semble que l'on puisse attribuer aux Babyloniens la maîtrise des équations. En effet ils classaient les problèmes en type et donnaient pour chaque type beaucoup d'exemples de résolution. Ceci permettait de répéter la procédure de calcul, et de résoudre un problème différent mais similaire.

La forme que prend cette équation, 3000 ans avant notre ère, est étourdissante. Les procédés qui conduisent à la solution sont implicitement écrits par l'accumulation d'exemples.

Lorsque l'on a suffisamment manipulé d'équations, on peut commencer à exprimer des règles qui ne dépendent pas des problèmes que l'on traite, des simplifications qui peuvent s'opérer dans tous les cas, et qui dépendent uniquement de la forme des équations.

On note à ce titre ce qui semble être les premiers opérateurs formels vers le IX siècle dans la civilisation arabe, avec le célèbre Al-Khwārizmï, qui définissait deux opérateurs formels d'équations :

« al-jabr » qui consiste à passer les termes qui sont à soustraire d'un membre de l'équation à l'autre. Ceci a d'ailleurs donné la mot « algèbre ».

« Al-muqābāla » qui consiste à réduire deux termes égaux appartenant aux deux membres.

Ce rudiment de calcul symbolique s'étend ensuite avec le calcul symbolique des dérivées, des intégrales, des résolutions d'équations …

En 5000 ans les notions aussi simples que celle d'une multiplication, ou bien celle d'une équation, ont certes évolué, mais surtout, il y a 5000 ans que des hommes écrivent des équations. Il y a 5000 ans que les hommes apprennent à faire confiance au calcul numérique et 12 siècles qu'ils apprennent à faire confiance au calcul symbolique. Ils apprennent à exploiter la puissance de ces calculs, à savoir jusqu'où on peut leur faire confiance. Il y a 1 siècle environ que l'on sait déléguer le calcul numérique à des machines, et quelques décennies que l'on sait déléguer certains calculs symboliques à ces mêmes machines.

L'équation est un des plus vieux acquis de la science humaine, et reste encore un domaine de recherche très actif. On écrivait des équations avant l'invention du papier, et on écrit encore des équations après l'avènement de l'ordinateur. Dans la recherche d'un formalisme qui sert de support à la connaissance scientifique et technique, il semble que celui des équations ait fait ses preuves.

Deux des autres grands formalismes de la connaissance sont celui de l'écriture et du langage. C'est pourquoi il faut rédiger des rapports de thèse. Celui-ci va traiter d'un outil et d'une méthodologie pour le dimensionnement technique. Le système qui lie science et technique, et qui lie également industrie et recherche, se construit itérativement et interactivement, chacun profitant des avancées de l'autre. Les services de conception des entreprises industrielles, « Recherche et Développement » ou « Division Scientifique et Technique », ont tous sensiblement la même vocation : la conception de leur produit. Cette conception a beaucoup évolué dans le sens d'une automatisation et d'une informatisation, par le dessin assisté par ordinateur et le prototypage virtuel. Mon équipe de recherche a estimé que les prochaines évolutions se situaient au niveau de ce qui se passe avant même le premier prototype virtuel. De plus il nous a semblé logique que l'activité précédant ce prototype est le dimensionnement, c'est donc dans ce sens que nous avons œuvré.

Dans le premier chapitre nous voyons comment se situe le dimensionnement dans la conception, et quels sont les outils actuels qui aident au dimensionnement. Nous regardons aussi où en sont les méthodes de développement logiciel, afin de pouvoir implanter les méthodes que nous dégageons, soit dans un outil existant soit dans un nouvel outil.

Dans une première partie nous essayons de voir comment se situe le dimensionnement par rapport aux activités actuellement présentes en conception, c'est à dire le marketing, la production, le choix de structure et la validation. Nous essayons de donner une définition au dimensionnement ainsi qu'aux autres activités de conception qui nous semblent jouxter le dimensionnement. Nous voyons également comment interagissent ces différentes activités et quels sont les cycles qui apparaissent entre-elles. En conclusion nous essayons de faire une ontologie de l'activité de dimensionnement, en définissant les termes et les concepts qui semblent se dégager d'une pratique du dimensionnement, comme grandeur dimensionnante ou phénomène dimensionnant.

Dans une deuxième partie nous nous intéressons aux outils existant sur le marché, ou en recherche, qui aident au dimensionnement. Nous essayons de dégager des situations représentatives d'une utilisation d'outils de dimensionnement, et d'en faire des catégories. Fort de ces cas d'utilisations, nous dégageons 11 critères qui permettent de mesurer les 3 types de logiciels pour le dimensionnement que nous connaissons. Ces trois logiciels et les 11 critères sont synthétisés dans un crible qui permet de dégager une mesure et une direction pour l'amélioration de l'existant.

Enfin dans une troisième partie nous analysons les méthodes de développement logiciel existantes, et essentiellement le concept assez innovant de composant logiciel. Pour cela nous dégageons 4 concepts rattachés usuellement aux composants qu'il nous semblait indispensable de définir : le framework, le triptyque metaclasse-classe-instance, les différences entre un objet et un composant, et les deux états usuels d'un composant Design-Time Runtime.

Dans un deuxième chapitre nous présentons la méthodologie sous jacente à un dimensionnement assisté par ordinateur, en introduisant les entités que manipulera le dimensionneur.

Nous commençons par donner une description générale de la méthode, et voyons ensuite la place centrale que prend la modélisation dans une telle méthodologie.

Cela nous conduit à définir le composant COB (Computational OBject), essentiel dans notre méthodologie. Le COB est l'artefact informatique représentant l'aspect calculable de la modélisation.

Nous introduisons ensuite le composant de génération appelé gen-X, qui est le pendant du COB mais pour le calcul symbolique. Le dernier composant est le composant d'optimisation, qui représente tous les algorithmes d'optimisation pouvant résoudre le problème central du dimensionnement : trouver les dimensions qui satisfont le cahier des charges.

Cette méthodologie nous a conduit à développer une technologie composant. Nous avons alors implanté un prototype, afin de valider la faisabilité et l'efficacité de cette technologie. Le troisième chapitre est la présentation détaillée de l'architecture de ce logiciel.

Nous présentons un aperçu plus concret du fonctionnement du générateur de COB. Nous détaillons l'optimiseur, qui permet de définir le problème central du dimensionnement de façon conviviale pour l'utilisateur. Nous insistons sur l'environnement, qui permet de sélectionner un algorithme d'optimisation et de le configurer en utilisant une technologie présentée au chapitre précédent.

Nous présentons enfin la façon dont cet environnement interagit avec l'extérieur. Il utilise des passerelles, des classes externes. Pour bien comprendre la pertinence des composants implémentés dans ce prototype, nous illustrons deux cas d'utilisations emblématiques et très fréquemment rencontrés chez les utilisateurs du prototype : le cycle direct qui est le plus simple, et un raffinement, le cycle incrémental.

Dans un dernier chapitre nous présentons deux cas d'étude de dimensionnement de dispositifs industriels. Nous détaillons les difficultés rencontrées et les résultats obtenus en suivant notre méthodologie.

Nous présentons deux autres cas d'étude qui, tout en suivant la même méthodologie, nécessitent les nouvelles capacités de modélisation proposées par le prototype. Il s'agit des équations implicites et des équations différentielles.

Enfin, dans la conclusion, nous dégageons trois points. En premier lieu, le formalisme des équations est un bon formalisme de la connaissance.

Deuxièmement, le prototype fonctionne. De plus, la méthode et ses concepts sont facilement appréhendés par les concepteurs.

Mais le logiciel devra prendre sa place dans les services de conception des entreprises. Seule son industrialisation nous permettra de savoir si le dimensionnement tel que nous le proposons répond aux attentes des concepteurs.

CHAPITRE PREMIER - ETAT DE L'ART

« Du savoir extrême à la
connaissance vulgaire, la différence
est nulle. »
Bataille (Georges)

I- Etat de l'art

Dans ce chapitre nous présentons une synthèse des analyses que nous avons faites sur trois axes importants. Le premier axe correspond à la compréhension de la place que prend le dimensionnement au sein d'une entreprise industrielle, le deuxième axe correspond à la détection de lacunes dans les logiciels qui interviennent dans le dimensionnement, et le dernier axe correspond à l'étude des méthodes de développements logiciels qui permettent de faire collaborer des ingénieurs de tous les métiers sans nécessiter trop d'expertise informatique.

Pour commencer il est nécessaire de poser une définition du dimensionnement.

« Le dimensionnement c'est l'acte de trouver des valeurs aux grandeurs qui caractérisent un dispositif, de sorte que toutes ces grandeurs, aussi bien caractéristiques qu'induites par l'environnement, satisfassent les contraintes du cahier des charges ».

Cette définition possède plusieurs points d'entrée. Premièrement, elle pose le dimensionnement comme un acte, c'est à dire qu'il ne faut pas perdre de vue que le dimensionnement va faire intervenir différents acteurs, dans des processus plus ou moins complexes. Deuxièmement, cette définition donne également l'objet du dimensionnement, à savoir trouver les valeurs pour les grandeurs qui définissent le dispositif, ce qui revient à dire, même si cela semble trivial, que l'environnement du dimensionnement est articulé autour du couple grandeur-valeur. Troisièmement cette définition introduit la notion d'environnement qui est la pierre triangulaire de l'activité de dimensionnement, elle renferme des notions telles que les lois de la physique, le monde réel, des multi-univers, la notion de marché, de productivité [Blanco-98, Mer-98]. Quatrièmement, cette définition introduit la notion de cahier des charges qui est l'ensemble des contraintes qui s'appliquent sur les grandeurs qui définissent le dispositif et celles qui sont induites. Tous les concepts présents dans la définition sont résumés dans un synoptique (Figure Représentation schématique du dimensionnement) qui permet de visualiser les relations entre elles.

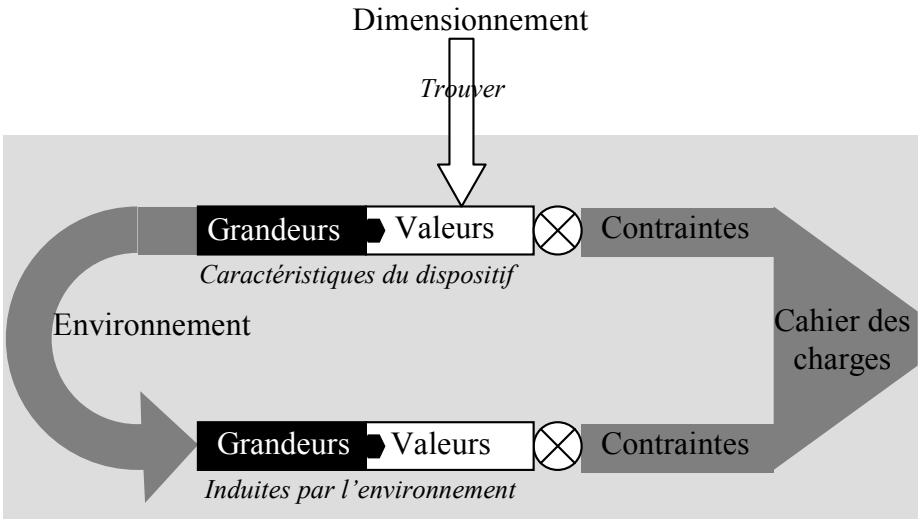


Figure I-1Représentation schématique du dimensionnement

Pour dimensionner, on ne peut pas se contenter de poser les valeurs pour les grandeurs caractéristiques et déclarer que le dimensionnement est fait. Il est raisonnable d'espérer trouver un processus calculable qui permette de trouver un jeu de valeurs pour les grandeurs caractéristiques qui satisfasse les contraintes associées. En revanche les valeurs des grandeurs de sortie sont induites par l'environnement, et il n'est pas trivial de trouver une inversion du système qui donne les grandeurs de sortie en fonction des grandeurs d'entrée. Par conséquent l'acte de dimensionnement n'est pas un acte trivial, et réaliser un dimensionnement revient à résoudre un problème difficile.

Les entreprises industrielles sont confrontées au problème de dimensionnement quand elles conçoivent leur produit. Il est donc important de se demander si l'on peut améliorer les processus qui conduisent au dimensionnement dans les entreprises, et le cas échéant de proposer des solutions d'amélioration.

Si l'on observe ce qu'il s'est passé dans la conception pour l'amélioration de processus, on constate que certaines de ces activités ont bénéficié d'outils logiciels qui les ont grandement améliorées. Dans l'ordre d'apparition, le CAD (Computer Aided Design, Conception Assistée par Ordinateur) qui consiste à définir le produit essentiellement par le dessin industriel. Ensuite est venue le CAE (Computer Aided Engineering, Ingénierie Assistée par Ordinateur), qui consiste à simuler le produit pour vérifier la satisfaction du cahier des charges. L'essentiel des produits qui adressent ce marché sont des logiciels fondés sur des méthodes de type éléments finis (ou élément de frontière, ou élément simple) qui reposent sur la géométrie du dispositif. Enfin, le PIDO (Process Integration and Design Optimisation, Intégration des Processus et Conception par Optimisation) qui consiste à enchaîner les simulations de CAE, et à effectuer des optimisations sur les modèles de simulation de CAE.

Ces secteurs de marché sont en forte croissance, révélant le fort besoin industriel en solutions pour améliorer les processus de la conception. Le PIDO, et les produits qui adressent ce marché, ne sont-ils pas les solutions d'amélioration que nous souhaitons pour le dimensionnement ?

Pour pouvoir répondre à cette question un préalable nécessaire est d'observer les relations qui existent entre les différentes activités du dimensionnement, pour savoir dans quelle mesure il bénéficie d'un dimensionnement par optimisation sur les modèles de CAE.

I.A le dimensionnement dans la conception

I.A.1- LES ACTIVITES DE LA CONCEPTION

Sans chercher à définir complètement la conception, on peut dire qu'elle est constituée d'un certain nombre d'activités qui toutes conduisent au même objectif : la définition précise du produit à concevoir [Bel Habib-00, Blanco-98, Harani-97, Mer-98, Trichon-91]. Ces activités sont :

- Le choix de structure. Cela consiste à choisir les formes, les technologies, les solutions. Tous les choix qui changent la nature du dispositif appartiennent au choix de structure. Par exemple, choisir entre un contacteur, bistable ou monostable pour remplir une fonction, est du ressort du choix de structure [Lechevalier-97].
- Le dimensionnement. C'est l'activité que nous avons décrite en introduction. Par exemple trouver les dimensions d'un déclencheur qui rentre dans un volume donné,

tout en respectant une tenue au choc et en étant le plus rapide possible [Bergeon-98, Wurtz-96].

- L'analyse. Les deux étapes précédentes proposent des solutions pour le problème de conception. La fonction de l'analyse est de vérifier que les solutions proposées sont viables. En tant que telle elle ne concourt pas directement à la résolution du problème de conception, ce n'est donc pas une activité intrinsèque à la conception, mais bien une activité qui améliore le processus de conception en réduisant le coût des erreurs de dimensionnement. La fonction de l'analyse consiste donc à simuler le plus précisément possible le dispositif pour vérifier son bon fonctionnement, et la satisfaction du cahier des charges. Cette activité a beaucoup apporté à la conception en réduisant les temps de mise sur le marché des produits (une simulation prend moins de temps qu'un prototype), en réduisant les coûts de fabrication de prototypes. Il est intéressant de noter alors l'engouement industriel pour les solutions logicielles qui assistent le concepteur dans cette tâche.
- Les activités de la production. Il est utile de concevoir un produit qui fonctionne, mais il faut aussi concevoir un produit qui peut être industrialisé. Les activités de production ramenées en conception, consistent à introduire des contraintes ou des préférences sur les solutions techniques, de sorte que le produit soit réalisable.
- Les activités du marketing. Le marketing ramené en conception revient à définir des éléments du cahier des charges de la conception afin de définir un produit qui soit conforme aux exigences du marché.
- La conception est une activité de négociation entre les intérêts de chacun des intervenants [Blanco-98], le résultat de la conception est un compromis entre les différentes contraintes issues des différentes activités. Le choix de structure est un compromis qualitatif, tandis que le dimensionnement est un compromis quantitatif.

Synthétiquement on peut dire que les activités de marketing et de production vont donner un cahier des charges technique pour le produit. Ce cahier des charges technique va servir à l'activité de choix de structure pour proposer une solution qualitative du produit. Cette solution qualitative du produit sert pour le dimensionnement, qui va chercher à la quantifier. Une fois le produit qualifié et quantifié, il reste seulement à analyser la solution pour vérifier qu'il respecte les exigences finales du cahier des charges technique.

Ce qui peut se résumer dans un schéma synthétique dans la figure ci-dessous.

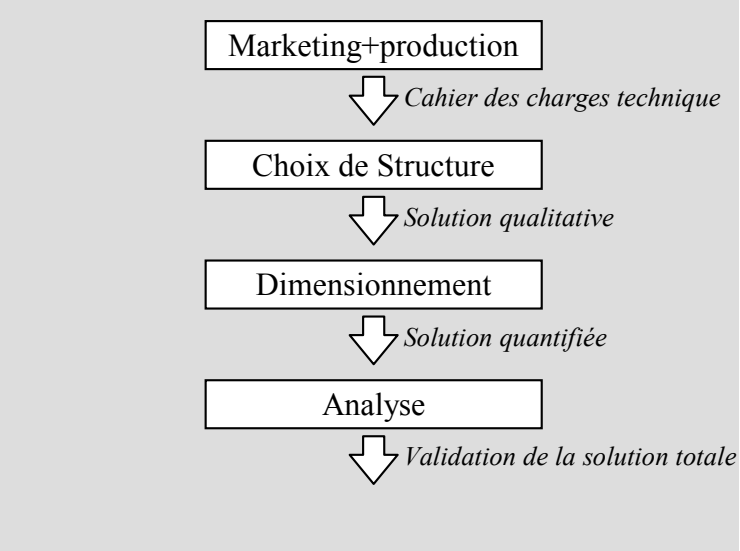


Figure I-2 : Représentation schématique de la conception.

I.A.2- APPROCHE SYSTEME

I.A.2.a- ISOLATION, FUSION DES ACTIVITES DE LA CONCEPTION

Lorsque l'on cherche à intervenir sur l'amélioration du processus de conception [Trichon-91], on peut adopter deux positions différentes selon sa propre culture : soit on va souhaiter mieux isoler les fonctions de la conception pour améliorer la maintenabilité de cette activité, soit on va chercher à fusionner les activités pour améliorer la « productivité » de cette activité.

Isoler les fonctions cela consiste à définir clairement les interactions entre les activités sous la forme d'un « contrat », et à préserver la liberté de chacune, tant qu'elle se conforme au « contrat » qui la lie aux autres. Fusionner les fonctions, c'est abolir les frontières entre les activités, et permettre à chacune d'imposer des manières de faire.

La séparation des fonctions de la conception permet d'améliorer la maintenabilité, et la compréhension, car l'isolation contractuelle des fonctions permet d'intervenir sur l'une sans mettre en danger les autres. Ce cas se présente si l'on souhaite par exemple investir dans un nouvel outil pour le CAE, on est certain que ce nouveau logiciel ne va pas intervenir sur les activités du marketing. Mais cela intervient également si l'on observe l'évolution de la simple notion d'équation au cours du temps [Taton-95-1,2,3,4], l'effort de symbolisation et de réduction de ces symboles permet d'aller sans cesse plus loin dans la réflexion. En observant l'écriture d'une équation au Moyen-Age on comprend que leur résolution n'était accessible qu'au plus grands esprits. Le processus qui a conduit aujourd'hui à l'écriture d'une équation presque universelle (bien qu'il reste quelques formalismes résistants en chimie ou en calcul différentiel avec les notations de Monge), a laissé de coté bien des notations qui présentaient chacune des avantages, pour élire celle d'aujourd'hui qui n'est pas sans défaut. Aujourd'hui, utiliser la même écriture pour écrire des équations thermiques, ou des équations électro-magnétiques ne choque personne, c'est pourtant le résultat d'une séparation des fonctions qui a commencé il y a longtemps.

La fusion des fonctions est une solution qui présente l'avantage de l'efficacité théorique. En effet, l'abolition de toute forme de contrat entre les activités de la conception

permet de mieux faire interagir les activités, et il n'y a pas de temps perdu à se conformer au contrat.

La séparation des fonctions peut alourdir le processus dans deux situations : si les contraintes industrielles conduisent à revoir régulièrement, en profondeur les contrats entre les activités, voire à devoir redéfinir la séparation en activités, et si le temps passé dans chacune des activités devient du même ordre que le temps passé à se conformer aux contrats.

La fusion des fonctions est limitée par l'intelligence humaine [Ganascia-96, Pitrat-95, Vignaux-91] : la complexité d'un système peut atteindre des seuils qui ne sont plus intelligibles par un humain ou un groupe d'humains. Dans ce cas il est obligatoire de procéder à une isolation des fonctions qui va permettre à un humain d'appréhender l'ensemble par « blocs », sans connaître le détail. Ces seuils cognitifs ne sont pas extrêmement élevés, il est usuel [Pitrat-95] de donner le nombre de 7 items. Sans structuration (qui est une forme de séparation des fonctions) il est difficile d'atteindre une vue d'ensemble qui contient plus de 7 items. Cette valeur semble assez basse, néanmoins pour s'en convaincre il suffit d'essayer de comprendre ces deux phrases qui ont strictement le même sens (extraite de [Pitrat-95] page 12)

« Les souris que le chat que le chien que la grand-mère que le petit chaperon rouge que le loup guette va voir caresse a mordu chasse mangent le fromage. »
« le loup guette le petit chaperon rouge qui va voir la grand-mère qui caresse le chien qui a mordu le chat qui chasse les souris qui mangent le fromage »

La complexité de la conception est très certainement supérieure à celle de ces petites phrases qui déjà sont en limite de complexité pour l'entendement humain. Une fusion intégrale des fonctions au niveau de la conception est donc impensable. Reste à savoir si l'état actuel de la conception nécessite un effort dans le sens de la fusion ou de l'isolation des fonctions.

Aujourd'hui les activités de la conception existent, elles produisent leur effet, et concourent à la conception des produits techniques dans toutes les entreprises. Ces activités se sont agrégées au fil du temps pour répondre aux besoins industriels.

Un processus complexe qui s'est complexifié par agrégation d'activités et de fonctions, est le plus souvent désorganisé. En effet il est très peu probable qu'une organisation qui convient à un groupe d'activités, le soit encore avec l'adjonction d'un nombre sensible de nouvelles activités. C'est pourquoi nous faisons l'hypothèse que nos efforts pour améliorer le processus de conception doivent se faire au maximum en privilégiant l'isolation des fonctions.

I.A.2.b- CYCLE DE LA CONCEPTION

La description linéaire des activités de la conception ne résout pas le problème de sa projection dans le temps. Il est nécessaire de connaître les durées des étapes, et le nombre de répétitions de chaque étape, en somme la séquence réelle de succession de chaque étape avec sa durée. C'est ce qu'on appellera le cycle de conception.

Si l'on cherchait à procéder linéairement, il faudrait être capable d'intégrer des phénomènes d'activités avales, dans une activité amont : par exemple des phénomènes qui sont liés au dimensionnement dans le choix de structure.

Si l'on cherchait, par contre-coup, à procéder de manière brutalement cyclique, il faudrait alors attendre la dernière étape pour se rendre compte que les décisions prises dans la première étape sont irréalistes. Et le temps investi dans les étapes intermédiaires s'avère inutile si le résultat de la dernière étape était prévisible. Il se trouve que le temps entre le

marketing et la production proprement dite, peut être long et coûteux puisqu'il peut faire intervenir l'ensemble du bureau d'études.

Il serait faux de croire que l'on peut faire tous les choix de structures sans faire intervenir des considérations qualitatives, Par conséquent il est nécessaire d'imbriquer rapidement les cycles de choix de structure et les cycles de dimensionnement. La rapidité de réponse du dimensionnement, ainsi que sa capacité à répondre à des structures partielles conditionnent la qualité des choix de structure.

Il faut donc que soient possibles des dimensionnement fréquents et rapides, en cours de choix de structure. Pour que soient possibles ces dimensionnements il faut pouvoir les effectuer sur une structure partiellement définie. Pour que ce dimensionnement soit rapide il est impératif qu'au cours du cycle de conception, on puisse procéder par modification des dimensionnements précédents.

I.A.3- LE DIMENSIONNEMENT : COMPROMIS, CAHIER DES CHARGES ET MODELES.

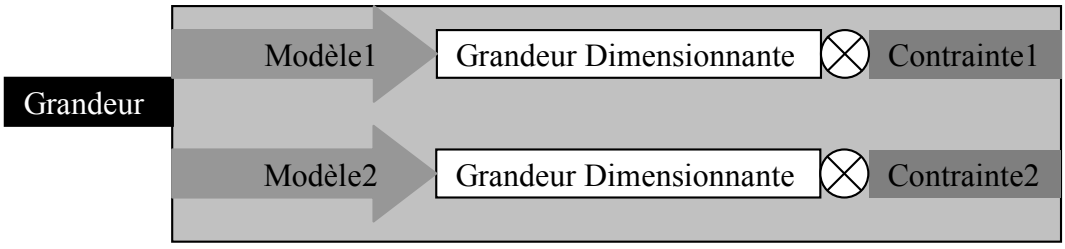
Le résultat de la conception est un compromis, qui va se préciser au cours des cycles. Le dimensionnement est le lieu de résolution de ces compromis, car c'est dans le dimensionnement seul que l'on peut quantifier les grandeurs en jeu.

I.A.3.a- DEFINITION

Pour comprendre ce qu'est un compromis en conception il suffit de prendre un exemple : sur un contacteur électromagnétique, la section de fer du circuit magnétique est dimensionnée au minimum par la quantité de flux qui doit passer dans cette section, et au maximum par le poids du dispositif. Cette affirmation n'est pas toujours vraie, elle dépend du contacteur, voire du point de fonctionnement du contacteur.

Un compromis (ci-dessous) en conception c'est un triplet constitué d'une grandeur du dispositif à dimensionner, et de deux phénomènes, dit dimensionnants, qui encadrent cette valeur. Dans notre exemple la grandeur est la section de fer et les deux phénomènes sont d'une part le poids qui augmente avec la section, et d'autre part la réluctance qui diminue avec la section.

Comment peut-on exprimer les compromis de sorte qu'ils puissent être résolus par un processus calculatoire ? Une façon de procéder consiste à exprimer les phénomènes dimensionnants dans un modèle approché, relié à un cahier des charges.



FigureI- 3 Représentation schématique d'un compromis.

Dans l'exemple de dimensionnement, le plus probable est que l'on modéliserait le poids du dispositif en fonction entre autres de la section de fer, et que l'on ajouterait dans le cahier des charges une contrainte sur le poids. Pour l'autre phénomène, on modéliserait l'ensemble du schéma réluctant [Roters-41] du dispositif, pour pouvoir quantifier l'idée que plus la section est grande plus le dispositif est performant. On voit bien ici la complexité de la modélisation pour le dimensionnement, car il ne suffit pas de calculer la réluctance du fer pour borner la section de fer. La réluctance du fer, qui dépend de la section de fer, va intervenir dans un autre compromis, par exemple le compromis Ampère-tours d'alimentation force de collage du contacteur.

I.A.3.b- TYPOLOGIE

Il existe quatre types logiques de base de compromis. Tous les compromis se ramènent à une composition de ces quatre types.

- Les compromis non bornés. Il s'agit des compromis dont un des deux phénomènes dimensionnants a été oublié. Ce cas est assez fréquent et un environnement de dimensionnement automatique doit permettre de révéler ce genre de lacune.
- Les compromis mal bornés. Il s'agit de compromis dont un des deux phénomènes dimensionnants est mal choisi. Par exemple dans la conception d'un contacteur, la section de fer peut être dimensionnée comme on l'a dit par la réluctance qui augmente, mais parfois elle est aussi dimensionnée par la saturation qui peut intervenir dans le fer. En fait selon les situations un des deux phénomènes peut intervenir, le phénomène dimensionnant est donc la fusion des deux phénomènes, n'avoir pas pris en compte la saturation, dans ce cas conduirait à un compromis mal borné. Un phénomène dimensionnant mal choisi peut être dû à une mauvaise modélisation du phénomène, qui conduit alors à des solutions fausses.
- Les compromis insolubles. Il s'agit d'un compromis ou d'un groupe de compromis qui sont contradictoires, et qu'il est donc impossible de résoudre. Un cas évident est celui d'un moteur qu'on veut d'une puissance supérieure à 1 kW, et d'une taille inférieure au centimètre cube. Ce genre de conflit est rarement facilement soluble.
- Les compromis imbriqués. Il s'agit de compromis qu'on ne peut pas relier directement à une contrainte, mais plus facilement à un autre compromis. Par exemple, pour la section de fer d'un contacteur un des phénomènes dimensionnants est la réluctance de ce dernier qui augmente avec la diminution de la section de fer. Mais il est difficile de ramener ce phénomène à une contrainte du cahier des charges. En revanche cette réluctance va être utilisée pour calculer par exemple la force motrice. Il sera alors possible d'exprimer une contrainte sur la force motrice.

I.A.4- CONCLUSION

L'apparition de la CAE, c'est à dire l'informatisation des activités de validation des solutions de conception, a permis de réduire les cycles globaux de conception et les coûts de conception par la réduction de l'étape de vérification. L'étape suivante, l'informatisation du dimensionnement permettra normalement d'obtenir de nombreux bénéfices pour la conception entière. Il peut s'agir de gains sur les coûts de conception par un dimensionnement plus rapide, mais aussi sur les coûts des produits conçus par un dimensionnement qui tient

compte de la contrainte économique. Il peut s'agir aussi d'une meilleure exploitation de la veille technologique par une réponse plus rapide aux évolutions technologiques.

Le dimensionnement est un point clef de la conception car c'est le lieu de résolution des compromis issue de la négociation entre les acteurs. Il est donc crucial d'instrumenter cette activité, pour lui donner les mêmes performances que celle de la CAE et du CAD aujourd'hui.

Les outils pour le dimensionnement ne doivent pas être les outils de la CAE ni du CAD, car cela ne va pas dans le sens d'une meilleure isolation des fonctions, et donc d'une meilleure réactivité. Cela n'exclut pas l'utilisation des outils de la CAE par les concepteurs, mais cela exclut la fusion des outils propres au dimensionnement et des outils propres à la CAE.

I.B Les outils de dimensionnement

Les outils qui sont utilisés aujourd'hui dans le dimensionnement ne nous semblent pas complètement adaptés à l'activité, car tous viennent d'activités connexes. On a vu la place que prenait le dimensionnement dans la conception, et on s'attend à ce que les outils du dimensionnement soient conçus pour s'adapter à ces situations.

I.B.1- LES CAS D'UTILISATION DES OUTILS DE DIMENSIONNEMENT

La position du dimensionnement dans la conception permet de définir un certain nombre de situations qui correspondent à des utilisations potentielles d'un environnement dédié au dimensionnement.

Pour pouvoir analyser les outils existant dans le dimensionnement, et pour pouvoir construire un crible, nous allons lister ces cas d'utilisation d'un environnement de dimensionnement, et nous verrons alors s'il existe un outil adapté à toutes les situations.

I.B.1.a- DES CAS D'UTILISATION

La notion de cas d'utilisation est une notion importée de l'informatique pour la conception de logiciel, nous avons repris ici cette notion, et nous l'utilisons pour construire notre crible d'évaluation des logiciels. La théorie qui englobe les cas d'utilisation en informatique est essentiellement UML [Uml] qui propose une méthode de conception de logiciel. Cette théorie ne présente pas d'intérêt direct pour notre sujet, nous avons seulement réutilisé le concept de cas d'utilisation, car il nous est familier.

I.B.1.i- PREDIMENSIONNEMENT D'UNE SOLUTION

On appelle prédimensionnement l'activité de dimensionnement qui consiste à résoudre le cahier des charges sur une connaissance lacunaire, incomplète du dispositif.

Cette activité correspond à une situation de conception classique où les activités de choix de structure sont en cours de définition des structures, et ont une vue encore globale de parties entières du produit.

On attend d'un prédimensionnement qu'il fournisse des ordres de grandeurs raisonnables qui permettent de valider les choix de structure. A défaut on attend du

prédimensionnement qu'il soulève des problèmes dans la conception de la structure, en révélant des compromis mal bornés ou non bornés.

I.B.1.ii- DIMENSIONNEMENT D'UNE GAMME

Une forme particulière du prédimensionnement est la prise en compte d'une gamme de dispositifs pour couvrir une plage de fonctionnement, mais également pour une production à différentiation retardée. L'enjeu industriel est important car concevoir l'ensemble d'une gamme rend possible des effets d'échelles sur la production, certaines parties des dispositifs pouvant être communes.

Par exemple si l'on veut dimensionner une gamme de capteurs de courant, il faut disposer d'un paramètre qui permette d'ajuster le seuil de déclenchement d'un capteur, c'est à dire la valeur de courant à partir de laquelle celui-ci va signaler la présence de courant. Il en résulte un courant seuil minimum et un courant seuil maximum. Le cahier des charges de la gamme consiste donc à fixer le courant seuil minimum du plus petit dispositif à la valeur minimale de la plage de fonctionnement, puis fixer la valeur maximale du courant seuil d'un dispositif à la valeur minimale du courant seuil du dispositif suivant dans la gamme. Ainsi de suite jusqu'à fixer la valeur maximale du seuil de courant du dernier dispositif à la valeur maximale de la plage de fonctionnement.

Ainsi le dimensionnement réalisé tient compte de tous les dispositifs pour choisir la segmentation de la gamme. Cette technique peut conduire à une solution réduisant le nombre de dispositifs dans la gamme.

I.B.1.iii- COMPARAISON DE SOLUTIONS

Vaut-il mieux être riche et laid ou pauvre et beau ? Dans le cas général il n'y a pas de réponse, mais seulement sur un cas particulier. Il en va de même avec certains conflits de choix de structure, on ne trouve pas de critère qui permette de trancher entre les avantages et les inconvénients de solutions techniques. En effet il est difficile de comparer deux solutions techniques différentes, le plus souvent chacune présente plusieurs avantages et plusieurs inconvénients, tous différents. La seule possibilité de comparer les solutions est de se fixer une fonction objectif (par exemple le prix, ou le volume), et de comparer les solutions optimales de chacune des technologies par rapport à cette fonction objectif.

Ainsi donc, l'activité du choix de structure va fournir des solutions technologiques, et un cahier des charges commun à celles-ci. L'activité de dimensionnement consiste alors à trouver les solutions optimales pour chacune des technologies, et comparer les fonctions objectifs.

I.B.1.iv- VEILLE TECHNOLOGIQUE

La veille technologique est un problème industriel important qui dépend de technologies développées par d'autres domaines. On peut citer par exemple, les caractéristiques des matériaux, les qualités des aimants etc.

Il en résulte que les technologies du génie électrique ne sont pas bonnes ou mauvaises indépendamment du reste du monde, mais se définissent par rapport à un contexte industriel. Ce contexte est susceptible de changer avec le temps. La veille technologique consiste donc à surveiller les dépendances technologiques, et surtout à détecter les seuils.

Les seuils technologiques ne peuvent pas être exprimés directement ni simplement, il ne suffit pas de dire par exemple que quand l'aimantation atteindra telle valeur il y aura un seuil

technologique. En effet ils sont interdépendants, et dépendent des solutions technologiques choisies par une entreprise, de sa capacité de production etc. La meilleure façon de faire de la veille technologique est donc de répéter le dimensionnement en ne changeant que les valeurs qui ont changé avec le temps, et de voir si le cahier des charges est satisfait.

Ainsi donc un environnement de dimensionnement doit pouvoir répéter des conceptions dans le temps et par d'autres intervenants que le concepteur initial qui a mis le modèle au point.

I.B.1.v- MODIFICATION D'UN MODELE

Lors d'une conception il est fréquent de procéder par touches successives, les solutions technologiques se précisent avec les itérations de conception. Lors de la mise au point du premier jet du dispositif en conception il est acceptable que le passage de la solution dans un environnement de dimensionnement soit assez long. En revanche le temps d'intégration des modifications du dispositif dans l'environnement de dimensionnement est critique pour la durée d'un cycle de conception. Cette durée d'intégration des modifications doit être la plus faible possible.

Il en résulte qu'un environnement de dimensionnement doit trouver une forme de représentation des connaissances stable pour les modifications usuelles de la conception. Une représentation géométrique n'est pas adaptée à ce genre de besoin, car une géométrie n'est pas l'expression directe des choix que l'on a fait, mais plutôt la représentation finale de ces choix. Pour intégrer, par exemple, un changement de nombre d'encoches les outils d'éléments finis recalculent la géométrie avec le nouveau nombre d'encoches.

I.B.2- CRIBLE D'EVALUATION DES OUTILS POUR LE DIMENSIONNEMENT

De tous les cas d'utilisation précédemment définis on peut tirer un certain nombre de critères qui vont servir à comparer les principaux logiciels existants (Tableau page I-22).

I.B.2.a- LES LOGICIELS

Les différents logiciels que l'on peut voir utiliser dans un bureau d'études peuvent être regroupés dans trois catégories.

I.B.2.i- LES LOGICIELS DU PIDO

Rappelons que le PIDO est l'acronyme anglais pour le Process Integration Design Optimisation. C'est le nom d'un marché industriel dont les produits sont des logiciels qui se caractérisent par une structure centrée sur les logiciels éléments finis. Les logiciels éléments finis (au sens large, éléments de frontières, etc.) sont utilisés pour réaliser des simulations, tandis qu'un algorithme d'optimisation exploite ces résultats pour chercher des solutions meilleures.

I.B.2.ii- LES LOGICIELS GENERALISTES DU CALCUL

Les logiciels généralistes du calcul sont des logiciels qui se présentent sous la forme d'un environnement de programmation qui fournit des facilités de programmation. Le leader

dans ce domaine est Matlab [Matlab-96, Mathcad-98]. Les logiciels généralistes du calcul possèdent des modules (ou « toolbox » pour Matlab) qui permettent de réaliser certaines fonctions utiles pour le calcul. Il y a par exemple la « toolbox » pour l'optimisation qui va contenir des algorithmes d'optimisation. Il y a également des « toolbox » pour le calcul par élément fini, on peut citer à titre d'exemple un de ces acteurs : FEMLAB. Bien que Matlab dispose d'un module élément fini, et un module optimisation, il n'est pas un acteur du PIDO. En effet le temps de mise en œuvre de cette solution est bien trop élevé pour faire concurrence à un logiciel du PIDO.

I.B.2.iii- LES LOGICIELS GENERATEURS DE CODE

Les logiciels générateurs de code sont des logiciels qui vont prendre une description du dispositif sous forme d'équations, et générer le code pour l'exploiter avec de l'optimisation.

De la même manière que les logiciels généralistes du calcul ne sont pas des acteurs du PIDO, Matlab possède des modules qui permettent de faire de l'optimisation, du calcul formel et de la génération de code, pour autant Matlab ne peut pas être considéré comme un logiciel générateur de code car pour pouvoir exploiter conjointement ces trois fonctionnalités, il y a un effort de programmation tel que l'on peut estimer qu'il s'agit d'un nouveau logiciel fondé sur Matlab, et non Matlab lui-même.

L'idée de générer le code de programmation à partir d'une information minimale (ce qui améliore la maintenabilité des modèles) n'est pas neuve, [Gentilhomme-91, Trichon-91], et il faudra attendre GENTIANE [Gerbaud] pour voir cette capacité de génération de code exploitée sur des équations mathématiques, et utiliser le calcul formel [Maple-96, Macsyma-95] pour enrichir la connaissance du modèle. Enfin PASCOSMA, [Wurtz-96] a utilisé le calcul formel des dérivées et la génération de code pour exploiter les modèles dans l'optimisation.

I.B.2.b- CRITERES

I.B.2.i- INCREMENTAL

La conception exige du dimensionnement qu'il supporte une définition incrémentale, c'est à dire qu'il faut pouvoir spécifier le problème à résoudre à partir d'un problème à résoudre proche, et en le modifiant. Pour cela il faut que les données utilisées se prêtent à ce genre de manipulation.

Les logiciels qui sont incrémentaux sont ceux qui ont une méthode de description qui permet la modification sans reprendre tout le travail. Dans ce crible il nous a semblé que les logiciels fondés sur une description de la géométrie par lignes et points ne sont pas incrémentaux. Ceux qui se fondent sur une description du dispositif intégralement géométrique ne sont pas non plus incrémentaux car il y a une partie importante de l'information qui n'est pas géométrique, les propriétés des matériaux, les coûts, les contraintes de fonctionnement etc. Les dimensionnement faits à partir d'outils fondés sur la géométrie doivent obligatoirement s'adjoindre le concours d'un nouvel outil, qui lui doit contenir les informations non géométriques. Dans ce cas l'information étant éclatée il est impossible d'offrir une conception incrémentale.

I.B.2.ii- LACUNAIRE

Le dimensionnement est d'autant plus intéressant qu'il est capable de proposer des solutions avec des informations lacunaires, le dispositif étant partiellement décrit, le reste du fonctionnement pouvant être par exemple supposé idéal (perméabilité magnétique infinie).

Les logiciels fondés sur la géométrie ne peuvent pas prétendre à un dimensionnement lacunaire car il n'y pas de possibilité de définir un problème dans ces outils sans totalement spécifier la géométrie.

I.B.2.iii- MULTI DISPOSITIF

On a vu que le dimensionnement de gamme, permettait de dimensionner plusieurs dispositifs d'une même gamme. Il est donc important que le logiciel soit capable de prendre en compte des modélisations multi dispositifs.

I.B.2.iv- MULTI POINT DE FONCTIONNEMENT

Beaucoup de dispositifs ont un fonctionnement complet qui se déroule sur plusieurs points de fonctionnement (à vide, en charge, fermé, ouvert). Il est très probable que certains des phénomènes dimensionnants qui interviennent dans le dimensionnement de ces dispositifs s'appliquent pour un point de fonctionnement, tandis que d'autres phénomènes s'appliquent sur un autre point de fonctionnement. Il est donc nécessaire que les outils de dimensionnement supportent des modèles qui sont multi point de fonctionnement.

I.B.2.v- RAPIDE

La vitesse de réponse des outils de dimensionnement est importante. Si l'on souhaite que les concepteurs réalisent de nombreux dimensionnements au cours d'un cycle de conception il est nécessaire que les dimensionnement se fassent très rapidement.

La vitesse n'est pas un critère unique il se divise en plusieurs sous critères selon les types de logiciels. Pour les logiciels du PIDO le temps de simulation pour un problème d'une complexité raisonnable en conception multiplié par le nombre de fois où il est nécessaire de faire appel à cette simulation peut rendre inutilisable cette solution. Certaines simulations de dispositif en conception peuvent atteindre quelques heures.

Pour les logiciels généralistes du calcul, le temps le plus important est celui passé à maintenir les modèles réalisés. En effet, les problèmes atteignant des complexités suffisantes nécessitent parfois l'utilisation de code numérique (par exemple une résolution implicite, ou une équation différentielle). Dans ce cas la part du code qui ne sert qu'à utiliser les routines numériques est importante et source principale de problème de maintenance. En effet, dans [Singh-92], il apparaît que quand l'on cherche à faire de l'optimisation sur des dispositifs techniques, la première cause d'échec et d'erreur est le calcul numérique des dérivées. La réponse la plus facile à apporter consiste à programmer en plus du modèle, également les dérivées du modèle pour aider les algorithmes d'optimisation. Dans ce cas par exemple la part de code purement numérique représente une part très supérieure au code qui permet de calculer le modèle simplement. Par exemple pour un modèle de 50 équations qui contiendrait

20 paramètres de sortie et 20 paramètres d'entrée il faudrait coder 20*20=400 dérivées partielles. Il y a donc 8 fois plus de dérivées que d'équations. Par ailleurs un changement dans le modèle induit des changements extrêmement complexes dans le calcul des dérivées. En effet il suffit d'introduire une dépendance entre un paramètre intermédiaire et un paramètre d'entrée pour que toutes les dérivées de tous les paramètres qui dépendent de ce paramètre intermédiaire soit à recalculer. En génie logiciel, une estimation courante consiste à dire que le temps de maintenance d'un code est égal au temps passé à le développer. Dans notre cas, le code est d'une qualité exécrable pour les canons de la programmation, en effet une partie importante (voire majoritaire) du code peut se déduire d'une autre partie du code. Il est donc probable que le temps de maintenance de modèles codés dans un logiciel généraliste du calcul est très élevé.

Pour les logiciels générateurs de code, le temps qui compte est celui de génération du code. En effet on peut espérer que le temps de calcul sera plus rapide que pour les logiciels du PIDO car les méthodes de calcul utilisées sont moins puissantes, mais plus rapides. On peut également espérer que les temps de maintenance seront beaucoup plus courts que ceux pour les généralistes du calcul, puisque le code se réduit au maximum à la part qui compte pour l'utilisateur, les codes de calcul des dérivées, et des normalisations, et des connexions aux algorithmes d'optimisation sont générés et donc non maintenus par l'utilisateur. De plus, la technique du générateur permet à l'utilisateur d'être sûr de ne pas faire d'erreur de programmation. Les équations données sont codées suivant un processus entièrement automatisé qui est le même pour tous les modèles et pour tous les utilisateurs. Il est impossible de garantir que ce processus est sans bugs. En revanche on peut garantir que les bugs trouvés dans le code généré, peuvent être corrigés dans le processus générateur par l'éditeur du logiciel, et par conséquent qu'ils ne se reproduiront plus. La maintenance que doit assurer le concepteur, est donc réduite à la maintenance du modèle sous formes d'équations.

I.B.2.vi- STATIQUE

La nature des dispositifs que l'on peut concevoir est un critère important des outils de conception. Ainsi un dispositif dont on va modéliser le comportement statique pourra être conçu par les outils remplissant ce critère. On appelle modèle statique un modèle où le temps n'intervient pas dans un calcul sous forme différentielle. C'est à dire que tous les phénomènes sont exprimés sous forme intégrale.

I.B.2.vii- IMPLICITE

Certains dispositifs sont fondamentalement implicites. On appelle modèle implicite un modèle qui ne peut pas exprimer les grandeurs induites du produit à partir de ses grandeurs caractéristiques. En revanche le modèle va utiliser une grandeur intermédiaire, qu'il va utiliser comme pivot pour exprimer les grandeurs induites.

Typiquement, tous les cas de résolution d'un circuit électrique ou équivalent, font souvent intervenir des modèles implicites. Par exemple le cas d'une pompe dont on connaît la caractéristique (pression en fonction du débit), et un circuit de tuyauterie dont on connaît également la caractéristique (pression en fonction du débit). Résoudre un tel circuit, est usuel en ingénierie, et consiste à calculer l'intersection des deux caractéristiques. Cette méthode est la même en électricité, en magnétisme etc. Si les caractéristiques sont linéaires, alors l'intersection des deux courbes est calculable, il n'y a pas lieu alors de considérer que le modèle est implicite. En revanche, si la caractéristique n'est pas linéaire, l'intersection des

deux courbes est moins facile à calculer. On peut même ajouter que les cas où il est possible de calculer l'intersection dépendent fortement de la modélisation des deux caractéristiques.

Pourquoi ne pas profiter des cas où l'intersection serait calculable ? Il y a deux raisons pour cela. La première est que ces cas semblent rares, et que surtout, il faut faire un effort non négligeable pour résoudre cette intersection, et que cet effort est à refaire si on fait un changement dans la modélisation d'une des deux caractéristiques, le problème étant encore plus criant s'il y a plusieurs points de fonctionnement. La seconde raison est qu'au cours des conceptions que nous avons réalisées au cours de la thèse avec différentes personnes dans le rôle de modélisateurs, nous avons n'avons jamais trouvé de solution calculable pour l'intersection et surtout parce que la modélisation, voire le dispositif changeait au cours de l'étude.

I.B.2.viii- DYNAMIQUE

De même que pour le critère implicite, le critère dynamique caractérise des dispositifs dont le cahier des charges fait appel à des notions temporelles non facilement calculables. Bien souvent cela signifie qu'il doit y avoir une résolution d'une équation différentielle numériquement.

Beaucoup de dispositifs sont dynamiques. Tous les dispositifs qui ont un mouvement et dont le régime transitoire est dimensionnant. Tous les dispositifs dont le régime permanent ne peut pas être approximé correctement par les premiers harmoniques.

I.B.2.ix- MULTI PHYSIQUE

Les domaines de la physique ne sont pas des frontières qui ont un sens pour la conception. Ce n'est pas parce qu'un dispositif est essentiellement électrotechnique qu'il ne faut pas introduire des considérations de mécanique. Il est évident que l'ingénieur-concepteur électrotechnicien connaît et applique souvent le principe fondamental de la dynamique. Bien souvent néanmoins les considérations mécaniques ne sont pas très poussées, mais il est plus important de pouvoir faire intervenir de la mécanique dans la conception d'un dispositif électrotechnique que de pousser très loin les considérations magnétiques.

I.B.2.x- ECONOMIQUE

L'économie est presque un domaine de la physique pour le concepteur tant sa place dans la conception est cruciale. Les compromis de la conception sont souvent résolus par une quantification des contributions sur le prix des différents phénomènes. En effet en physique, il est interdit de comparer les Ampère et les Newton, et la seule grandeur physique qui relie toutes les autres est l'énergie. Malheureusement l'énergie n'a pas de sens pour les activités du marketing. Il reste que dans une entreprise à vocation commerciale par essence, les seuls éléments qui relient les différentes activités sont les considérations économiques.

I.B.2.xi- PRECISION UNIFORME

Nous avons vu que tous les logiciels s'appuyaient sur une modélisation du dispositif soit analytique soit par méthode numérique fine (éléments finis, éléments de frontière etc.). La notion de précision d'un modèle n'est pas très clairement définie, et il semble qu'une définition assez consensuelle serait de définir la précision d'un modèle comme l'inverse de l'écart entre la valeur d'une grandeur simulée et la grandeur réelle en éliminant les bruits perturbateurs. Quelle que soit la définition de la précision, il manque une notion importante pour la conception : la précision uniforme.

Nous appelons dans ce rapport « précision uniforme » la norme de la précision et de la précision de la dérivée appliquée sur tout l'espace des solutions. La précision sur l'espace des solutions est la moyenne des précisions sur tout l'espace des solutions. La précision sur la dérivée est la précision sur le calcul des dérivées à partir du modèle.

La précision uniforme est la précision qui compte en dimensionnement, pour deux raisons : on ne s'intéresse pas au dispositif sur un point de fonctionnement donné, mais sur l'ensemble des points de fonctionnements possibles qui représente l'espace des solutions. Par conséquent une estimation raisonnable de la précision d'un modèle pour le dimensionnement est la pire précision du modèle appliquée sur une des solutions de l'espace des solutions. Le maximum des précisions n'est autre que la norme Infini sur l'espace des solutions. La seconde raison qui permet de préférer la précision uniforme comme précision pour le dimensionnement est que l'on ne s'intéresse pas exclusivement aux valeurs des grandeurs, mais également à leur variation, et donc aux tendances d'évolution. La variation d'une grandeur se ramène à sa dérivée.

Les logiciels de simulation utilisés dans le PIDO ont une précision uniforme très mauvaise, en revanche ils ont une précision simple (précision usuelle) très bonne. En effet, fondés sur un maillage, ou équivalent, les logiciels de simulations souffrent d'un bruit numérique dû au maillage. Ce bruit est à « fréquence élevée » c'est à dire que sa dérivée est importante. En effet si l'on remaille à chaque calcul, il se peut que pour deux calculs très proches il y ait une différence de maillage. Disons que pour une dimension X du dispositif qui change, nous allons observer la valeur d'une grandeur Y simulée. Il existe une valeur x0 de X et une valeur delta telle que entre la simulation à X=x0 et la simulation à X=x0+delta, le maillage sera différent. Alors l'écart de valeur pour Y est une constante qui ne dépend pas de delta, et dans ce cas la dérivée de Y par rapport à X est infinie. Ce phénomène est inhérent à tous les calculs fondés sur une discrétisation de l'espace, la dérivée d'un phénomène discret contient des valeurs infinies. Si l'on ne remaillait pas à chaque calcul, alors il faudrait « tirer » sur le maillage, et c'est la précision simple qui pourrait devenir grande.

Les logiciels généralistes de calcul n'ont pas de précision *a priori* non plus que les logiciels générateurs de code, sinon la précision de la machine. La précision du modèle est laissée au concepteur et à sa capacité à exprimer ses modèles dans un langage mathématique. Il faut noter néanmoins qu'il existe, dans l'électrotechnique du moins, des concepteurs qui savent exprimer en équations des modèles dont la précision simple est bonne, et la précision uniforme également.

La précision des dérivées est mauvaise si l'on ne calcule pas les dérivées formellement, ou du moins avec la même précision que le calcul lui-même. Par conséquent, comme il n'est pas usuel de calculer formellement les dérivées dans les logiciels généralistes de calcul, nous considérons que la précision uniforme est mauvaise pour les logiciels généralistes de calcul.

I.B.2.c- LE CRIBLE

	PIDO	Généraliste	Générateur
Incrémental	-	-	+
Lacunaire	-	+	+
Multi Dispositif	-	+	+
Multi Point	-	+	+
Rapide	-	-	+
Statique	+	+	+
Implicite	+	+	-
dynamique	+	+	-
Multi Physique	-	+	+
Economique	-	+	+
Précision Uniforme	-	-	+
Total	**8- 3+**	**3- 8+**	**2- 9+**

Tableau 1 Crible d'évaluation des logiciels

I.B.3- CONCLUSION

Le crible montre que les logiciels du PIDO ne sont pas adaptés au dimensionnement. Cela ne veut pas dire qu'ils sont mauvais mais seulement adaptés à des phases plus avales de la conception, éventuellement dans une phase de dimensionnement fin qui précède l'analyse, mais certainement pas pour aider les concepteurs dans leurs choix de structure.

Le crible ne permet pas de différentier franchement les outils Généralistes et les outils Générateurs de code. Ce ne serait pas le cas si les outils Générateurs de code pouvaient traiter les équations de type implicites et dynamiques. Dans ce cas le résultat serait de zéro moins et onze plus pour les Générateurs, ce qui ne laisse aucun doute à ce sujet, sachant que tous les critères de ce crible n'ont pas la même valeur et que nous ne les distinguons pas ici. En effet dans certains contexte un des critères peut devenir prépondérant : s'il faut un mois de travail pour « coder » le modèle, et un mois de plus pour coder les dérivées, alors la différence entre les logiciels capable de répondre au critère incrémental permettent d'économiser 1 mois de travail par modification du modèle.

Néanmoins il est légitime de se demander si la limitation des logiciels Générateurs faces aux équations implicites et aux équations différentielles est une limitation intrinsèque, ou historique. On peut d'ores et déjà répondre que les logiciels de type Générateur de code sont des prototypes développés en recherche visant chacun à démontrer un aspect du principe.

Pour l'instant aucun d'entre eux n'avait pour objectif de traiter les équations implicites et les équations différentielles dans un objectif de dimensionnement. Il est donc raisonnable d'espérer pouvoir combler cette lacune, et fournir aux outils Générateur de code, des fonctionnalités suffisantes pour vivre par eux même.

Le traitement des équations implicites et des équations différentielles est donc un enjeu qui peut faire passer les outils Générateurs de code, d'une problématique seulement recherche, vers une problématique conjointe recherche et industrielle.

I.C Les méthodes de développement logiciels

Nous avons souhaité fournir une assistance pour les concepteurs dans le dimensionnement. Cette assistance, à l'instar des autres domaines de la conception, doit être logicielle. Nous avons donc procédé à un rapide état de l'art des technologies de développement logiciel, pour pouvoir réaliser un logiciel qui réponde bien aux besoins du dimensionnement en même temps qu'il s'appuie sur les développements logiciels les plus récents. En effet, la marche de la science est plutôt logiquement tournée vers le progrès technique, et dans ce cas les nouvelles technologies informatiques apportent des avantages aux développeurs de logiciels.

Une des technologies informatiques qui peut contribuer de façon majeure à un logiciel pour le dimensionnement est la technologie des composants.

Nous n'avons pas souhaité analyser les technologies de la même manière que le ferait une thèse en informatique, mais plutôt de manière pragmatique, en analysant ce que nous pourrions tirer des technologies existantes, quels changements nous allons devoir faire pour les adapter à notre situation.

Nous avons vu que les logiciels générateurs de code permettaient de rapidement atteindre les performances des logiciels généralistes du calcul. Nous avons souhaité continuer dans cette voie prometteuse. C'est pourquoi nous avons été très sensibles aux technologies des composants qui permettent d'écrire un logiciel qui utilisera des codes qui n'existent pas encore. Il va de soi que nous analysons les technologies avec pour objectif de les utiliser dans un logiciel générateur de code.

I.C.1- DEFINITION

Un composant est un artéfact informatique qui permet d'exécuter une partie d'un logiciel. [Meijler-95, Nierstrasz-93, Udell-94]. Pour nous, cela revient à dire qu'un composant est une entité informatique (souvent un fichier) qui va contenir un morceau d'un logiciel. Le logiciel architecturé en composant va pouvoir utiliser ce morceau de logiciel pour se construire.

Il existe aujourd'hui trois grandes technologies qui font les composants.

COM est la technologie Microsoft pour les composants. Elle s'appuie exclusivement sur le système d'exploitation Windows de Microsoft. Néanmoins cette technologie bien que plus ancienne était au moment du choix de technologie que nous devions faire en pleine mutation vers ce qui est devenu la plate-forme « .NET ». La sortie tardive de la plate-forme « .NET» ne nous a pas permis d'évaluer cette technologie de composant, néanmoins il semble qu'il n'y ait pas de différence majeure avec celle développée par Java, mais que les seules différences soient commerciales.

Il y a la technologie CORBA, développé par l'OMG (Object Management Group) qui fournit une technologie pour faire des composants indépendants du langage (pourvu qu'il soit Objet), indépendants de la plate-forme, et indépendants de la machine. Cette technologie est assez lourde à mettre en œuvre, car elle a été conçue pour faire inter-opérer des logiciels hétérogènes écrits dans des langages différents qui tournent pour des raisons historiques sur des machines différentes. Il n'est pas utile de s'appuyer sur une telle technologie pour concevoir un logiciel mono langage, et mono machine. En revanche il est intéressant de fournir une compatibilité avec CORBA des composants que nous ferons, de sortes que d'autres logiciels, dans d'autres environnements pourront utiliser s'ils le souhaitent ces composants.

Il y a la technologie Bean issue de Java [Gosling-95], qui comprend les JavaBean et les Enterprise JavaBean (EJB). Cette technologie se fonde sur Java, un langage de programmation qui a un grand succès, par sa simplicité d'utilisation, et sa souplesse. En revanche Java n'est pas un langage particulièrement performant en calcul, mais il faut voir que la facilité de programmation de Java permet une plus grande souplesse dans la programmation, et qu'il n'est pas étonnant de voir un programme Java tourner plus vite qu'un programme écrit dans un des langages les plus rapides : FORTRAN. Bien évidemment, pour un programme optimisé en FORTRAN et le même en Java, il n'y a pas de doute quant à la vitesse de calcul. En fait le choix d'un langage de programmation pour un logiciel se fait selon des critères de génie logiciel, et FORTRAN est obsolète pour le génie logiciel. En revanche, pour un logiciel scientifique, il est important que le langage choisi puisse appeler des langages rapides pour les parties numériques (comme FORTRAN ou C).

I.C.2- ENVIRONNEMENT- FRAMEWORK

Un composant logiciel est un morceau de logiciel qui dépend d'un environnement. Dans le cas de Java cet environnement est la machine virtuelle Java qui va fournir tous les services nécessaires au fonctionnement du composant. Dans le cas des EJB l'environnement est également la machine virtuelle mais avec aussi le serveur d'application (appelé serveur J2E).

On appelle framework, ou établi, le morceau de programme qui permet de manipuler les composants. En effet, un morceau de programme seul ne peut pas être exploité. On a vu qu'un logiciel architecturé autour des composants, utilisait un composant comme une brique pour réaliser sa propre structure. Or un composant logiciel qui n'interagirait pas avec les autres composants est parfaitement inutile. En effet sans interaction, il n'y a pas de résultat possible, et donc *a fortiori* pas d'intérêt. Comment donc un composant peut-il interagir avec les autres composants, voire avec le reste du logiciel ? C'est l'objectif du framework que de prévoir une structure qui permette aux différents composants d'interagir.

Une façon de spécifier le framework est donc de définir les services dont peut avoir besoin le composant, et les services que doit rendre le composant. On essaiera de fournir une définition la plus générique possible pour ces services. Ainsi, pour les JavaBean, le service rendu par un composant se définit au travers de propriétés (de couples valeur-grandeur), et d'évènements. En quelque sorte un JavaBean est un composant qui a des propriétés, et qui peut renvoyer des évènements. De plus la définition des JavaBean prévoit une clause particulière pour les classes qui peuvent être graphiquement représentées (comme un bouton, une zone de texte). Dans ce cas, le framework est l'ensemble des classes qui permettent la définition des propriétés (les types primitif, la classe Object), les classes qui permettent les évènements (les classes mères de tous les évènements), et les classes qui permettent les objets visuels (Component etc).

Il y a une difficulté dans la définition du framework du composant JavaBean, qui vient de ce que le langage Java et les JavaBean sont apparus presque en même temps et que les concepteurs de ce langage ont intégré le framework des JavaBean à l'environnement de Java.

Néanmoins si nous voulons définir notre propre composant pour le dimensionnement, nous ne pouvons pas procéder de la même manière que Java, en intégrant le framework de nos composants à Java, il nous faudra par conséquent définir clairement quel est ce framework et le mettre à disposition.

I.C.3- METACLASSE, CLASSE, INSTANCE

Comment un programme peut-il manipuler un objet dont on ne connaît rien à l'avance ? C'est impossible, sans avoir un minimum de connaissances sur cet objet. Cette connaissance fait partie de la spécification du composant.

Les composants qui existent sont tous fondés sur le triptyque métaclasse-classe-instance. Ces trois entités sont reliées par un mécanisme d'instanciation. L'instance est le résultat de l'instanciation d'une classe (comme en Objet traditionnel), mais la classe est l'instanciation de la métaclasse. Cela veut dire qu'une métaclasse définit un ensemble de règles de construction à respecter, et les classes qui instancient cette méta classe doivent les respecter. Donc définir un composant c'est définir l'ensemble des règles de la métaclasse, instancier un composant c'est écrire une classe qui respecte ces règles.

Ces règles de bonne construction peuvent prendre plusieurs formes :

- Un mécanisme objet usuel. C'est à dire profiter des mécanismes comme l'héritage pour garantir une bonne construction des classes. Parmi les mécanismes que l'on peut signaler, il y a l'héritage d'une classe abstraite, l'implémentation d'interface.
- Un mécanisme de « design pattern » [Gamma-95]. Cela consiste à définir des règles non compilées qui permettent à l'environnement de reconnaître des « patrons » dans la construction de la classe. Par exemple en JavaBean, pour définir une propriété « Y » de type « double », il suffit de créer deux méthodes « public double getY() ; » et « public void setY(double) ; ». A charge pour l'environnement de reconnaître dans ces deux méthodes qu'il s'agit de la propriété Y.

I.C.4- OBJET VS COMPOSANT

On a depuis longtemps entendu parler des langages orientés Objet qui étaient sensés remplir les possibilité de réutilisation, capitalisation etc. Il n'y a pas de conflit entre les composants et les objets.

En technologie Objet on définit [Gamma-95] usuellement des patrons de conception de modèles objets (façade, singleton). Dans ce cas les composants seraient une hypertrophie d'un patron simple, le polymorphisme. En effet il suffit en programmation orientée objet de définir une classe abstraite, et de réaliser beaucoup de classes qui en héritent. Alors tous les programmes qui n'utilisent que les méthodes de la classes abstraite s'appliquent également pour toutes les sous-classes. C'est le principe des composants.

Il reste néanmoins un problème lorsque l'on n'utilise que le modèle objet. Comment instancier une classe qui hérite de la classe abstraite, et comment éventuellement, utiliser une classe qui n'existait pas au moment où on écrit le logiciel, mais dont on sait qu'elle héritera de

la classe abstraite ? La technologie composant tire tout son intérêt de la réponse à ces deux questions.

La technologie composant va donc fournir un mécanisme d'instanciation de classes qui hérite de la classe abstraite. Ces classes ne doivent pas être nécessairement présentes dans le logiciel au moment de sa conception.

De cette idée simple au départ, il semble que la spécification des composants s'est enrichie d'un ajout important, l'introspection. En effet, on a vu que dans la définition d'une métaclasse il y avait deux types de règles qui entraient en jeu : le mécanisme que nous venons de décrire, et un mécanisme de règles de construction d'une classe. Dans l'exemple des assesseurs « get » et « set » en Java, il faut pouvoir instrospecter la classe, c'est à dire lister les méthodes, reconnaître le couple « get-set » en extraire le nom de la propriété, et être capable d'invoquer ces méthodes.

Il a été rajouté la possibilité pour une classe d'utiliser d'autres classes pour réaliser la fonction globale, par conséquent un composant n'est plus seulement la classe qui instancie la métaclasse, mais un ensemble de classes qui collaborent.

En Java on préfèrera implémenter une interface à hériter d'une classe abstraite (pour des raisons d'indépendances), mais le principe est le même. On peut donner une idée du mécanisme qui permet d'instancier une classe qui instancie une métaclasse. Pour pouvoir contenir toutes les classes qui font le composant, le fichier qui contient le composant est un fichier multi-archive (type zip, jar, etc). Cette archive contient toutes les classes qui font le composant, et un fichier dit « fichier manifeste ». Ce dernier contient la liste des fichiers .class (en Java chaque classe est contenue dans un fichier class) qui sont dans l'archive, ainsi que le nom de la classe qui implémente l'interface qui fait la spécification du composant. Ainsi il est possible de connaître la classe qu'il faut instancier. Un mécanisme Java permet de charger dans la machine virtuelle les classes que l'on vient de lire. Il suffit alors d'instancier la bonne classe, et de la forcer au type de l'interface (« caster » en anglais). On dispose alors d'une instance d'un objet qui implémente l'interface définie, et il suffit d'exploiter la fonctionnalité prévue par l'interface.

En résumé, un composant est presque un objet, mais il contient en plus un mécanisme d'instanciation générique. Quel est l'intérêt de ce mécanisme ?

Ce mécanisme, on l'a vu, permet de charger un composant qui n'existait pas au moment de la création du logiciel, et de l'exploiter grâce à l'interface qu'il implémente. Supposons que l'on puisse assimiler une interface à une fonctionnalité. Alors le mécanisme d'instanciation permet d'exploiter la fonctionnalité de morceaux de programme qui n'existent pas encore. Si l'on choisit une fonctionnalité stratégique dans un domaine, il apparaît alors qu'on va pouvoir disposer d'un coté de services qui exploitent la fonctionnalité, et de l'autre de composants qui remplissent la fonctionnalité. Il faut donc choisir une fonctionnalité qui permette de disposer d'une quantité importante de composants, de sorte que les services qui se contraignent à exploiter le composant puissent être rendus très utiles par la multiplication du nombre de composant pour lesquels ils peuvent être utilisés. De même, il faut choisir la fonctionnalité que remplit le composant de sorte que les acteurs qui implémentent le composant bénéficient instantanément de services importants.

I.C.5- DESIGN TIME VS RUN TIME

Il est usuel de définir deux phases pour le fonctionnement d'un composant ; « Design time », et « Run time ». « Design time » désigne la phase où l'on va manipuler le composant

pour l'intégrer dans le logiciel que l'on va concevoir, la phase « Run time » désigne la phase où le composant va s'exécuter, va être exploité.

Cette définition ne s'applique bien que dans le cas où un composant est utilisé pour construire un logiciel. On peut très bien les utiliser en dehors de toute création de logiciel. Les composants que nous avons définis pour réaliser le logiciel d'aide au dimensionnement ne fonctionnent pas suivant ces deux phases.

La grille d'analyse qui s'appuie sur la distinction entre les deux phases, n'a pas de valeur pour les composants que nous avons définis.

Cela révèle que la technologie composant est surtout pensée pour l'instant dans le cadre du génie logiciel, et qu'il nous faudra fournir un effort d'adaptation pour exploiter cette technologie dans le cadre du logiciel scientifique.

Le génie logiciel en offrant au public les technologies composants, nous permet d'offrir aux concepteurs de dispositifs techniques les avantages que l'on peut tirer de cette technologie. Le plus grand avantage est la possibilité de créer des frameworks génériques qui vont permettre aux composants métiers d'exister.

I.D Conclusion de l'état de l'art

Nous avons vu la place importante et névralgique que prenait le dimensionnement dans la conception, nous avons également vu les différents outils qui existent dans le dimensionnement, et nous avons vu les technologies logicielles qui permettent de réaliser un logiciel.

On voit que le dimensionnement est l'espace à conquérir pour continuer d'instrumenter la conception en entreprise. Aujourd'hui la chaîne qui va de l'idée au produit est instrumentée En commençant par le produit, en remontant petit à petit vers l'idée. Le dimensionnement est la nouvelle frontière. Il semble que des outils pour le dimensionnement, il n'y en ait aucun qui se détache vraiment de la concurrence et qui puisse se prétendre comme le logiciel de référence pour le dimensionnement. Enfin la démocratisation des technologies informatiques autour des composants, permet d'espérer pouvoir bientôt fournir des environnements intégrateurs, dans lesquels les programmes métiers (c'est à dire les programmes développés par des gens du métier) pourront s'intégrer et bénéficier des efforts mutualisés dans le framework.

CHAPITRE DEUXIEME - METHODOLOGIE

II- Méthodologie

Nous avons vu comment le dimensionnement s'articulait forcément entre modèle et cahier des charges, le modèle servant à exprimer les grandeurs dimensionnantes en fonction des dimensions, pour le relier au cahier des charges. Cette articulation ne résout pas le problème de l'activité humaine qui entoure le dimensionnement, ni leur importance relative.

II.A La méthodologie

II.A.1- ALGORITHMES D'OPTIMISATION

Nous avons vu que beaucoup de logiciels de dimensionnement utilisent l'optimisation sous contrainte pour conduire le dimensionnement. Le cahier des charges de dimensionnement étant adapté au cahier des charges d'optimisation. Malheureusement, l'optimisation sous contraintes d'un modèle est un problème compliqué, et les mathématiciens peinent à fournir un algorithme qui trouve à coup sûr la solution optimale, si elle existe, en un temps raisonnable. Néanmoins il existe des algorithmes d'optimisation sous contraintes, et il est quand même possible d'obtenir des résultats intéressants pour l'ingénieur.

Les problèmes de l'ingénieur en dimensionnement ne recouvrent pas tous les problèmes d'optimisation envisageables en mathématiques. En effet, il y a de nombreuses contraintes dans un cahier des charges de dimensionnement. Il faut prendre en compte les contraintes géométriques, qui dépendent de la complexité de la géométrie, les contraintes physiques, les contraintes liées aux normes, les contraintes liées au fonctionnement, qui expriment qu'un dispositif se comporte conformément à ce qu'on attend de lui.

Lorsqu'il faudra choisir les algorithmes d'optimisation à utiliser, il faudra tenir compte de ce critère de choix et sélectionner en priorité les algorithmes qui supportent un grand nombre de contraintes. Néanmoins, si l'on observe PASCOSMA, un logiciel qui utilise un algorithme d'optimisation pour le dimensionnement, on remarque que cet algorithme est de type SQP (pour Sequential Quadratic Programming en Anglais), choix justifié dans [Wurtz-96]. Il semble donc que l'on puisse associer à la caractéristique précédemment donnée, une caractéristique importante des algorithmes. Les algorithmes qui supportent un grand nombre de contraintes exploitent pleinement le calcul des dérivées du modèle pour choisir leur stratégie de résolution de contrainte. Il existe deux façons de calculer les dérivées : on peut soit effectuer un calcul formel des dérivées, et l'évaluer numériquement (c'est le cas de PASCOSMA), soit évaluer numériquement les dérivées par différence finie. Le calcul par différence finie introduit un paramètre numérique (le pas de la différence finie) qui s'avère être imprévisible, et en tous cas la première source de problème lorsque l'on fait une optimisation [Singh-92]. Dans ce cas, utiliser des différences finies pour dériver un modèle, est la première source d'échec de l'optimisation.

Le premier problème du concepteur qui réalise des optimisations est lié aux défaillances des algorithmes de dérivation numérique, en conséquence, il nous a semblé important de générer conjointement aux équations, les dérivées formelles de sorte que l'on puisse éliminer la première cause d'échec de l'optimisation, en se passant des dérivations numériques approchées.

II.A.2- MODELISATION

Nous avons vu que les modèles en dimensionnement servaient à exprimer les compromis de dimensionnement. Il en résulte qu'un bon modèle est un modèle qui exprime tous les compromis de dimensionnement et seulement ces compromis. Nous avons également vu que pour exprimer les compromis de dimensionnement il était parfois nécessaire de passer par l'expression des performances du dispositif, sur l'espace des solutions. Par conséquent un bon modèle est un modèle qui a une bonne précision uniforme.

Les modèles ne sont pas faciles à mettre au point, et nécessitent une bonne connaissance technique, une bonne connaissance de la modélisation, de la créativité. Néanmoins, on peut se demander comment mettre au point les modèles de dimensionnement.

Pour obtenir un modèle qui a une bonne précision uniforme, on dispose de deux outils. Dans un premier temps : exprimer le modèle avec des équations analytiques assure d'avoir un modèle relativement « lisse », et dont les dérivées le sont aussi. Il ne reste plus qu'à s'assurer que le comportement simulé suit le comportement réel sur tout l'espace des solutions. Encore une fois, si l'on a le souci de créer un modèle analytique dont les dérivées sont justes, il vient naturellement que les variations du modèle suivent les variations du dispositif, et donc les tendances du comportement réel et simulé vont dans le même sens. Dans un deuxième temps, il faut estimer la précision ponctuelle du modèle pour se faire une idée de la précision uniforme du modèle. Pour cela le mieux est de s'appuyer sur des outils de simulation fine, et d'analyse, qui permettent de simuler un dispositif avec une précision usuellement très supérieure à celle attendue pour les modèles pour le dimensionnement. Ainsi on pourra s'appuyer sur un jeu de simulations ponctuelles réparti dans l'espace des solutions pour valider la précision ponctuelle du dispositif.

Pour obtenir un modèle qui exprime tous les compromis, nous ne disposons pas d'outils dédiés. Nous proposons d'utiliser notre outil pour mettre au point par essai-erreur le modèle sans oublier de compromis. En effet, que se passe-t-il si nous optimisons un modèle qui n'est pas complet, qui oublie un compromis ? Le plus souvent, l'optimisation conduit à une solution instantanément absurde comme une grandeur menée à l'infini ou de manière équivalente à zéro. Il faut alors s'interroger sur les raisons de cet échec, et de la réponse à cette interrogation on peut extraire les préconisations pour l'amélioration de la modélisation.

Par exemple, la tenue à la tension de foudre d'un déclencheur est susceptible de contraindre le bobinage, mais dans le cas pratique qui nous a concerné, on se rend compte qu'il suffit de faire deux galettes avec le bobinage pour que la tension de claquage du bobinage soit loin d'être atteinte. Il est donc inutile de tenir compte de la tenue à la tension de foudre pour ce déclencheur basse tension.

Les deux étapes fondamentales du dimensionnement que sont modélisation et optimisation, ne sont pas des activités indépendantes, et il n'est pas souhaitable d'envisager la modélisation comme l'activité principale du dimensionnement, et l'optimisation comme le processus mécaniste qui conduit à la solution grâce à la bonne modélisation. Les deux activités interagissent, et le passage de l'une à l'autre et fréquent dans un processus de dimensionnement. La méthodologie de conception logicielle s'appuie sur une séparation logicielle des activités à l'aide de composants informatiques. Nous verrons donc quel composant informatique nous proposons pour faire la navette entre les deux activités de modélisation et d'optimisation sous contraintes. Nous verrons également les trois autres composants que nous avons proposés pour servir d'interface entre des sous activités du dimensionnement.

II.B Les COB

II.B.1- PRESENTATION

Le COB, comme on le verra dans sa définition est un composant logiciel qui sert d'interface entre les modèles et les services pour le dimensionnement. Cette interface permet de scinder des activités très imbriquées aujourd'hui et de gagner un facteur d'échelle considérable pour chacune des activités. Ainsi les personnes qui réalisent des modèles aujourd'hui ne peuvent les exploiter que dans l'environnement qui leur a permis de les mettre au point, et à des fins de simulation seulement. De même de l'autre coté du contrat, les algorithmes d'optimisation nécessitent de la programmation pour être utilisés. L'effet de seuil que rend possible un tel composant, est que les gens qui mettent leur modèle dans un COB pourront l'exploiter dans un optimiseur, mais aussi dans des environnements de simulation etc. La réciproque est bien sûr tout aussi vraie, tout algorithme d'optimisation par exemple qui pourrait optimiser un COB peut sans programmation s'appliquer sur tous les autres, et donc voir le nombre de ses applications considérablement augmenté.

Il faut donc trouver un contrat qui satisfasse tout à la fois les programmes de modélisation, et les programmes d'exploitation des modèles.

II.B.1.a- DEFINITION

Le COB est le composant de base dans l'environnement de dimensionnement que nous proposons, en effet il fait la navette entre ce qui nous a semblé être les deux activités majeures du dimensionnement : la modélisation pour celui-ci et l'optimisation. Le cas usuel que nous envisageons est donc le suivant: dans l'environnement de modélisation le concepteur met au point un modèle. Une fois au point, l'environnement émet un COB vers l'environnement d'optimisation. Dans l'environnement d'optimisation, le concepteur (qui peut être une autre personne) saisit le cahier des charges d'optimisation sous contraintes pour son COB, l'optimisation utilise alors la capacité de calcul fournie par le COB.

C'est pourquoi le COB est un composant qui définit un ensemble de paramètres. Les paramètres de sorties sont les grandeurs physiques ou économiques qui sont calculées en fonction des paramètres d'entrées. Il possible de commander le COB pour lire les valeurs et les noms de tous les paramètres. Evidemment il est possible de changer la valeurs des paramètres d'entrée. Il est également possible de commander le COB pour qu'il recalcule les paramètres de sortie en fonction des nouvelles valeurs des paramètres d'entrée.

Le COB permet également de calculer les dérivées des paramètres de sortie en fonction des paramètres d'entrée.

II.B.1.b- CALCUL DES DERIVEES

Le calcul des dérivées est une partie importante de la spécification des COBs. Il existe un certain nombre de techniques pour piloter le calcul des dérivées, nous verrons ensuite pourquoi en avoir privilégié une.

II.B.1.i- PRELIMINAIRES

Pour bien adapter un environnement informatisé de calcul, il est important de bien comprendre les différences entre quelques méthodes de calcul des dérivées.

Dans la suite on utilisera toujours la notation suivante pour décrire le système d'équations que l'on étudie:

$$\left(I_{n,m,l,(f)_{n+m},(\alpha)_l}\right)=\begin{cases}\left(I_{l,(\alpha)_l}\right)=\begin{cases}\alpha_1\\ \vdots\\ \alpha_l\end{cases}\\ \left(I_{n,(f)_n}\right)=\begin{cases}x_1=f_1(\alpha_1,\ldots,\alpha_l)\\ \vdots\\ x_i=f_i(x_i,\ldots,x_{i-1},\alpha_1,\ldots,\alpha_l)\\ \vdots\\ x_n=f_n(x_i,\ldots,x_{n-1},\alpha_1,\ldots,\alpha_l)\end{cases}\\ \left(I_{m,n,(f)_{m+n}}\right)=\begin{cases}x_{n+1}=f_{n+1}(x_i,\ldots,x_n,\alpha_1,\ldots,\alpha_l)\\ \vdots\\ x_{n+m}=f_{n+m}(x_i,\ldots,x_n,\alpha_1,\ldots,\alpha_l)\end{cases}\end{cases}$$

Équation II-1Représentation générale d'un modèle analytique

On remarque après examen détaillé que ce système comporte l paramètres d'entrée, n paramètres intermédiaires et m paramètres de sortie. En effet on appelle paramètre d'entrée les paramètres qui ne sont jamais évalués, paramètre de sortie les paramètres qui sont évalués, et jamais utilisés, et paramètre intermédiaire, les paramètres qui sont utilisés et évalués.

Ce système est strictement équivalent à un autre :

$$\left(I'_{m,l,n,(g)_m,(\alpha)_l}\right)=\begin{cases}x_{n+1}=g_1(\alpha_1,\ldots,\alpha_l)\\ \vdots\\ x_{n+m}=g_m(\alpha_1,\ldots,\alpha_l)\end{cases}$$

Équation II-2 Système réduit

où les $(g)_l$ se déduisent des $(f)_{n+m}$.

Nous nous restreignons au cas où nous désirons être capable d'évaluer le Jacobien du système, c'est à dire :

$$J_{i,j}=\left(\frac{\partial x_{n+i}}{\partial \alpha_j}\right)_{\substack{1\le i\le m\\ 1\le j\le l}}=\frac{\partial}{\partial \alpha_j}g(\alpha_1,\ldots,\alpha_l)$$

Équation II-3 : Définition du Jacobien

Pour arriver à ce but il existe plusieurs méthodes mathématiquement équivalentes mais néanmoins distinctes.

Lorsque l'on manipule des expressions formelles on peut appliquer soit une opération formelle, soit une opération numérique. On appelle opération formelle toute opération qui prend en entrée une expression formelle, et donne en sortie une expression formelle. On appelle opération numérique toute opération qui prend en entrée une expression formelle et qui donne en sortie un scalaire. Notre objectif est d'obtenir à partir des expressions formelles des équations la valeur scalaire du Jacobien. Nous avons aussi vu que pour des raisons d'efficacité, nous souhaitions utiliser la génération de code en langage de programmation et la compilation en langage machine pour effectuer rapidement l'opération numérique. Il en resulte que l'étape de génération de code est la frontière nécessaire entre les opérations

formelles et les opérations numériques. Il va de soi que nous ne nous risquerons pas à effectuer des opérations formelles sur les codes compilés, par conséquent cette étape est irréversible. Nous avons deux espaces disjoints, celui des opérations formelles, et celui des opérations numériques. Le passage de l'un à l'autre se fait par une étape de génération de code et de compilation.

Nous verrons alors que le processus qui conduit à l'évaluation numérique de la dérivée d'un système d'équations peut se décomposer en trois étapes, et que les positions relatives de ces trois étapes déterminent fortement les propriétés de ce processus et nous a conduit à en préférer un.

II.B.1.ii- LES DIFFERENTES APPROCHES

Il existe plusieurs approches que nous allons décrire de façon non formelle dans cette partie pour qu'elle reste compréhensible. Nous allons donc traiter un petit exemple caractéristique pour illustrer le propos. La généralisation de ce problème à la définition du système que nous avons proposé ne pose pas de problème majeur autre que celui dû à la lourdeur de l'écriture dans le cas général. Soit le système suivant :

$$(S)=\begin{cases} u = a^2 + 3\cdot b\cdot c \\ s = b^2 + a\cdot u \end{cases}$$

Il comporte deux entrées (a,b,c) et une sortie (s), et un paramètre intermédiaire (u).

- APPROCHE PAR SUBSTITUTION

L'approche par substitution est la plus simple conceptuellement. Elle consiste à calculer formellement les $(g)_l$, telles que définies dans le cas général, c'est à dire les expressions qui donnent directement les paramètres de sortie en fonction des paramètres d'entrée sans utiliser les paramètres intermédiaires. Il faut ensuite dériver formellement ces expressions, et remplir le Jacobien numériquement.

- EXEMPLE

➥ On évalue **formellement** la sortie en fonction des entrées c'est à dire la composition:

$$s = b^2 + a^3 + 3\cdot a\cdot b\cdot c$$

Équation II-4 : Système formellement composé.

➥ on évalue **formellement** le gradient de la fonction ainsi réalisée :

$$\overrightarrow{grad}(s)=\begin{pmatrix} 3\cdot a^2 + 3\cdot b\cdot c \\ 2\cdot b + 3\cdot a\cdot c \\ 3\cdot a\cdot b \end{pmatrix}$$

Équation II-5 : Gradient formellement évalué sur système formellement composé.

➥ On **génère** automatiquement du source capable d'effectuer le calcul, et on évalue **numériquement** le Jacobien.

```
Sub double[][] jacobian(a,b,c)
J[0,0]:=3*a*a+3*b*c;
J[0,1]:=2*b+3*a*c;
J[0,2]:=3*a*b;
```

```
End sub
```

• AVANTAGES

Cette méthode présente l'avantage d'être très facile à coder si l'on dispose d'outils de calcul formel existants, qui possèdent tous la capacité formelle de transformer une expression formelle en code de programmation, C ou FORTRAN.

• INCONVENIENTS

Cette méthode est très consommatrice de mémoire, puisqu'il faut manipuler des expressions sans cesse plus grosses, au point que parfois la génération sature la mémoire des ordinateurs courants.

Il semble qu'à l'exécution un certain nombre de calculs soit répétés plus que nécessaire.

Il est certain que cette méthode impose de recalculer formellement toutes les dérivations si l'on change une équation du modèle, ce qui signifie que l'on doit tout régénérer si l'on fait la moindre modification.

• APPROCHE PAR COMPOSITION

L'approche par composition est un peu plus complexe à mettre en œuvre, et à décrire proprement. On peut néanmoins dire avant de préciser par des formules, qu'elle consiste à calculer analytiquement les gradients de chacune des $(f)_n$, c'est à dire les expressions d'évaluation de chaque variable, intermédiaire ou de sortie, et de calculer numériquement la valeur. On calcule ensuite analytiquement le Jacobien en fonction des valeurs des gradients des $(f)_n$. Il ne reste plus qu'à l'évaluer numériquement.

• EXEMPLE

↳ on évalue **formellement** les gradients :

$$\overrightarrow{grad}(u) = \begin{pmatrix} 2 \cdot a \\ 3 \cdot c \\ 3 \cdot b \end{pmatrix}$$

Équation II-6 Gradient formelle évalué, sur système non-composé

On évalue **formellement** les règles de composition des dérivées partielles :

$$\overrightarrow{grad}(s) = \begin{pmatrix} u + a \cdot \overrightarrow{grad}(u)_1 \\ 2 \cdot b + a \cdot \overrightarrow{grad}(u)_2 \\ a \cdot \overrightarrow{grad}(u)_3 \end{pmatrix}$$

Équation II-7 : Composition formellement composée

↳ On **génère** automatiquement du source capable d'effectuer le calcul, et on évalue **numériquement** le Jacobien.

```
Sub double[][] jacobian(a,b,c)
u=a*a+3*b*c;
u1:=2*a;
```

```
u2:=3*c;
u3:=3*b;
J[0,0]:=u+a*u1;
J[0,1]:=2*b+a*u2;
J[0,2]:=a*u3;
End sub
```

- AVANTAGES

Cette méthode permet de prendre bien moins de place en mémoire car le travail peut être découpé en morceaux. Elle permet de factoriser des calculs.

- INCONVENIENTS

Cette méthode oblige à recalculer toutes les dérivées formellement si l'on modifie une équation, ce qui signifie que l'on doit tout régénérer si l'on fait la moindre modification dans le système.

- APPROCHE PAR DIFFERENTIELLE

C'est la méthode qui remonte au plus près du fondement de la composition des dérivées partielles. Elle consiste à associer à chaque équation, l'équation de la différentielle totale. Chaque différentielle est calculée dans le même ordre que la valeur. On obtient ainsi facilement les valeurs des différentielles de sortie en fonction des différentielles des entrées. Pour obtenir les dérivées globales pour le système, des variables de sortie en fonction d'une variable d'entrée (a) il suffit de mettre à zéro toutes les autres différentielles des paramètres d'entrées, et de mettre la différentielle de a à 1. Dans ce cas les valeurs des différentielles de sorties sont exactement égales à la dérivée partielle par rapport à a.

En terme plus cohérent avec les autres approches, celle-ci se définit par un calcul formel des dérivées de chacune des équations, et par un calcul numérique de la composition des dérivées.

- EXEMPLE

↳ on évalue **formellement** les différentielles : $df(x_i) = \overrightarrow{grad}(f(x_i)) \cdot (dx_i)$

$$(dS) = \begin{cases} du = 2 \cdot a \cdot da + 3 \cdot c \cdot db + 3 \cdot b \cdot dc \\ ds = u \cdot da + 2 \cdot b \cdot db + a \cdot du \end{cases}$$

Équation II-8 Différentielles formellement évaluées

↳ on **génère** automatiquement des sources capables d'effectuer le calcul, et d'évaluer **numériquement** la composition:

```
Sub double diff(a,b,c,da,db,dc)
u=a*a+3*b*c;
du=2*a*da+3*c*db+3*b*dc;
ds=u*da+2*b*db+a*du
return ds
```

- On utilise cette routine générique qui évalue **numériquement** la composition, pour remplir **numériquement** le Jacobien.

```
Sub double[][] jacobian(a,b,c)
J[0,0]:=diff(a,b,c,1,0,0);
J[0,1]:=diff(a,b,c,0,1,0);
J[0,2]:=diff(a,b,c,0,0,1);
Return J
End sub
```

• AVANTAGES

Cette méthode présente l'avantage de permettre une totale souplesse dans le système : la modification d'une équation du système n'engendre pas la régénération de tout le système, mais de seulement l'équation modifiée.

• INCONVENIENTS

La composition étant numérique, certaines expressions sont évaluées formellement, puis numériquement alors qu'il n'est pas nécessaire de le faire. En effet, on pourrait prédire qu'il n'est pas nécessaire de calculer la différentielle par rapport à une constante (comme la variable PI souvent utilisé dans les modèles). Mais par principe, si l'on veut garantir que l'on pourra rajouter des équations *a posteriori* sans régénérer l'existant, il faut pouvoir créer une dépendance nouvelle. Cela peut sembler absurde pour le cas de PI, pourtant il est possible d'envisager une situation où le cas peut se présenter.

Il semble que l'on risque de ne pas procéder à certaines simplifications fort utiles. En effet certaines dépendances entre des équations d'un système peuvent parfois se simplifier, mais en dérivant indépendamment les équations on élimine toute chance de simplifier les expressions. Il faudrait alors, si l'on souhaitait obtenir de tels résultats, procéder à une phase de simplification du modèle avant de procéder à sa dérivation.

• APPROCHE INVERSE

C'est la méthode duale de l'approche par différentielle. C'est à dire qu'elle est fondée sur le même principe, mais cette fois : on calcule de façon récursive la contribution de chaque équation aux dérivées partielles. Dans le cas de la méthode différentielle, on calcule la dérivée partielle de toutes les variables du système par rapport à une entrée, dans la méthode inverse on calcule la dérivée partielle d'une sortie par rapport à toutes les variables du système.

En termes plus cohérents avec les autres approches, cette méthode ne se distingue de l'approche par différentielle que par la façon de fournir le résultat de la dérivation formelle, et la façon d'évaluer numériquement la composition.

• EXEMPLE

- on évalue **formellement** les dérivées partielles.:

$$(pS)=\begin{cases} pa = pa + \dfrac{\partial s}{\partial a}\cdot ps = u\cdot ps \\ pb = 2\cdot b\cdot ps \\ pu = a\cdot ps \\ pa = pa + 2\cdot a\cdot pu \\ pb = pb + 3\cdot c\cdot pu \\ pc = pc + 3\cdot b\cdot pu \end{cases}$$

Équation II-9 : différentielles duales évaluée formellement

- on **génère** automatiquement des sources capables d'effectuer le calcul, et d'évaluer **numériquement** la composition:

```
Sub double[] diff(a,b,c,ps)
pa=u*ps;
pb=2*b*ps;
pu=a*ps;
pa=pa+2*a*pu;
pb=pb+3*c*pu;
pc=3*b*pu;
return {pa,pb,pc}
End sub
```

- On utilise cette routine générique qui évalue **numériquement** la composition, pour remplir **numériquement** le Jacobien.

```
Sub double[][] jacobian(a,b,c)
J[0]:=diff(a,b,c,1);
Return J
End sub
```

• AVANTAGES

En tant que méthode duale de la méthode par différentielle, cette méthode présente les mêmes avantages que celle par différentielle.

La différence entre la méthode différentielle duale et la méthode différentielle directe réside dans le nombre d'appels qu'il faut pour remplir le Jacobien. La méthode différentielle directe remplit le Jacobien ligne par ligne, il faut donc l'invoquer pour chacune des entrées. La méthode duale remplit le Jacobien colonne par colonne, et donc il faut l'invoquer pour chacune des sorties.

Dans notre petit exemple il n'y a qu'une sortie, le calcul est donc plus simple. Dans le cas général il n'y pas de raison d'estimer que le nombre d'entrées sera très différent du nombre de sorties, il n'y a donc de ce point de vue, aucune raison de privilégier une méthode par rapport à une autre.

• INCONVENIENTS

Cette méthode présente les mêmes inconvénients que l'approche différentielle.

On peut néanmoins ajouter que le calcul se fait « en commençant par la fin », c'est à dire que l'on va calculer les valeurs des paramètres, puis seulement ensuite les valeurs des p. Ceci a une fâcheuse conséquence sur la capacité de cette méthode à calculer des modèles non plus mathématiques mais calculatoires.

En effet jusqu'à présent, on a fait l'hypothèse implicite que les modèles avaient une propriété importante : chaque paramètre a une valeur dans tout le modèle et une seule. Par conséquent il est possible de calculer toutes les valeurs de tous les paramètres, puis les valeurs des dérivées (qui dépendent des valeurs des paramètres aussitôt que des équations sont non-linéaires). En revanche dans un modèle non plus mathématique mais calculatoire une variable peu se voir assigner plusieurs valeurs à différents instants du calcul du modèle. On ne peut dire qu'il est possible de calculer toutes les valeurs de tous les paramètres puis les valeurs des dérivées, car on ne sait pas quelle valeur du paramètre utiliser.

Prenons un exemple simple, en définissant un modèle qui permettent de calculer la puissance nième de a.

$$y=a^n$$

Équation II-10 : exemple de fonction

La dérivée par rapport à a vaut :

$$\frac{\partial y}{\partial a}=n*a^{n-1}$$

Équation II-11 : dérivée formelle

Mais si l'on n'utilise pas la forme mathématique, mais un algorithme qui permet de calculer cette fonction, on a :

```
P=1 ;
Pour i=0 tant que i<n
   i=i+1
   P=P*a ;
Retourne P
```

Comment dans ce cas dériver formellement ce système sans reconnaître formellement qu'il s'agit de la fonction puissance ? Le problème est que la variable P ne prend pas qu'une valeur, mais n !

Il suffit de différencier le code de calcul, en calculant conjointement les valeurs et les dérivées.

```
P=1 ;
dP=0 ;
da=1 ;
Pour i=0 tant que i<n
   i=i+1
   dP=P*da+a*dP
   P=P*a ;
Retourne P
```

Où dP est la différentielle de P. Il est à noter qu'il faut impérativement calculer la valeur de la différentielle avant la valeur de la grandeur elle-même.

Dans ce cas on peut vérifier que dP est exactement égal à la formule calculée par un autre moyen.

Dans cet exemple on voit tout l'intérêt de la différentielle qui permet de mener le calcul de la dérivée dans le même ordre et en même temps que le calcul des valeurs.

II.B.1.iii- PRESENTATION SYNTHETIQUE

Nous allons ici présenter une vision unificatrice de ces différentes méthodes afin de se convaincre que le passage de l'une à l'autre n'est pas un changement de nature, et qu'elles sont bel et bien liées.

α – L'APPROCHE THEORIQUE

Il est bon de rappeler que la seule façon rigoureuse de démontrer comment sont composées les dérivées d'ordre 1, est d'introduire la notion de différentielle totale (ou exacte) d'une équation. Dans ce cas la différentiation totale s'exprime par :

$$diff : \begin{cases} \Im \to \Im \\ \left(f : \begin{cases} \Re^n \to \Re \\ (x_i) \mapsto f(x_i) \end{cases} \right) \mapsto df : \begin{cases} \Re^n \to \Re \\ (dx_i) \mapsto \vec{grad}(f(x_i)) \bullet (dx_i) \end{cases} \end{cases}$$

Équation II-12 : Définition de l'opérateur différentielle

• DERIVATION

Si dans le système $\left(I_{n,m,l,(f)_{n+m},(\alpha)_l}\right)$ on applique l'opérateur diff à toutes les équations on obtient un autre système qui se ramène à :

$$\left(dI'_{m,l,n,(g)_m,(\alpha)_l}\right) = \begin{cases} dx_{n+1} = dg_1(d\alpha_1,\ldots,d\alpha_l) \\ \vdots \\ dx_{n+m} = dg_m(d\alpha_1,\ldots,d\alpha_l) \end{cases}.$$

Équation II-13 : définition du système dérivé

• COMPOSITION

Pour construire le Jacobien il ne reste plus qu'à évaluer numériquement ce système pour L vecteurs :

$$\begin{pmatrix} d\alpha_1 \\ \vdots \\ d\alpha_l \end{pmatrix} = \left\{ \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ \vdots \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \ldots, \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 1 \end{pmatrix} \right\}$$

Équation II-14 : Ensemble de vecteurs conduisant au Jacobien.

• EVALUATION

il ne reste plus qu'à remplir numériquement le Jacobien

$$J_{n,l} = \left(dI'_{m,l,n,(g)_m,(\alpha)_l}\right)\begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & \ddots & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \dots & 0 & 1 \end{pmatrix}$$

Équation II-15 : Défintion du Jacobien.

On voit bien que l'évaluation de la matrice identité ci dessus peut se faire de deux façons différentes: soit par colonne et c'est la composition dite directe, soit par ligne et c'est la composition dite inverse. C'est pourquoi les deux approches utilisant la composition des dérivées, ont un dual.

- APPROCHE DIFFERENTIELLE

l'approche différentielle se décompose en :

- une évaluation **analytique** de la dérivation
- une évaluation **numérique** de la composition (directe ou inverse)
- une évaluation **numérique** du Jacobien.

Comme on a vu en introduction, les évaluations analytiques ont lieu dans la phase de génération, tandis que les évaluations numériques ont lieu dans la phase d'utilisation. Par conséquent l'évaluation numérique des règles de la composition permet de composer les équations lors de l'utilisation.

- APPROCHE PAR COMPOSITION

L'approche par composition se décompose en :

- une évaluation **analytique** de la dérivation
- une évaluation **analytique** de la composition (directe ou inverse)
- une évaluation **numérique** du Jacobien.

Il est donc nécessaire d'être dans une phase de génération afin de pouvoir recomposer les équations.

- APPROCHE PAR SUBSTITUTION

L'approche par substitution se décompose en :

- une évaluation **analytique** de la composition
- une évaluation **analytique** de la dérivation
- une évaluation **numérique** du Jacobien.

Il est donc également nécessaire d'être dans une phase de génération afin de pouvoir recomposer les équations.

———

Si l'on note que le passage de la phase d'opérations analytiques à la phase d'opération numérique nécessite la génération de code, on peut alors poser un crible qui permet de choisir le mode de dérivation adapté au dimensionnement.

Il est nécessaire de calculer analytiquement les dérivées car, comme on a vu la première cause d'échec de l'optimisation est l'algorithme de dérivation numérique.

Il est nécessaire d'évaluer numériquement le modèle et le Jacobien.

Il est souhaitable de ne pas devoir générer à nouveau le modèle si l'on souhaite le composer avec un autre modèle (il se peut même que l'on souhaite composer des modèles que l'on ne peut pas soi-même générer).

Il en ressort que la seule solution consiste à calculer analytiquement les dérivées, puis numériquement les compositions, et les valeurs. C'est la solution différentielle.

II.B.1.iv- CONCLUSION

Le calcul des dérivées est un facteur déterminant entre deux environnements d'optimisation si l'on s'appuie sur le fait que la première cause d'échec de l'optimisation est le calcul des dérivées numériques.

On se rend compte avec une analyse des différentes techniques de calcul des dérivées que toutes ne sont pas équivalentes, et que surtout elles ont des conséquences importantes sur la génération de code.

Le choix de la technique par différentielle est un choix qui va permettre au delà du simple calcul des dérivées d'un modèle, de calculer les dérivées de COB composés entre eux, et aussi de morceaux d'algorithmes.

II.B.2- CONCLUSION SUR LES COB

Les COBs sont les composants qui réalisent un compromis entre différents univers, d'un coté l'univers de la modélisation des phénomènes, de l'autre l'univers de l'exploitation des modèles (tracés de courbe, optimisations etc.). En tant que compromis il ne satisfait forcément pas tous les acteurs de tous les univers, mais en tant que compromis il permet de gagner un facteur d'échelle et de proposer une unification des interactions entre ces mondes. En effet il est dommage pour un modélisateur électrotechnicien de devoir travailler à rendre son modèle compatible avec la multitude d'environnements dans lequel il pourrait être utile et réciproquement il est dommage en tant de que développeur d'un algorithme d'optimisation de devoir travailler à importer différents modèles à son propre format.

II.C Les composants corollaires

II.C.1- LES GEN-X

Les COBs sont le plus souvent générés à partir de modèles analytiques. Se cantonner à des modèles purement analytiques est très restrictif. Il faut pouvoir étendre les capacités de génération de code sans changer les capacités existantes. On souhaite prendre en compte les équations différentielles, les équations implicites, des langages métiers (comme le langage de description de circuit appelé « netlist »). Pour cela nous avons défini un composant qui permet à des développeurs extérieurs de rajouter des capacités de génération particulière.

II.C.1.a- PRINCIPE

Le principe sur lequel s'appuie le composant gen-X est la création d'un COB à partir d'un ensemble d'équations et de fonctions. Le composant gen-X définit un langage dans lequel est écrit le modèle. Par exemple des équations analytiques pour le composant gen-X de base. Puis à partir de la lecture de ce modèle il déduit un ensemble d'équations et de fonctions. Les équations définissent des paramètres en fonction d'autres paramètres qui peuvent être utilisés dans d'autres équations fournies par d'autres langages. De même, les fonctions fournies par un langage peuvent être utilisées par d'autres équations, ou d'autres fonctions générées par un autre composant gen-X.

Ainsi donc le résultat d'une équation différentielle est une fonction. Le langage de prise en compte des équations différentielles peut donc générer la fonction solution (même si elle utilise une technique numérique pour en calculer la valeur).

Il reste que le composant gen-X, en créant des équations et des fonctions, permet d'écrire des fonctions et des fonctionnelles. Les fonctionnelles sont par définition des fonctions de fonctions. Ainsi donc une équation différentielle simple (Ordinary Differential Equation) est une fonctionnelle, tout comme une équation implicite. Quelles sont les autres fonctionnelles, et comment dériver des fonctionnelles, c'est ce que nous allons examiner maintenant.

II.C.1.b- DERIVATION DES FONCTIONNELLES

Les fonctionnelles sont les applications des scalaires vers les fonctions.

La différentiation totale est un calcul bien connu pour les fonctions exprimées à partir d'une équation analytique explicite. Nous allons voir comment différentier deux types de fonctionnelles non explicites :

II.C.1.i- LES FONCTIONS IMPLICITES

Une fonction g(a) peut être définie par g(a)=y / f(y,a)=0. Cette définition n'est pas très rigoureuse car l'équation f(y,a)=0, peut avoir plusieurs solutions (ou aucune) , néanmoins cette difficulté n'intervient pas dans la différentiation. Par récursivité, on connaît la différentielle totale de f : df(y,a,dy,da). On en déduit dg(a,da) par :

$$dg(a,da) = -\frac{df(y,a,0,da)}{df(y,a,1,0)}$$

Équation II-16différentielle d'un système implicite

en effet il est facile de démontrer que :

$$df(y,a,0,1) = \frac{\partial f}{\partial a}(y,a)$$

Équation II-17 : Obtention de la première dérivée partielle à partir de la différentielle totale

et

$$df(y,a,1,0) = \frac{\partial f}{\partial y}(y,a)$$

Équation II-18 : Obtention de la seconde dérivée partielle à partir de la différentielle totale

Partant du théorème des fonctions implicites

$$df(y,a)=\frac{\partial f(y,a)}{\partial y}dy+\frac{\partial f(y,a)}{\partial a}da$$

Équation II-19Théorème des fonctions implicites

Sachant que f(y,a) est contraint à toujours valoir zéro, il est évident que la différentielle de f, soit partout nulle ! l'expression ci-dessus se simplifie alors en :

$$\frac{dy}{da}=\frac{dg(a)}{da}=-\frac{\frac{\partial f(y,a)}{\partial a}}{\frac{\partial f(y,a)}{\partial y}}$$

Équation II-20 Dérivée

Il vient alors l'expression que l'on cherchait pour la différentielle d'une équation implicite.

L'intérêt ici n'est pas mathématique (la démonstration généralisée à N dimensions prendrait trop de place), mais cela prouve que pour le cas des fonctions définies implicitement par une fonction, la connaissance de la différentielle totale se transmet récursivement.

II.C.1.ii- LES EQUATIONS DIFFERENTIELLES

De même que pour les fonctions définies implicitement on peut définir une fonction g(t,a) par :

$$g(t,a,b)\left|\frac{\partial g}{\partial t}=f(g,a)\ \text{et}\ g(0,a)=b\right.$$

Équation II-21 : Définition d'une équation différentielle explicite du premier ordre.

En fait on voit bien que la valeur de g dépend aussi de la condition initiale. La dérivée partielle de g par rapport au temps est triviale (f(g(t,a,b)).

La dérivée par rapport aux paramètres de f est aussi évidente :

$$\frac{\partial}{\partial a}g(t,a,b)\left|\frac{\partial^2 g}{\partial a\partial t}=\frac{\partial}{\partial a}f(g,a)+\frac{\partial}{\partial g}f(g,a)*\frac{\partial g}{\partial a}\right.\quad\text{condition initiale}\quad\frac{\partial g(0,a,b)}{\partial a}=0$$

Équation II-22 : Dérivée de la solution de l'équation différentielle par rapport aux paramètres.

La dérivée par rapport à la condition initiale est plus complexe, mais la formule est la même :

$$\frac{\partial g}{\partial t}=f(g,a)$$
$$\frac{\partial^2 g}{\partial t\partial b}=\frac{\partial f(g,a)}{\partial y}*\frac{\partial g}{\partial b}\quad\textbf{avec comme condition initiale}\quad\frac{\partial g(0,a,b)}{\partial b}=1$$

Équation II-23 : dérivée de la solution de l'équation différentielle par rapport aux conditions initiales

On voit bien que l'on est capable de calculer le gradient de la fonction solution de l'équation différentielle, par une résolution d'équations différentielles.

Ces équations différentielles peuvent se ramener à une seule, qui concerne la différentielle de g.

$\frac{\partial}{\partial t} dg(t,a,b,dt,da,db) = df\left(g(t,a,b),a,dg(t,a,b,dt,da,db),da\right)$. **comme condition initiale :** $dg(0,a,b,dt,da,db) = db$

Équation II-24 Synthèse des différentes dérivées avec la différentielle.

Attention, nous sommes dans le cas de la dérivation où il faut connaître la valeur de g(t) pour pouvoir calculer dg(t).

II.C.1.iii- CONCLUSION

On a vu que la connaissance de la différentielle était héréditaire pour les opérateurs «équation différentielle » et « fonction définie implicitement ». De plus pour les fonctions définies par des équations analytiques explicites on sait calculer la différentielle. Par conséquent, tant que l'on définit des fonctions par un de ces trois opérateurs, on peut compter sur la connaissance de la différentielle. La connaissance de la différentielle garantit la capacité à propager la dérivée dans tout le calcul, pour connaître la variation des paramètres de sorties par rapport aux paramètres d'entrée.

Nous n'avons analysé ici que trois opérateurs, ce sont les seuls que nous avons rencontrés dans la modélisation des dispositifs pour l'électrotechnique, néanmoins on sait par ailleurs que l'on est capable de dériver n'importe quel fonctionnelle exprimée par des programmes. Sachant que le but de notre travail est la programmation automatique des modèles, une fonctionnelle qui ne serait pas calculable par un programme n'a pour nous aucun intérêt.

II.C.1.c- DERIVEE DES PROJECTEURS

Une fois postulé que l'on connaissait la différentielle de chaque fonction, il reste à ramener cette différentielle dans l'espace des scalaires. De même que pour les fonctionnelles, nous avons détecté trois types d'opérateurs.

II.C.1.i- L'OPERATEUR « VALEUR DE LA FONCTION »

$m = f(n,a)$ où f est une fonction du temps (f(t,a)) définit un projecteur. En effet m ne dépend pas du temps.

Il vient $dm = df(n,a,dn,da)$. On connaît donc la différentielle totale des paramètres statiques uniquement grâce à la connaissance de la différentielle de la fonction.

II.C.1.ii- L'OPERATEUR « RACINE DE LA FONCTION »

$m \,|\, f(m,n,b) = 0$ définit un projecteur.

$$dm = \frac{df(m,n,b,0,dn,db)}{df(m,n,b,1,0,0)}$$

Équation II-25 Différentielle du projecteur racine de la fonction

On connaît encore une fois la différentielle totale du paramètre m, par la seule connaissance de la différentielle totale de la fonction.

II.C.1.iii- L'OPERATEUR « ESPERANCE »

$$m = \int_{n}^{p} g(f(t,a))dt$$

Équation II-26 Définition du projecteur espérance

$$dm = g(f(p,a))dp - g(f(n,a))dn + \int_{n}^{p} dg(f(t,a), df(t,a,0,da))dt.$$

Équation II-27 Définition de la différentielle de l'opérateur espérance

si g est la fonction identité, alors, ce projecteur permet de définir la valeur moyenne, si g est la fonction x^2, alors ce projecteur est égale au carré valeur efficace.

II.C.1.iv- L'OPERATEUR « MAXIMUM »

On définit l'opérateur maximum, comme une application qui à une fonction associe les coordonnées du maximum. Par exemple :

$$ym = \max_{y}(f(y,a))$$

Équation II-28 Définition du projecteur maximum d'une fonction

en conséquence, la dérivée (si elle existe) consiste à trouver la direction du "chemin" qui conserve la propriété d'être max. Si l'on fait varier a, alors il faut faire varier y de façon à garder f maximum. Il en résulte la formule suivante.

$$dym = \max_{y}(f(y,a) + df(y,a,0,da))$$

Équation II-29 Définition de la différentielle du projeteur maximum d'une fonction

II.C.1.v- CONCLUSION

Avec ces trois opérateurs, il nous semble que l'on peut définir tous les projecteurs qui à notre connaissance nous sont utiles. On constate que dans tous les cas présentés, la dérivation ne fait jamais intervenir de calculs plus complexes que le projecteur lui-même.

II.C.2- LE COMPOSANT OPTIMISATION

Une des utilisations majeures des modèles analytiques réside dans l'optimisation sous contraintes. Il est difficile d'écrire un algorithme d'optimisation sous contraintes, et surtout de le diffuser, et le tester. A l'inverse, si l'on souhaite essayer d'optimiser son propre modèle sur différents algorithmes d'optimisation, on se rend compte que le travail à faire est assez important. La technologie composant apporte des solutions pour ce genre de problème.

II.C.2.a- LE PROBLEME D'OPTIMISATION SOUS CONTRAINTES

L'optimisation sous contraintes est souvent définie par la minimisation d'une ou plusieurs fonctions objectifs, en respectant des contraintes d'égalités et des contraintes d'inégalité. Ce qui se formalise mathématiquement par :

$$\begin{cases} \min\limits_{x_i}\left(f_j(x_i)\right) & \forall j < obj \\ g_j(x_i) = 0 & \forall j < constr \\ h_j(x_i) < 0 & \forall j < inequ \\ i < taille \end{cases}$$

Équation II-30 Définition du problème général d'optimisation

Où:

- -"obj" représente le nombre de fonctions objectif.
- -"constr" représente le nombre de contraintes d'égalité.
- -"inéqu" représente le nombre de contraintes d'inégalité.
- -"taille" représente le nombre de paramètres optimisable.

Une telle description du problème, bien que générale n'est pas suffisante. En effet la plupart des algorithmes d'optimisation introduisent des informations supplémentaires pour améliorer leur efficacité. A titre d'exemple les informations supplémentaires peuvent être de signifier les fonctions linéaires. Ainsi, si une des fonctions qui permet d'évaluer les contraintes d'égalités est linéaire par rapport aux paramètres d'entrée du modèle, alors il devient facile de gérer ce genre de paramètre (on retrouve les éléments de l'algorithme du simplex).

De plus, il est nécessaire de normaliser le problème d'optimisation. En effet si un paramètre est de l'ordre de 10^9 et qu'un autre paramètre est de l'ordre de 10^{-9} il devient délicat de les comparer numériquement. Le principe de la normalisation consiste à ramener tous les paramètres à un intervalle commun. Pour cela il faut connaître par l'ordre de grandeur des résultats et leur variation. C'est pourquoi il est indispensable de récupérer cette information de l'utilisateur, qui seul connaît son modèle.

Le composant optimisation se situe donc, après la normalisation, et en tout cas un composant optimisation doit considérer que le problème qui lui est donné contient des valeurs numériques correctes.

II.C.2.b- LE PROBLEME DE LA CONFIGURATION

L'efficacité d'un algorithme d'optimisation est souvent accrue par l'adjonction d'une information sur le modèle ramené au problème d'optimisation. Il s'agit par exemple de dire à l'algorithme qu'un paramètre est discret, linéaire, quadratique, ou bien variant lentement.

Un composant optimisation se doit de pouvoir faire fonctionner tous les algorithmes sur les mêmes modèles, sinon il n'y aurait pas d'intérêt à développer un atelier logiciel, et on se retrouverait dans la situation existante, où chaque algorithme dicte le problème qu'il résout, et le développeur est en charge d'adapter son modèle à l'algorithme. Cette solution n'est pas viable dans l'hypothèse d'un ingénieur qui développe les modèles.

Il est donc impératif que tous les algorithmes puissent répondre au même problème : l'optimisation sous contraintes. Ce type de problème n'interdit pas aux algorithmes d'adapter leurs performances à des classes de problèmes à l'intérieur même de l'optimisation sous contraintes. Par exemple certains algorithmes seront d'autant plus performants que le nombre de contraintes d'égalité est élevé, d'autre le seront pour des problèmes de grande taille, mais presque linéaires.

II.C.2.c- LA SOLUTION PROPOSEE

L'optimisation pour le dimensionnement consiste donc à transformer un problème de dimensionnement en un problème d'optimisation sous contraintes. Cela comprend certaines activités fastidieuses qui sont automatisables comme la normalisation du problème, la connexion avec un algorithme d'optimisation qui va résoudre le problème.

Ce genre de solution est réalisable de manière entièrement automatique dans un logiciel (PASCOSMA) mais une limitation apparaît rapidement pour la réalisation d'algorithmes d'optimisation. En effet comme on l'a déjà vu, les algorithmes d'optimisation ne se contentent pas de résoudre le problème général d'optimisation, mais tous tentent d'améliorer le processus d'optimisation en rajoutant de l'information au modèle. Il faut donc prévoir dans la technologie composant utilisée, de permettre la configuration du composant optimisation. Une part de l'activité de dimensionnement réside dans la configuration de l'algorithme d'optimisation.

II.C.2.i- CONFIGURATION NIVEAU GENERAL

La configuration niveau général consiste à permettre à un développeur de composant d'optimisation de proposer à l'utilisateur final une configuration générale de son algorithme. Cela permet de choisir des paramètres de l'algorithme comme une précision finale, mais aussi différentes stratégies.

De tous les algorithmes rencontrés dans la littérature ou bien développés au sein du laboratoire, aucun ne permettait à l'utilisateur de configurer les mêmes paramètres. Citons par exemple, la précision de fin [Harwell-87], la précision des contraintes d'égalité [Lawrence], ou bien la stratégie de recherche de ligne [Lawrence], l'allure de la fonction de répartition (algorithme BreitWeigner du LEG implémenté par Jean-Michel Guichon).

II.C.2.ii- CONFIGURATION NIVEAU MODELE

Les configurations au niveau général ne suffisent pas à fournir à l'algorithme toutes les informations nécessaires exploitables. Par exemple, CFSQP ne traite pas de la même manière les paramètres de sorties qui sont linéaires par rapport aux entrées.

Il nous a semblé nécessaire de permettre au développeur de composant d'optimisation de proposer à l'utilisateur final de configurer chacun des paramètres du modèle.

On peut citer par exemple, le traitement différent des paramètres de sorties linéaires (CFSQP), la configuration de l'amplitude du changement de chaque variable dans la recherche de ligne de type "watchdog (VF13).

II.C.2.iii- TECHNOLOGIE DE CONFIGURATION FACILE POUR LE DEVELOPPEUR

Nous avons essayé de fournir une configuration des algorithmes d'optimisation similaire à celle des JavaBean.

Un des premiers niveau de configuration consiste à définir des patrons de conception (Design pattern en anglais [Gamma-95]). C'est à dire des schémas de construction d'une classe Java. Il ne reste alors plus qu'à reconnaître le patron dans la classe Java et exploiter cette reconnaissance.

Pour reconnaître un patron dans une classe Java, il suffit d'introspecter la classe (fonctionnalité du langage Java) et de récupérer la liste des méthodes de la classe, la liste des champs de la classe. Un patron est donc une règle sur les noms des méthodes et des champs qui puisse être reconnue par un programme.

Le patron get-set. Ce patron est proche de celui des JavaBean©. Il consiste à créer deux méthodes pour un paramètre, une méthode get<nom parametre> qui renvoit la valeur du paramètre, et une méthode set<nom paramètre> qui prend en paramètre une valeur pour le paramètre. Lorsqu'il reconnaît ce patron, notre logiciel propose à l'utilisateur de modifier la valeur de ce paramètre : il affiche à l'utilisateur le nom du paramètre : <nom parametre> et la valeur du paramètre récupéré par la méthode get<nom parametre> et si l'utilisateur modifie la valeur, il répercute la modification en invoquant la méthode set<nom parametre>. Si le type de paramètre est un booléen, alors au lieu de fournir une zone de texte dans laquelle taper la valeur du paramètre, il propose une case à cocher.

Le patron indexOf : Ce patron n'existe pas dans JavaBean, et permet de choisir un index parmi une liste indexée (par exemple, un choix de stratégie). Il consiste en deux méthodes getIndexOf<nom parametre> et setIndexOf<nom parametre> (assesseur pour un int), et un tableau de String <nom paramètre>. Quand le logiciel reconnaît ce patron, il affiche à l'utilisateur une liste de choix avec les chaînes de caractères prises dans le tableau <nom parametre>. Lorsque l'utilisateur choisi un item dans la liste, le logiciel invoque la méthode setIndexOf<nom parametre> avec la valeur de l'index correspondant.

Cette méthode de programmation est éprouvée dans le cadre d'autre composant tel que les JavaBean©. Elle permet pour le développeur d'un algorithme d'optimisation de fournir facilement une configuration de son algorithme à l'utilisateur final. L'interface générique auto construite par introspection du composant est relativement conviviale, rapide de développement.

En revanche si les développeurs de composant optimisation veulent aller plus loin dans la configuration de leur algorithme, ils peuvent utiliser le second niveau de configuration.

II.C.2.iv- TECHNOLOGIE DE CONFIGURATION FACILE POUR L'UTILISATEUR FINAL

Un autre niveau possible de configuration des algorithmes est de permettre au développeur de l'algorithme d'écrire lui-même l'interface graphique de configuration de son algorithme. A l'inverse de la première technique, le travail à fournir par le développeur de composant optimisation est élevé, tandis que la configuration de l'algorithme peut être rendue conviviale.

La technique consiste non plus à passer une classe Java qui sera introspectée, mais plutôt une classe Java qui correspond à son objet graphique, et qui sera utilisée telle qu'elle. En fait lors de l'introspection de la classe Java qui est passée, le logiciel commence par regarder si la classe Java correspond à une classe graphique (JComponent). Si ce n'est pas le cas, le logiciel va alors utiliser les mécanisme d'introspection pour créer l'objet graphique, si c'est le cas, alors c'est l'objet graphique lui même qui est utilisé. L'utilisateur peut donc développer l'objet graphique comme il l'entend pour créer l'interface graphique de configuration de son composant (éditeur graphique de stratégie etc.).

En utilisant cette technique de configuration de son algorithme le développeur peut atteindre le niveau de convivialité qu'il souhaite pour son composant, en revanche le travail à fournir est assez élevé.

- CONCLUSION

Le cahier des charges est un point important du dimensionnement, et il est impératif de pouvoir tenir compte de contraintes d'égalité (par exemple pour construire un moteur à puissance donnée, il faut pouvoir contraindre la puissance calculée à la valeur que l'on se fixe). Dans ce cas les algorithmes d'optimisation utilisent une précision sur cette contrainte d'égalité pour savoir si elle est respectée. Si l'on prend une précision P, et que l'on pose un cahier des charges avec M contraintes d'égalité, l'espace des solutions qui satisfont le cahier des charges est très petit par rapport à l'espace des solutions complet. La probabilité de tomber par hasard sur un point qui satisfait le cahier des charges est donc de P^M. A titre indicatif, une précision de 10^{-3} et 4 contraintes d'égalité n'est pas un cas exceptionnel, et pourtant la probabilité est de 10^{-12}, soit 1 chance sur mille milliard !

Notre technologie de composant est ouverte, et rien n'empêche d'utiliser un algorithme intégralement stochastique, néanmoins ils nous semble que le cas d'utilisation le plus fréquent n'est pas adapté à un algorithme de ce type (un algorithme génétique est inenvisageable si seulement un individu sur 1000 Milliards et viable), c'est pourquoi nous avons commencé par implanter des algorithme de la famille SQP (Sequential Quadratic Programming) reconnus comme étant les plus performants sur des systèmes fortement non linéaires et fortement contraints.

CHAPITRE TROISIEME - OUTILS

« Attendre d'en savoir assez pour agir en toute lumière, c'est se condamner à l'inaction. »
Rostand (Jean)

III- Outils

Nous allons montrer dans ce chapitre comment se déclinent les composants spécifiés du point de vue de l'utilisateur. Pour expérimenter ces concepts nous avons construit un prototype qui s'appelle ProDESIGN. Son architecture suit rigoureusement les domaines de chacun des composants. Nous verrons donc module par module l'implication de la structure composant sur l'utilisateur.

Nous verrons ensuite quelques processus exécuter sur cette plate-forme, et comment les processus bénéficient de l'architecture composant.

III.A Architecture

ProDESIGN est structuré autour du composant COB, définissant l'environnement de travail premier. Il y a donc deux types de modules dans ProDESIGN : ceux capables de générer un COB, et ceux capables de lire un COB. Nous avons implémenté plusieurs de ces modules mais nous ne présenterons ici qu'un générateur, et qu'un utilisateur de COB.

A leur tour le générateur, et l'utilisateur de COB que nous avons implémentés sont structurés autour d'un composant.

III.A.1- LE GENERATEUR

Le générateur de COB est lui aussi structuré autour d'un composant de génération de COB le composant gen-X. Il permet de générer des COBs à partir d'un langage. Chaque composant gen-X définit un langage et la manière de générer le code Java dans le COB pour la description donnée dans le langage.

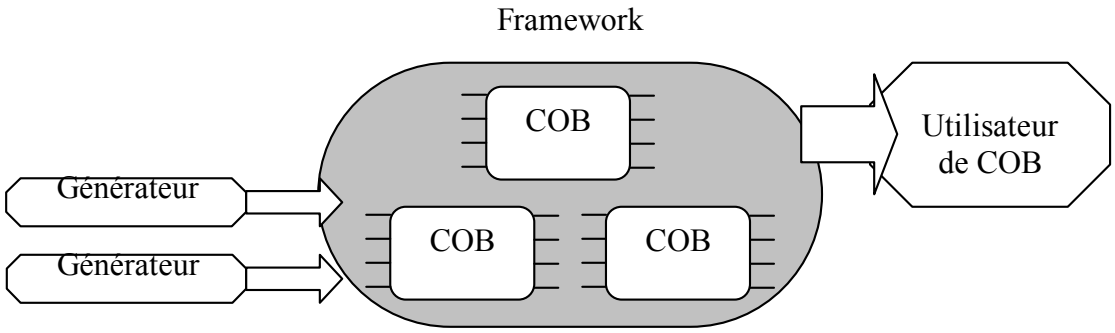


Figure 4 : Synoptique de la structure du logiciel.

III.A.1.a- LANGAGE PAR DEFAUT

Il y a dans notre logiciel un composant gen-X qui permet de gérer des équations analytiques explicites. Ce composant est le composant par défaut. Nous allons voir dans un exemple simple comment utiliser ce module.

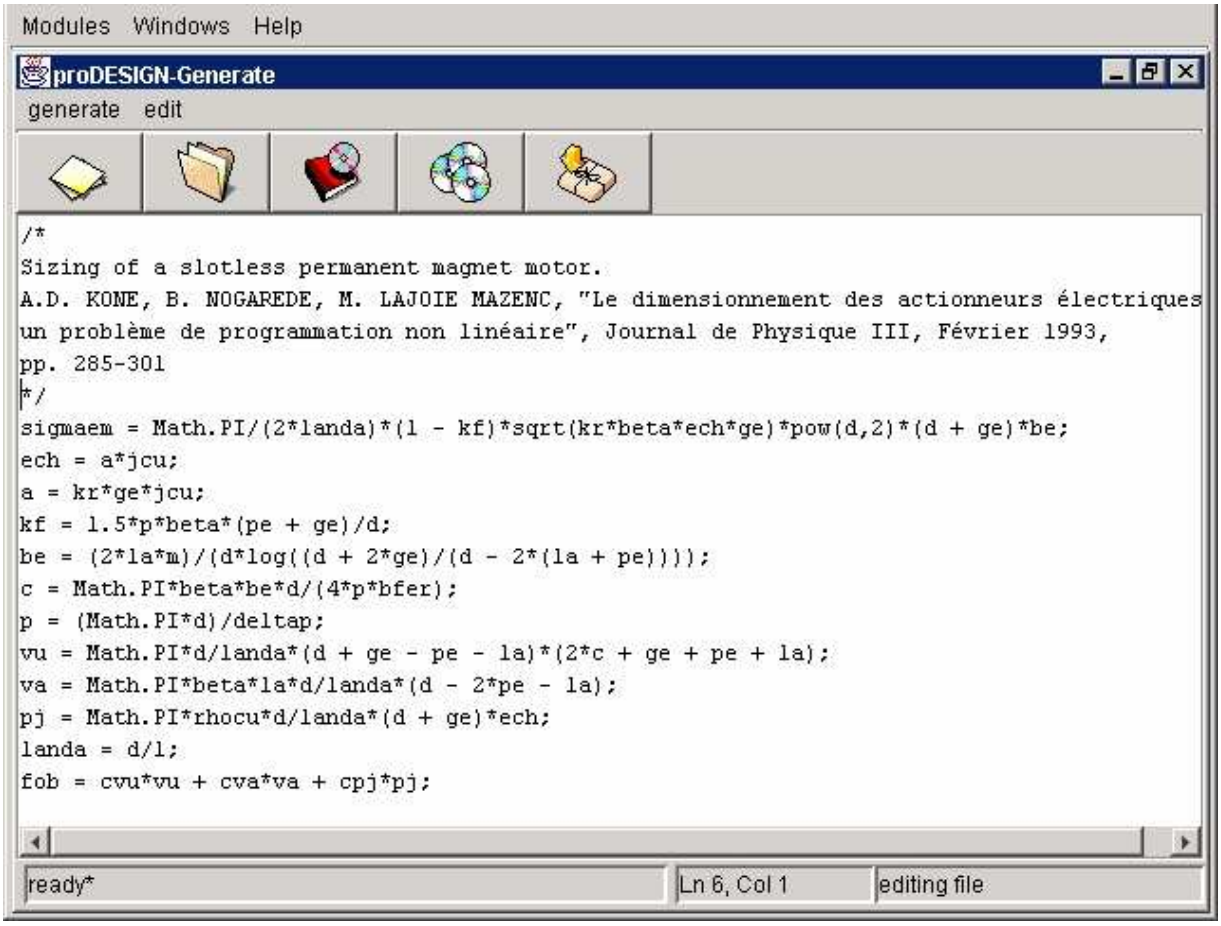


Figure 5 : Exemple de modèles analytiques

Ici donc un modèle saisi dans le langage par défaut du générateur. Ce langage permet de saisir toutes les équations analytiques explicites d'une publication [Kone-93].

III.A.1.b-UN MODELE MULTILANGAGE

Il est possible de fournir à ce module d'autres composants genX qui définissent chacun un langage différent. Ainsi il est possible de rédiger le modèle en utilisant plusieurs langages différents. Nous en avons implémenté trois.

Un premier pour décrire une équation différentielle. Les équations différentielles sont très utiles en modélisation pour décrire par exemple des transitoires, ou (et c'est fortement similaire) des phénomènes dissipatifs.

Un deuxième a été développé pour décrire un ensemble de fonctionnelles utiles. Ces fonctionnelles sont du type interpolation d'une courbe décrite à partir de quelques points, interpolation d'une nappe à partir de quelques points, extraction d'une racine d'une fonction (trouver x tel que f(x) = 0). Calcul d'une dérivée partielle.

Un troisième qui consiste à décrire un système d'équations implicites. Une équation implicite se définie par :

$$\begin{cases} f_1(x_1 \cdots x_n) = 0 \\ \vdots \\ f_n(x_1 \cdots x_n) = 0 \end{cases}$$

Il s'agit alors de trouver les x_i tels que les f_i soient nulles. Ceci arrive fréquemment en modélisation lorsque l'on ne parvient pas à exprimer analytiquement une des variables en fonction des autres. C'est le cas par exemple lorsque l'on prend en compte la saturation dans les matériaux magnétiques, et que l'on utilise une méthode par réseau de réluctance. En effet, lorsque l'on écrit les loi des mailles et les lois des nœuds pour le circuit réluctant, il en résulte un ensemble de relations entre les flux du circuit. Si toutes les réluctances sont linéaires (c'est à dire indépendantes du flux qui les traverse) alors la relation qui lie les flux aux réluctances est une relation multilinéaire. En revanche si l'on fait intervenir une non linéarité, ce qui se traduit par une dépendance de la réluctance par rapport au flux qui la traverse (le raisonnement est identique pour les Ampère-tours) il devient impossible de résoudre dans le cas général les équations.

Dans un cas à une dimension on voit bien que

Réluctances linéaires :

$$nI = \Re \cdot \phi$$ **Équation 31 : Définition d'une réluctance**

Réluctances non-linéaires:

$$nI = \Re(\phi) \cdot \phi$$ **Équation 32 : Généralisation à une réluctance non-linéaire**

Dans tous les cas il faut résoudre ces équations pour connaître le flux. Dans le cas linéaire cela peut être résolu analytiquement.

$$\phi = \frac{nI}{\Re}$$ **Équation 33: Solution dans le cas linéaire**

Tandis que dans le cas non linéaire, on ne peut pas trouver de solution pour le cas général, il faut faire intervenir l'expression analytique de la réluctance en fonction du flux. Cette expression est à la fois très dépendante du matériau, et de sa modélisation. Si toutefois il était possible de résoudre cette équation, il n'est pas souhaitable de le faire sans un outil qui permettrait de répéter l'opération. En effet, le moindre changement dans les propriétés du matériau, ou bien dans la technique de modélisation des matériaux, ou bien dans l'expression de la réluctance change complètement l'expression analytique de la solution, et tout le calcul est à refaire.

En conséquence, dans les cas de non linéarité des matériaux il est recommandé d'utiliser une description qui reste stable au changement de réluctances, de matériaux ou de modélisation des matériaux.

$$\phi = x \left| \Re(\phi) \cdot \phi - nI = 0 \right.$$ **Équation 34 : Solution dans le cas non linéaire**

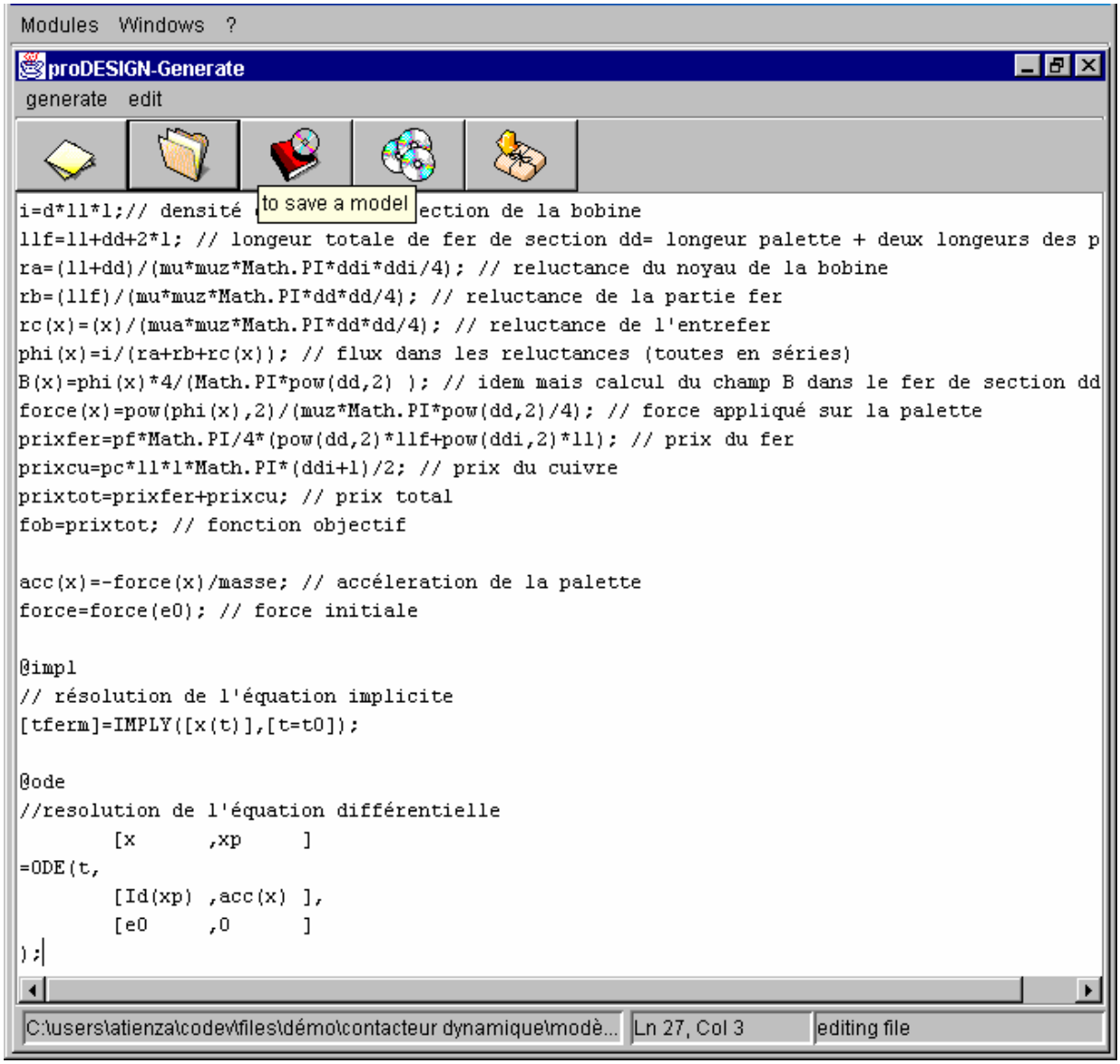


Figure 6: modèle dans le logiciel avec équations implicite et équations différentielles

Dans cet exemple (fig ci-dessus) on utilise conjointement les équations différentielles (langage "ode") et les équations implicites (langage "impl"), et le langage par défaut. Nous modélisons tout d'abord un contacteur par une méthode des schémas réluctants. Nous en déduisons une force en fonction de l'entrefer. Nous utilisons cette force dans le Principe Fondamental de la Dynamique, pour en déduire une équation différentielle qui permet de connaître la position de l'entrefer en fonction du temps (fonction x(t)). Seulement la connaissance de la position en fonction du temps ne nous intéresse pas directement, mais plutôt le temps de fermeture, à savoir l'instant pour lequel l'entrefer vaut 0. C'est ce qui est exprimé par l'équation implicite.

III.A.2- L'OPTIMISEUR

III.A.2.a- UN OPTIMISEUR PAR DEFAUT

Le dimensionnement, ou le prédimensionnement s'effectue en exprimant le cahier des charges de dimensionnement, en cahier des charges pour l'optimisation. Il s'agit en fait de

mettre en relation le code généré dans le COB, et le code de l'algorithme d'optimisation. Nous maîtrisons le code que nous générons, mais nous ne maîtrisons pas le code des algorithmes d'optimisation. Une première approche [Wurtz-96] à consisté à générer un code qui s'adapte complètement à un algorithme d'optimisation particulier. L'inconvénient d'une telle approche est que tous les codes générés deviennent complètement obsolètes si l'on souhaite changer d'algorithme d'optimisation. Nous avons choisi une méthode plus générale, en définissant d'une part le composant COB qui fournit la capacité de calcul du modèle indépendamment des algorithmes d'optimisation, et d'autre part un contrat pour les algorithmes d'optimisation. Ainsi il a été possible d'écrire un programme qui met en relation le code généré et les codes des algorithmes d'optimisation. Ce programme permet à partir des informations du COB (liste des paramètres d'entrée et de sortie, calcul des ces paramètres et de leur dérivées), de spécifier un cahier des charges de dimensionnement. Ce cahier des charges est automatiquement traduit en cahier des charges d'optimisation. Il suffit alors d'écrire un programme qui adapte un algorithme d'optimisation donné au cahier des charges d'optimisation. Nous avons adapté deux algorithmes d'optimisation qui étaient disponibles sur le web [Harwell-87, Lawrence]. Un algorithme a été écrit directement pour ce formalisme [Guichon-01]

Nous allons voir comment se présente une interface graphique fondée sur ces deux composants.

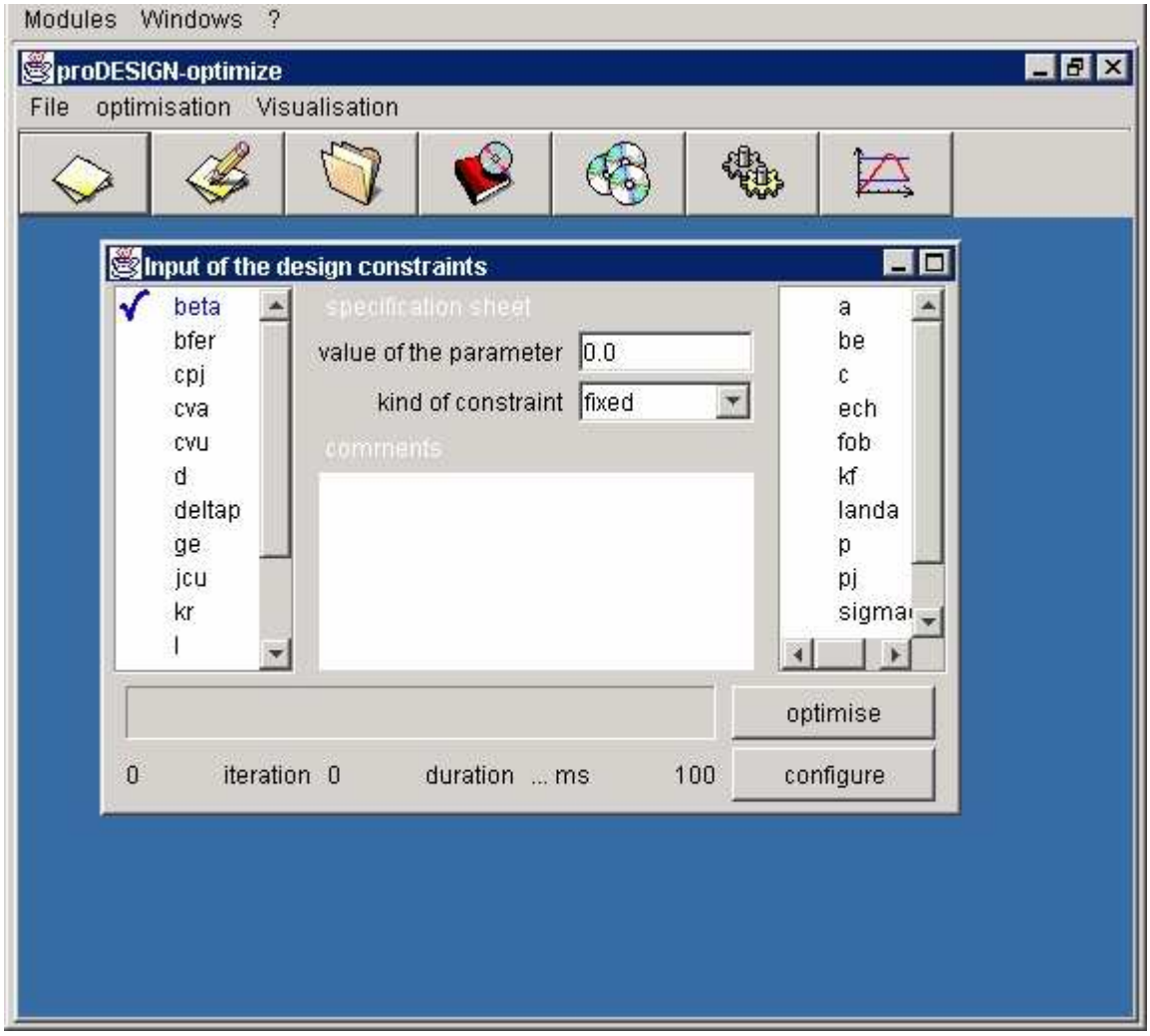


Figure 7 : Saisie du cahier des charges

Tout d'abord le modèle, sous la forme d'un fichier COB est ouvert dans l'interface graphique. La liste des paramètres du modèle est alors disposée en deux colonnes, à droite les paramètres de sorties, à gauche les paramètres d'entrée. Au milieu se trouve les informations associées au paramètre sélectionné, qu'il s'agisse d'un paramètre d'entrée ou d'un paramètre de sortie.

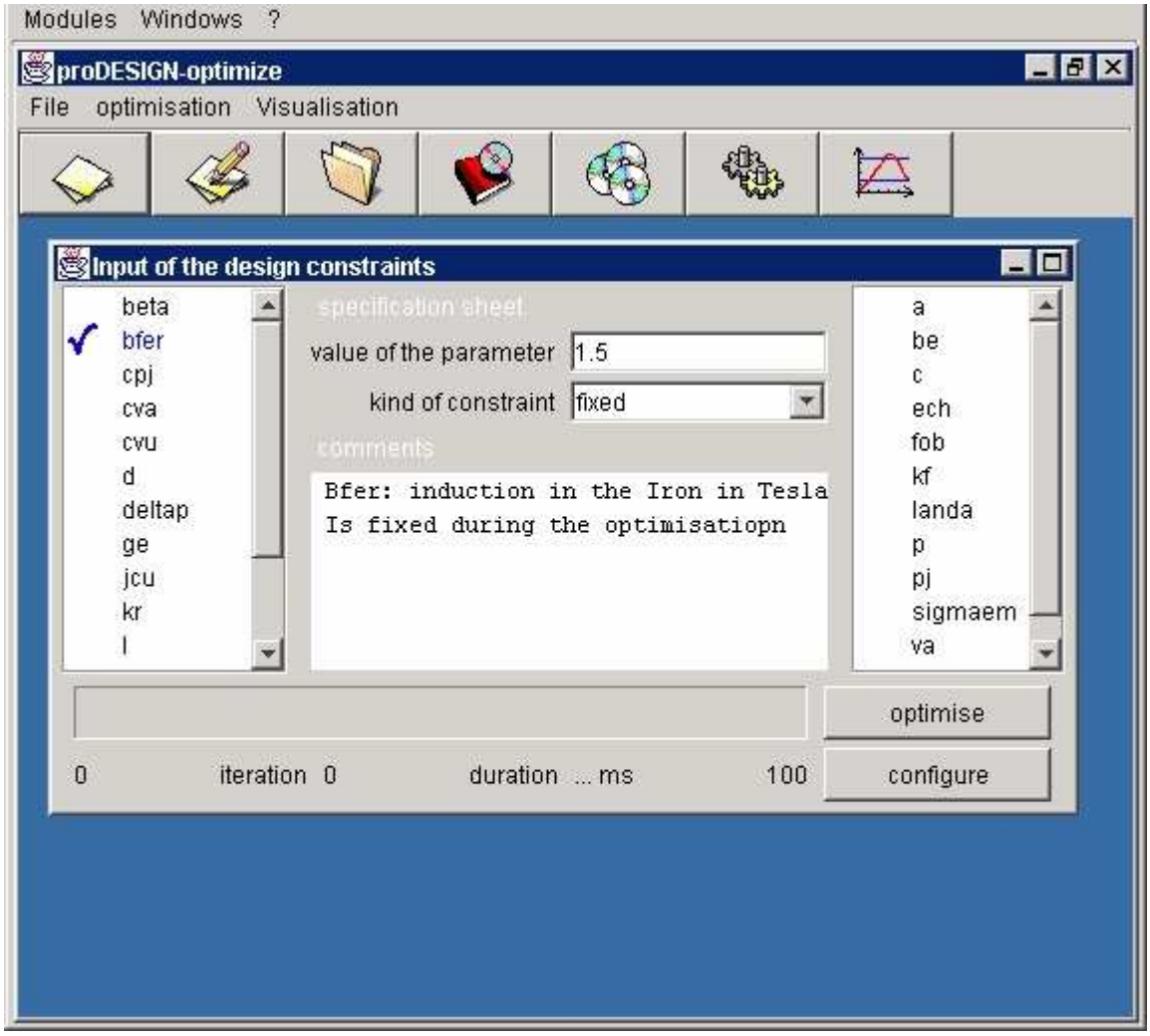


Figure 8 : Paramètres d'entrée fixe

Les paramètres d'entrée peuvent être de deux types : optimisable ou fixe. Si le paramètre est fixe (ci-dessus) alors il suffit de donner la valeur du paramètre d'entrée.

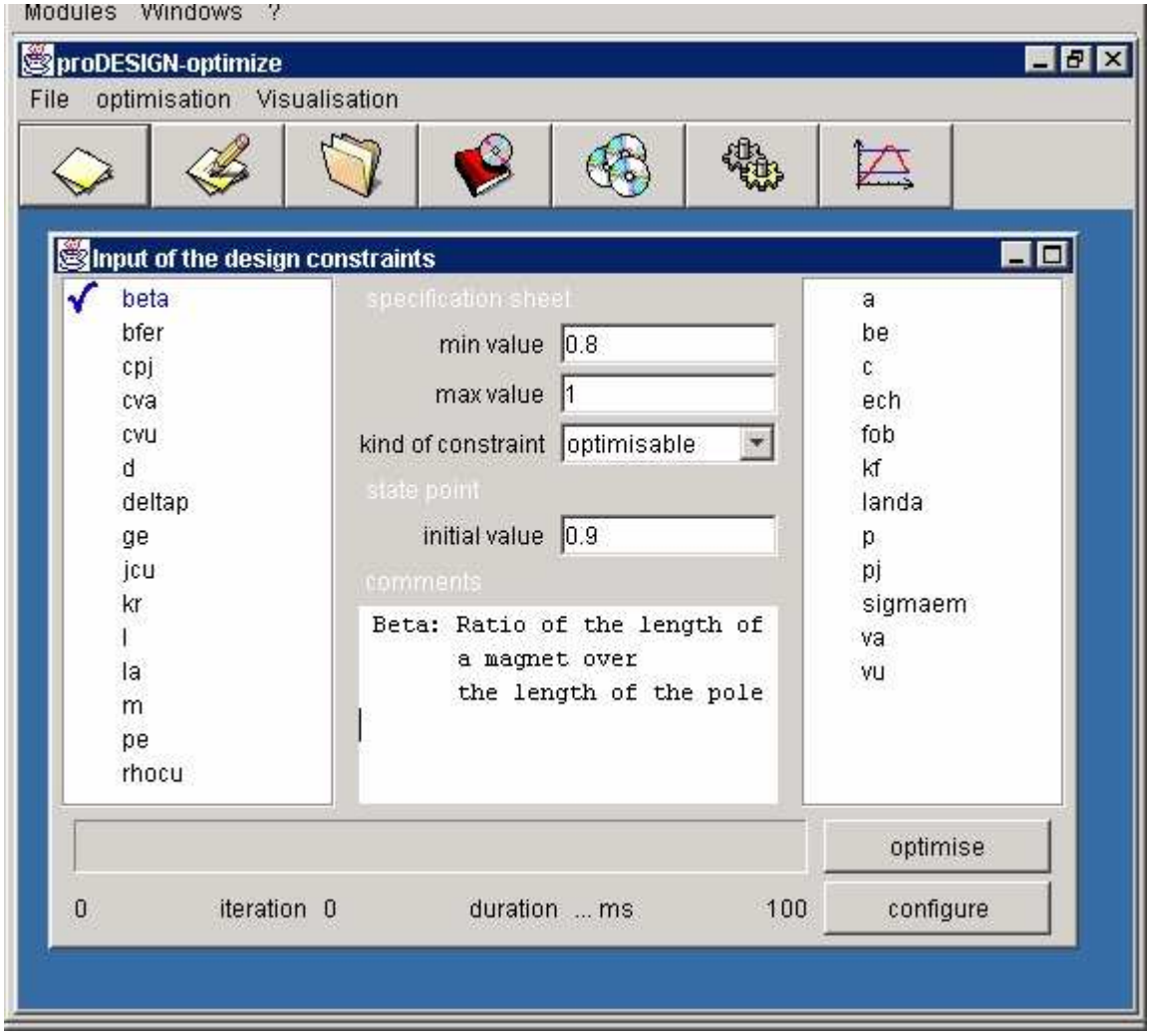


Figure 9 : Paramètre d'entrée optimisable

De même si le paramètre d'entrée est optimisable il suffit de donner l'intervalle dans lequel va évoluer la valeur de ce paramètre. Dans certains cas les utilisateurs ne souhaitent pas contraindre le paramètre sur un intervalle, mais simplement à être supérieur à une valeur (ce qui revient à le contraindre sur une demi droite). Il est néanmoins conseillé dans ce cas de ne pas trop exagérer la valeur de la borne qui ne doit pas être contrainte, car nous utilisons ces valeurs données par l'ingénieur pour normaliser le modèle. La normalisation est une nécessité en optimisation. En effet les algorithmes d'optimisation exigent pour la plupart des valeurs normalisées. Par exemple, l'ingénieur donne comme contrainte que le ratio entre la longueur d'un aimant sur la longueur du pole (fig) varie entre 0.8 et 1. Ce langage est celui de l'ingénieur électricien, mais le langage de l'optimisation est tout autre : CFSQP par exemple ne peut pas comprendre ce genre de demande, il sait seulement qu'un paramètre doit être positif. C'est l'environnement qui fait alors le travail de transformer les concepts, le paradigme de l'ingénieur électricien en celui de l'ingénieur numéricien qui a réalisé l'algorithme.

Les paramètres de sortie fonctionnent sur un principe similaire, les types disponibles pour les paramètres de sortie sont : ignoré, contraint, fixé, objectif.

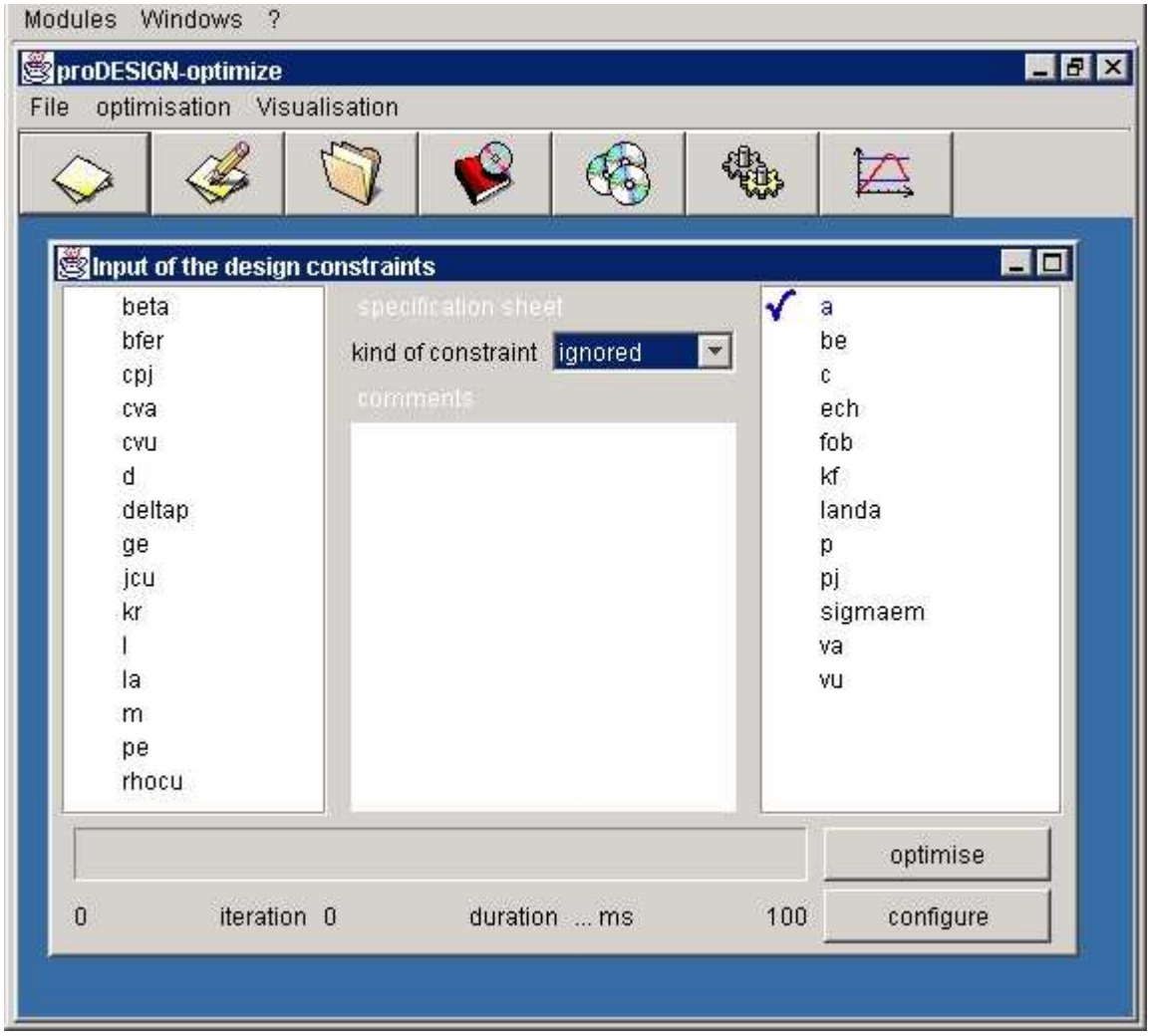


Figure 10 : Paramètre de sortie ignoré

Lorsqu'un paramètre de sortie est ignoré, il ne requiert aucun paramètre pour l'optimisation, ce paramètre n'est pas transmis à l'algorithme d'optimisation, et il ne nécessite pas de configuration. En revanche de part la structure composant, le COB ne sait pas que l'algorithme d'optimisation n'utilise pas ce paramètre. Il le calcule comme autre paramètre. D'ailleurs de manière plus générale, le COB n'est pas informé de la façon dont il est utilisé, par conséquent il calcule tous les paramètres de sortie.

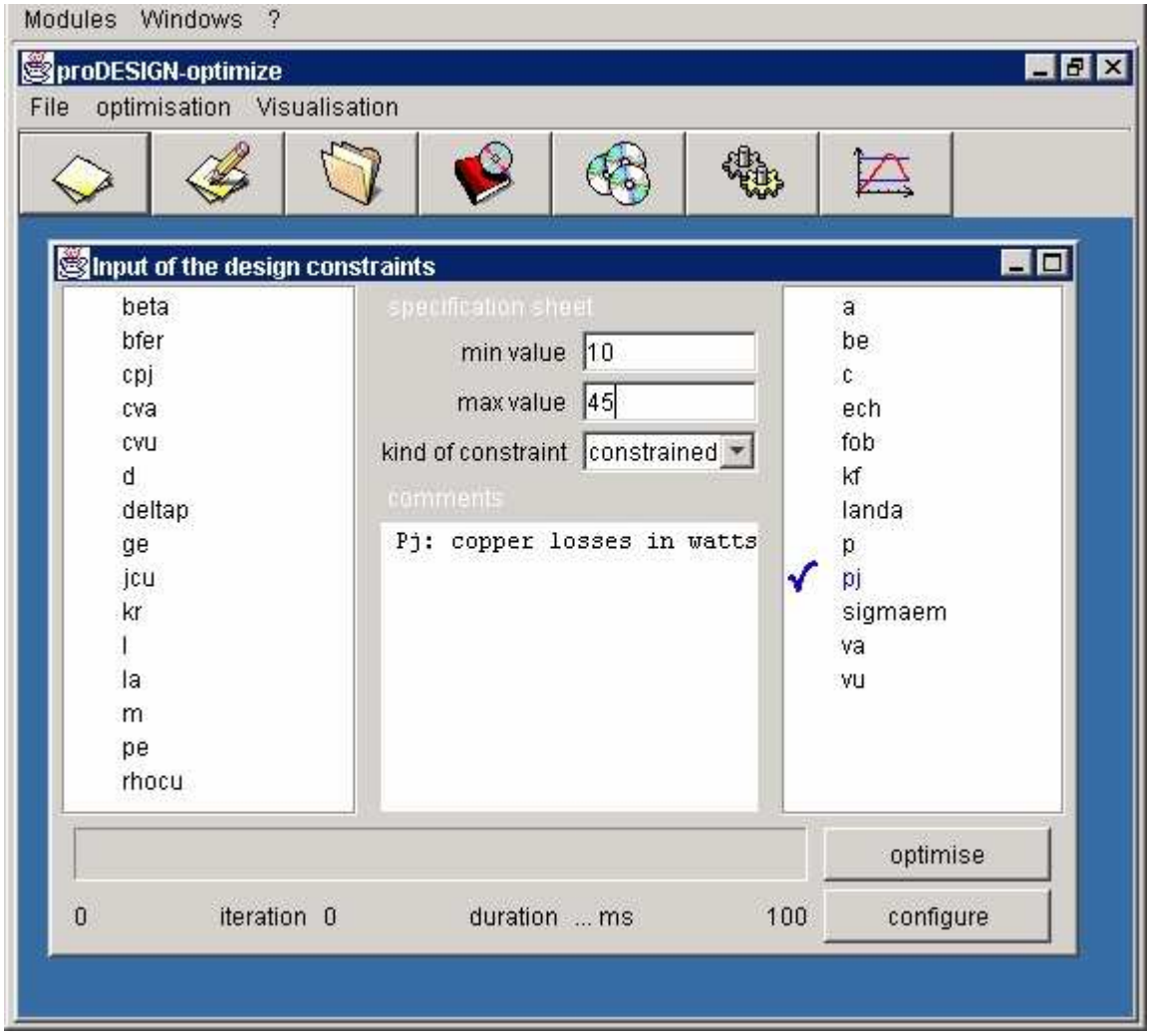


Figure 11 : Paramètre de sortie contraint

Quand un paramètre de sortie est dit contraint, c'est qu'il est autorisé à varier dans un intervalle seulement. De la même manière que pour les paramètres d'entrées optimisables, les valeurs sont utilisées pour la normalisation. Il est donc souhaitable de donner un intervalle plutôt serré, et de l'élargir s'il s'avère qu'une des bornes arrive en butée.

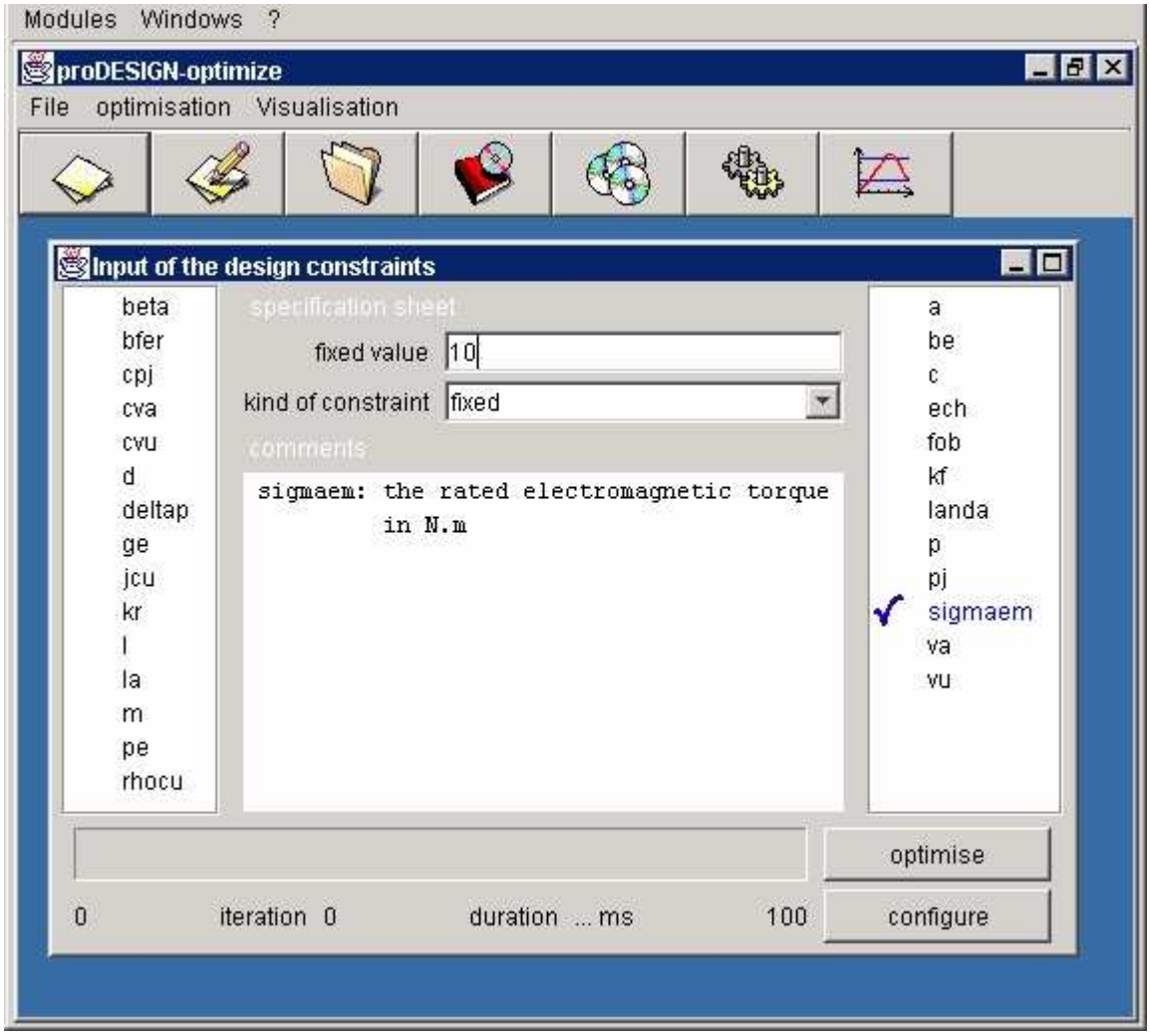


Figure 12 : Paramètre de sortie fixé

Une contrainte fixe en sortie ne prend qu'un paramètre. La normalisation appliquée consiste à rajouter 50% au paramètre s'il est non nul, et à ne pas le normaliser s'il est nul. Une contrainte fixe est très puissante pour le concepteur mais c'est aussi très lourd pour les algorithmes d'optimisation. Il faut les utiliser avec parcimonie, en effet si l'on considère l'espace des solutions, c'est un ensemble de taille n, où n est le nombre de paramètres d'entrée optimisables. Si l'on regarde la dimension de l'espace des solutions qui satisfont les cahier des charges, alors il apparaît que les contraintes d'égalité ne réduise pas la dimension de l'espace, mais seulement son étendue (comme un carré dans un carré plus grand), en revanche les contraintes fixes sur les paramètres de sortie réduisent à chaque fois l'espace de une dimension (soit une surface infiniment petite). La probabilité de tirer au hasard une solution satisfaisant le cahier des charges tombe à zéro. Dans la pratique nous ne fixons pas la contrainte strictement, mais nous donnons un intervalle de tolérance, ce qui a pour effet de conduire à une probabilité non nulle de tomber sur un point acceptable. Cette probabilité a été évaluée selon les cas de 1 pour 10 000, à 1 pour 1 000 000. Cette probabilité donne une approximation de la surface constituée des points acceptables par rapport à celle des points atteignables.

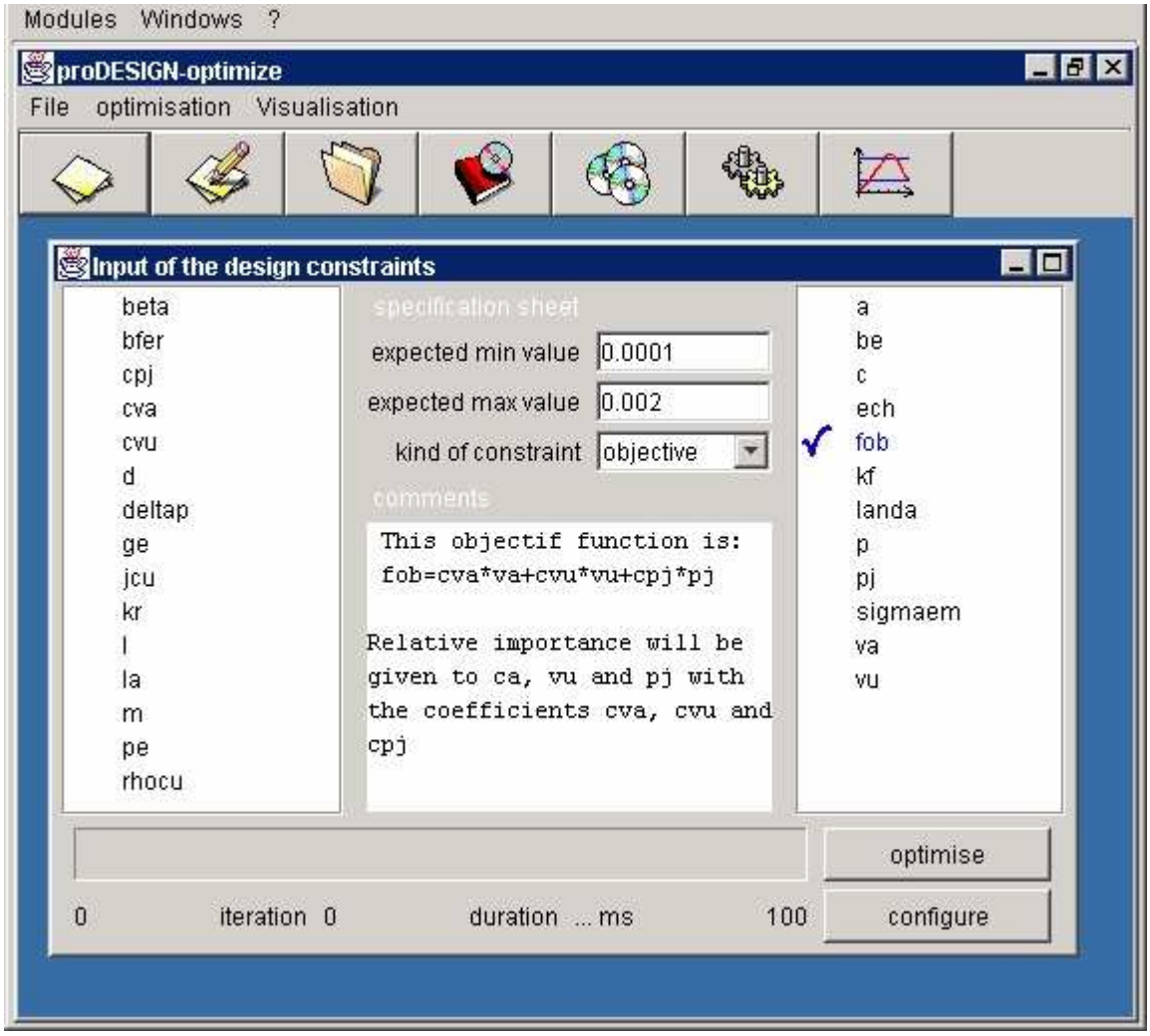


Figure 13 : Paramètre de sortie objectif

Pour finir un paramètre de sortie peut être une fonction objectif pour ce problème d'optimisation. Dans le cas général nous autorisons plusieurs fonctions objectif. Dans ce cas la solution à une optimisation multi fonction objectif dépend souvent de l'algorithme. Certains algorithmes ne savent pas optimiser plusieurs fonctions objectif.

paramètre	état	Description
Entrée	Fixe	Le paramètre de peut pas évoluer au cours de l'optimisation, c'est une constante pour le dimensionnement
	Intervalle	Le paramètre peut évoluer librement à l'intérieur d'un intervalle donné
Sortie	ignoré	Le paramètre peut évoluer librement sans aucune contrainte. Ces paramètres sont le plus souvent des valeurs qui servent à vérifier la cohérence du modèle, ou fournir des informations sur la solution proposée.

	contrainte	Le paramètre peut évoluer librement mais à l'intérieur d'un intervalle donné.
	fixe	Le paramètre doit égaler une valeur donnée. Ce type de contrainte est très coûteuse en optimisation, mais elle est nécessaire dans certains dimensionnement.
	objectif	Le paramètre est une fonction objectif de l'optimisation. Il n'est contraint à aucune valeur en particulier, mais il doit être minimisé.

Tableau 2 : récapitulatif des états d'un paramètre dans un cahier des charges

III.A.2.b-CHANGEMENT D'ALGORITHME

Dans certains cas l'algorithme d'optimisation par défaut ne converge pas, ou même ne parvient pas à attaquer le problème. Dans d'autre cas, les temps de calcul extrêmement courts suscitent l'envie de tester de nombreux points de départ pour pouvoir trouver des solutions un peu plus innovantes, ou en tout cas plus indépendantes du point de départ que nous pouvons fournir. C'est pourquoi il est possible de changer d'algorithme d'optimisation. La technologie composant utilisée ici pour définir l'algorithme qui va prendre en charge le service d'optimisation rend le changement d'algorithme très facile.

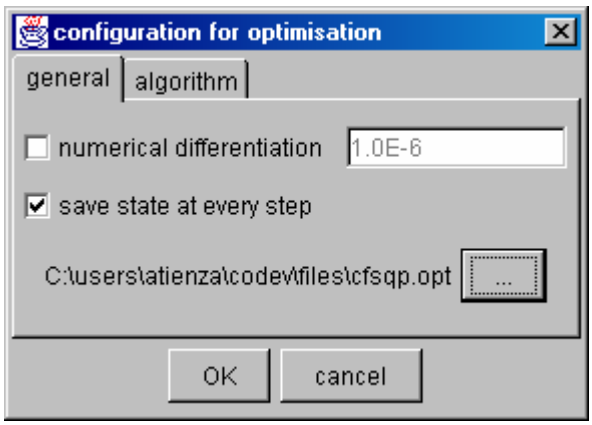


Figure 14 : Choix de l'algorithme d'optimisation

Pour changer d'algorithme d'optimisation il suffit de choisir le fichier qui contient le nouveau composant d'optimisation que l'on veut charger. Dans cet exemple nous avons choisi cfsqp.opt un composant d'optimisation qui se fonde sur CFSQP, un algorithme développé par l'université du Maryland. Dans ce cas la configuration du composant est accessible graphiquement.

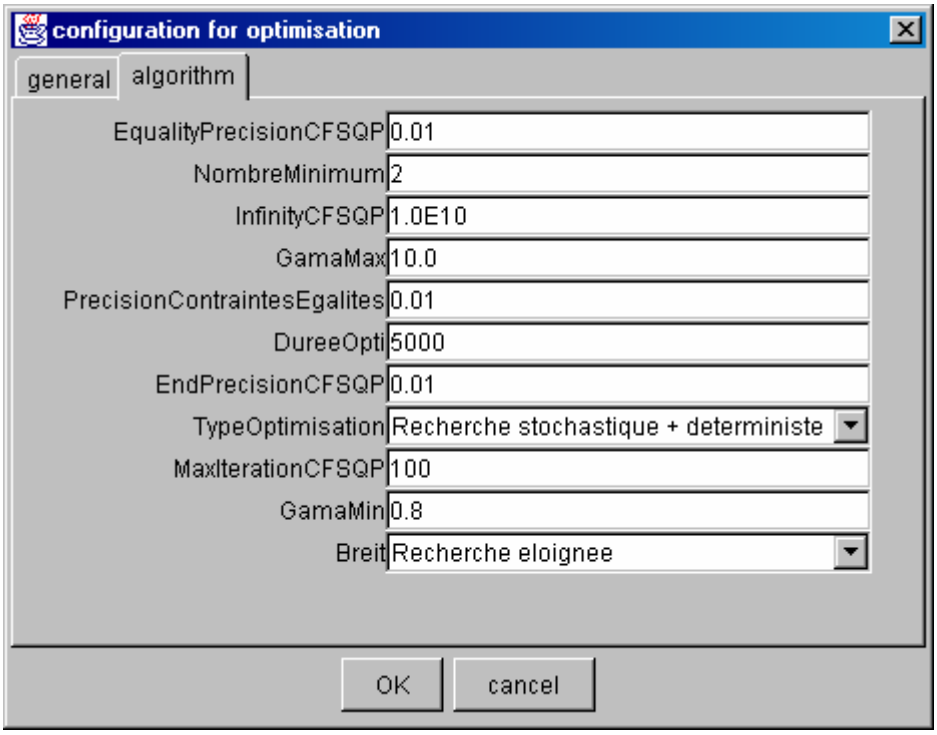


Figure 15 : Configuration d'un algorithme d'optimisation BreitWigner

Ici nous avons la configuration complète d'un algorithme dit de BreitWigner [Guichon-01], algorithme d'optimisation mixe stochastique-déterministe. Comme nous l'avons vu, les algorithmes purement stochastiques peinent à trouver des solutions satisfaisant le cahier des charges quand celui-ci contient des contraintes d'égalité (dans le cas présenté jusqu'à présent il y a deux contraintes d'égalité, et 15 000 itérations n'ont pas suffi à trouver une solution satisfaisante). C'est pourquoi il utilise les deux méthodes, avec la possibilité de faire une optimisation purement stochastique si cela est utile dans certains cas, et la possibilité de mixer un tirage aléatoire et un algorithme déterministe. Plutôt que de réimplémenter un algorithme déterministe performant, il réutilise le composant CFSQP décrit plus haut et utilise l'algorithme d'optimisation qu'il contient.

III.A.3- L'ENVIRONNEMENT

A l'instar de chacun des modules présentés la spécification de composant COB offre un certain nombre de possibilités. Nous en avons implémenté quelques exemples durant cette thèse pour valider par le menu ces possibilités.

III.A.3.a-PASSERELLE

J'appelle passerelle un morceau de programme qui s'insère dans un autre logiciel, et offre la possibilité d'utiliser un composant. Par exemple un programme qui permet d'exploiter les COBs dans Mathcad®.

Cette technique n'est utilisable que sous certaines conditions liées à l'ouverture des environnements cibles. Il faut que l'environnement cible permette à un programme extérieur de s'intégrer dans leur modèle. Cela est prévu dans Matlab® et dans Mathcad® mais pas de la même manière. En effet Matlab dispose d'une machine virtuelle Java d'une part, et il permet de passer toutes les informations que l'on veut aux programmes extérieurs (des chaînes de caractères , des tableaux de nombres etc), tandis que Mathcad est plus fermé, car il n'autorise

le passage que de tableaux de nombres complexes. Il est impossible alors de choisir un COB par son fichier, de valuer les paramètres en donnant leur nom.

Nous avons développé une passerelle vers Mathcad® afin de pouvoir valider le modèle rentré dans ProDESIGN le plus précisément par rapport au modèle développé directement dans Mathcad®.

III.A.3.b- TRANSFORMATEUR

J'appelle transformateur tout programme qui transforme un COB dans un autre composant d'un autre format. Par exemple un programme qui lit un COB et en fait un JavaBean® est un tranformateur.

Deux transformateurs [Delinchant-00] ont été développés vers des standards de composants. Le transformateur COB vers JavaBean© permet d'utiliser les COB dans un environnement dédié à la conception graphique de logiciel. Le transformateur COB vers CORBA, permet de donner une définition IDL [Delinchant-00] du composant COB est de fournir un Objet CORBA qui permet de déployer des COB sur des systèmes hétérogènes.

On peut faire un transformateur sous certaines conditions seulement. Il faut que l'environnement cible soit ouvert à l'intégration de composant. Il faut également que le format de composant de l'environnement cible soit compatible avec les COB. Il ne s'agit pas d'une compatibilité formelle ou syntaxique, mais d'une compatibilité fonctionnelle. Il faut pouvoir donner un sens à l'exploitation d'un COB dans ce nouvel environnement !

III.A.3.c- CLASSES EXTERNES

Une classe externe est une classe Java, un objet compilé, qui va contenir une fonction telle que définie par proDESIGN. Pour notre logiciel une fonction est un couple (f , df) où f est la fonction proprement dite, qui renvoit un réel en fonction de N réels, et df est la différentielle de f. Dans ce cas il est possible de coder une fonction particulière dans une classe Java et de l'utiliser dans le modèle analytique.

Les classes externes sont une ouverture qui permet de contourner un certain nombre de contraintes induites par l'utilisation exclusive de modèles analytiques.

Les classes externes permettent de coder une fonction qui ne s'exprime pas mathématiquement à partir de celles fournies en standard. Pour bien comprendre, il faut se rappeler que même les fonctions élémentaires comme sinus, cosinus, tangente sont le résultat d'un programme (souvent implanté directement dans le processeur, mais néanmoins un programme) de calcul. Il n'est pas possible de prévoir toutes les fonctions dont auront besoin les utilisateurs, et même pire, il y a des fonctions que l'on ne peut pas ramener à une fonction générique. Par exemple la courbe B(H) utilisée dans une modélisation n'est pas générique, et selon les matériaux et selon les modélisateurs, elle peut varier d'un modèle à l'autre. Donc les classes externes permettent à l'utilisateur expert de programmer des fonctions mathématiques que nous aurions omises, ou des fonctions propres à son métier.

Les classes externes permettent également d'utiliser un autre logiciel de calcul pour renvoyer la valeur d'une fonction. Par exemple, il a été possible d'utiliser INCA pour calculer toutes les répartitions de courant dans un jeu de barre [Guichon-01], et écrire les fonctions en Java qui renvoyaient les valeurs des courants dans chacune des barres, en fonction des positions respectives des barres, et des tensions. Il en résulte que l'on pouvait exprimer ces fonctions dans le modèle analytique et bénéficier de la génération de code, et des modèles

analytiques pour exprimer par exemple les contraintes sur les dimensions pour que les barres ne se chevauchent pas, ou bien pour exprimer la fonction objectif.

III.A.3.d-NOUVEAU LANGAGE

La structuration en composant du générateur même, permet *a priori* d'ajouter librement des langages de modélisation et de bénéficier de la génération de code des autres langages.

Nous avons par avance conçu trois langages, le langage par défaut des équations analytiques explicites, le langage des équations différentielles et le langage des équations implicites. En cours d'utilisation, plusieurs personnes ont regretté l'absence de certaines fonctionnalités comme la racine d'une fonction, ou l'interpolation d'une fonction à une dimension ou a deux dimensions [Larouci-02]. Nous avons répondu à ce besoin en écrivant un nouveau langage, le langage des fonctionnelles qui contient ces trois fonctions.

Pour implémenter un nouveau langage, il suffit d'écrire un composant gen-X , qui s'articule le plus souvent autour d'un parser et d'un générateur. Le parser va lire le modèle écrit dans le langage associé au composant (c'est donc lui qui définit le langage), et lire dans le modèle les informations nécessaires au générateur. Pour faciliter l'écriture d'un parser nous utilisons javaCC, générateur de parser, technologie usuelle en informatique, et cdiCC, générateur de générateur développé au cours de ma thèse. CdiCC est un langage qui permet de décrire un générateur directement par le code généré, en indiquant dans un fichier proche du code généré les informations nécessaires à la génération d'un générateur.

III.B Processus

On peut montrer quelques processus faisables avec cette architecture composant, et surtout voir l'intérêt d'une telle architecture pour ces processus.

III.B.1- LE DIRECT

Le processus le plus naturel d'utilisation de ProDESIGN consiste à saisir un modèle existant sous ProDESIGN, puis à vérifier que le calcul est conforme aux simulations, ou aux mesures faites pour mettre au point ce modèle, puis à optimiser ce modèle sur un cahier des charges donné.

Seulement, dans un tel processus on perd un grand intérêt de l'outil et de la méthode des modèles approchés qui consiste à aller vite, pour aller plus loin. En effet le temps de saisie du modèle dans ProDESIGN, quoique rapide, reste quand même une phase fastidieuse qui risque de rebuter des ingénieurs intéressés par se concentrer sur leur métier.

De plus, l'optimisation de modèle introduit avec force une notion parfois oubliée en modélisation, c'est la notion de phénomène prépondérant. En effet, lorsque l'algorithme d'optimisation tente de résoudre le cahier des charges sur le modèle qui lui a été donné, il ne s'encombre pas de considération sur le fait que la solution est absurde ou non. Il apparaît donc que si l'on oublie un phénomène prépondérant, l'optimisation conduit à une solution absurde.

Si concevoir c'est résoudre un certain nombre de compromis entre différents phénomènes, l'optimisation sanctionne un phénomène oublié, ou plus positivement, révèle un phénomène à ne pas négliger.

Par exemple, quel est le phénomène qui limite le nombre de spires dans la bobine d'un contacteur ? Il ne sert à rien de modéliser les capacités entre spires, la forme de la surtension

de foudre pour se rendre compte que dans certains cas le phénomène prépondérant est le nombre d'Ampère-tours que l'on veut voir sur le plan magnétique. Il est plus efficace d'optimiser le modèle en ne tenant compte que des phénomènes magnétiques, puis de vérifier s'il est pertinent de pousser la modélisation jusqu'à la capacité entre spire.

Sur un cas très simple comme celui du nombre de spires, il est rare que l'ingénieur découvre des phénomènes grâce à l'optimisation, en revanche lorsque le modèle met en jeu certains phénomènes contradictoires, d'autres concourants, il devient très délicat de prédire que l'on a pris en compte tous les compromis !

D'où l'apparition d'un nouveau cycle d'utilisation des composants. En effet, la mise au point du modèle ne se fait plus par rapport à un point de fonctionnement, voire des courbes correspondant à un dispositif, mais aussi par un cycle incrémental.

III.B.2- LE CYCLE INCREMENTAL

Une fois le cycle direct réalisé, le plus souvent l'utilisateur souhaite le répéter rapidement en apportant de petites modifications : il réalise alors un cycle incrémental.

Pour cela, il suffit de rajouter ou de modifier une équation sur le modèle analytique, et de régénérer un COB. Les modèles analytiques présentent un avantage sur toutes les autres formes de modélisation : il est possible de les concevoir maintenables. En effet, il est possible de structurer son modèle de sorte que l'on puisse en changer des morceaux entiers sans remettre en cause les autres parties du modèle. A l'instar des langages de programmation où il est possible d'écrire des codes maintenables, mais dont la maintenabilité de ces codes dépend de la qualité du programmeur, les langages de modélisation analytique permettent d'écrire des modèles dont la maintenabilité dépend de la qualité du modélisateur. Pour l'instant la taille des modèles analytiques (maximum 500 équations) rendent inutiles les méthodologies de la programmation objet qui tendent à aider le programmeur à écrire un code maintenable. Si l'usage des modèles analytiques s'intensifie il sera temps d'envisager des méthodes comme la programmation orienté objet, mais adaptées pour les modèles analytiques.

Une fois le nouveau COB généré, il suffit alors de l'ouvrir dans l'optimiseur en utilisant la fonction « Conception incrémentale ». Celle-ci va récupérer le cahier des charges précédents et l'adapter au nouveau COB. La règle de transformation est simple. Pour chaque paramètre d'entrée on cherche un paramètre d'entrée de même nom dans l'ancien cahier des charges. Si on en trouve un alors on copie la contrainte associée, sinon on cherche un paramètre de sortie ayant le même nom. Si on en trouve un, alors on adapte la contrainte sur le paramètre de sortie au paramètre d'entrée en utilisant la table suivante :

Sortie	Entrée
Fixé à X	Fixé à X
Intervalle [A,B]	Optimisable [A,B]

Tableau 3 : Table de conversion de cahiers des charges

De même pour tous les paramètres de sorties du nouveau COB on regarde si l'ancien cahier des charges contient un paramètre de sortie ayant le même nom. S'il en existe un, alors la contrainte sur ce paramètre est copiée sur celui du nouveau COB. Sinon, on regarde si l'ancien COB contenait un paramètre d'entrée qui portait ce nom. S'il en existe un alors on convertit la contrainte en utilisant le tableau précédent mais dans l'autre sens.

L’utilisateur retrouve alors l’essentiel de son cahier des charges, et le plus souvent n’a plus qu’à tenir compte des modifications qu’il a faites. S’il a introduit des paramètres, il lui suffit de les contraindre. S’il a supprimé des paramètres il n’a le plus souvent rien à faire.

CHAPITRE QUATRIEME – APPLICATIONS

« Si l'on sait exactement ce qu'on va faire, à quoi bon le faire ? »
Picasso (Pablo Ruiz)

IV- Applications

La méthode que nous proposons, a été conçue pour fonctionner sur des exemples industriels. La conception s'est faite itérativement entre le dimensionnement de dispositifs, et la réalisation du logiciel. Néanmoins nous présenterons les exemples suivants qui ont tous été choisis pour illustrer un des intérêts du logiciel.

IV.A Deux cas industriels

IV.A.1- LE R5

Dans le cadre d'un stage de D.E.A, effectué en 1999 par Benoît Roussin , en collaboration avec Schneider Electric, nous avons réalisé le dimensionnement d'un dispositif électrotechnique: le relais R5. Il s'agit d'un déclencheur pour disjoncteur. Au début de cette étude, nous ne connaissions pas le dispositif, et Benoît Roussin ne connaissait pas non plus la méthodologie de dimensionnement.

Cette étude s'est faite dans le cadre d'un DEA car elle avait essentiellement des objectifs de recherche, de validation de la démarche dans un cas industriel. Le relais R5 est un relais en courant propre, c'est à dire qu'il déclenche sur le courant directement issu du capteur de courant (ici un différentiel). Cette technique présente l'avantage d'économiser des sources d'énergie auxiliaires, des systèmes d'électronique pour servir de déclencheur électrique au déclencheur électromagnétique. En bref, elle permet d'économiser en coûts de fabrication, et en fiabilité ! La contrepartie d'un tel avantage est que la puissance nécessaire pour faire déclencher cet élément est plus élevée que celle des systèmes à sources d'énergies auxiliaires. Dans cette étude la question qui se posait était de savoir si l'on pouvait utiliser cette technologie pour des courants de déclenchement inférieurs à 15 mA, ce qui est très supérieur au seuil de déclenchement actuel du R5 qui est d'environ 25 mA. Le R5 est un déclencheur qui se décline dans une gamme qui ne contient pas de dispositifs pour les faibles courants. Pour ces courants d'autres technologies sont utilisées comme les déclencheurs à sources d'énergies auxiliaires. L'enjeu est de taille : étendre la gamme des R5 dans des valeurs à fort potentiel. En effet, plus le courant de déclenchement est faible, plus le dispositif est sensible et protecteur ; or les pays développés tendent à baisser le seuil du courant de déclenchement de la protection individuelle.

IV.A.1.a-PRESENTATION DU DISPOSITIF

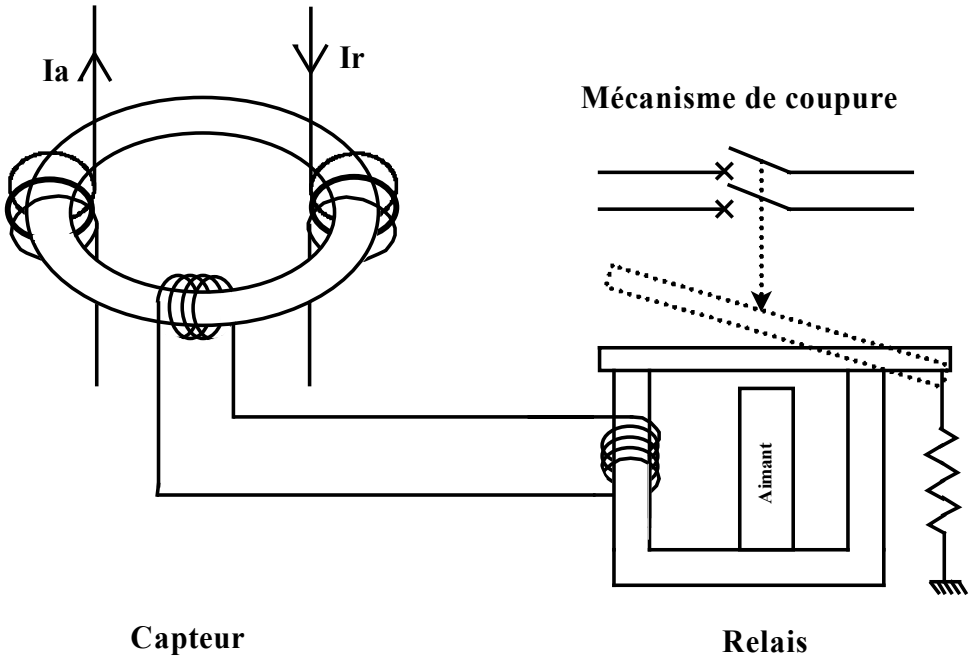


Figure 16 : La chaîne différentielle

Le déclencheur R5 est intégré dans une chaîne différentielle classique, avec un capteur différentiel (confer la figure ci-dessus), qui mesure la différence entre les courants d'entrée et les courants de sortie d'un système. Lorsqu'un défaut apparaît, un courant se crée dans la spire secondaire. Ce courant démagnétise localement l'entrefer1 (confer la figure ci-dessous) qui colle la palette sur le noyau, et le ressort peut alors ouvrir la palette. Ce mouvement est le premier de toute la chaîne mécanique qui va conduire à l'ouverture des contacts, et protéger le système.

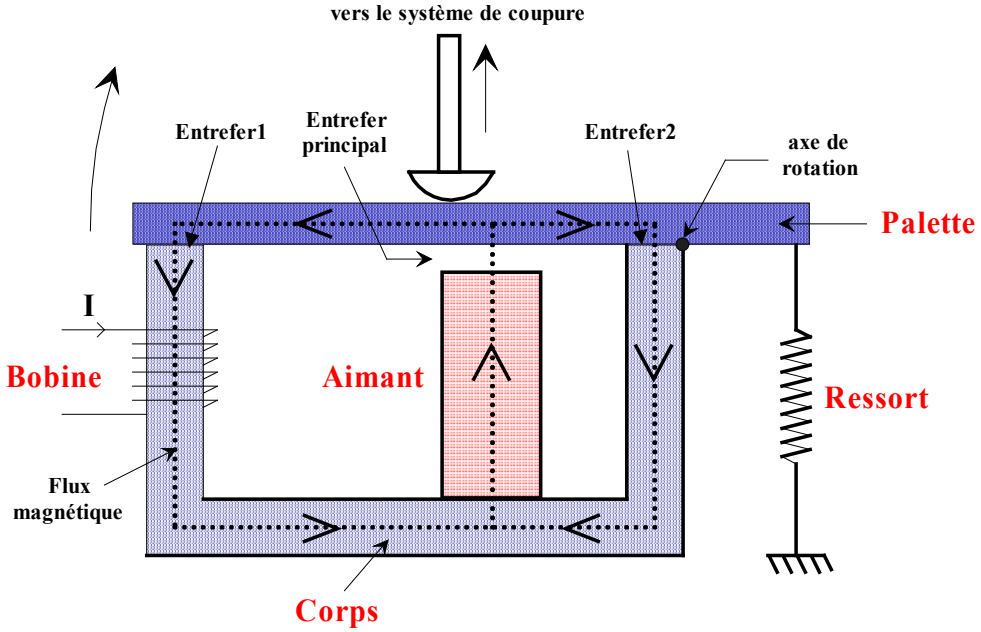


Figure 17 : Schéma simplifié du R5

IV.A.1.b-SCHEMA RELUCTANT

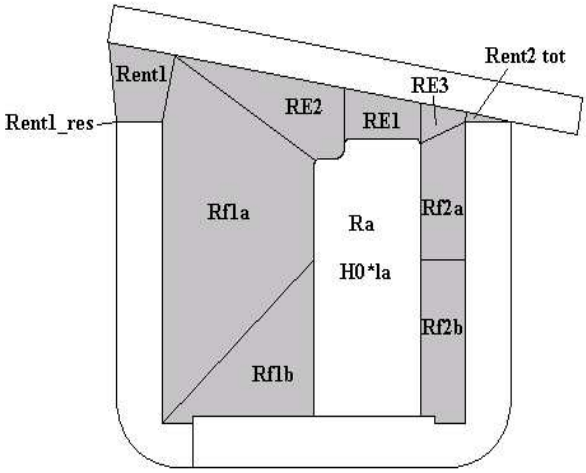


Figure 18 : Schéma réluctant du déclencheur pour angle d'ouverture <0.5

Le matériau ferromagnétique employé dans ce dispositif est suffisamment perméable pour négliger les effets du passage du flux magnétique dans le fer (le fer est considéré comme idéal). En observant les lignes de flux obtenues par simulation avec flux2D, il nous a semblé que l'on pouvait découper les lignes de flux en tubes assez simples (confer figure ci-dessus). Contrairement à l'impression que donne le schéma, ce modèle n'est utilisé que pour des valeurs d'angles d'ouverture inférieures à 0.5°. Ceci est très faible et revient à considérer la palette quasiment fermée.

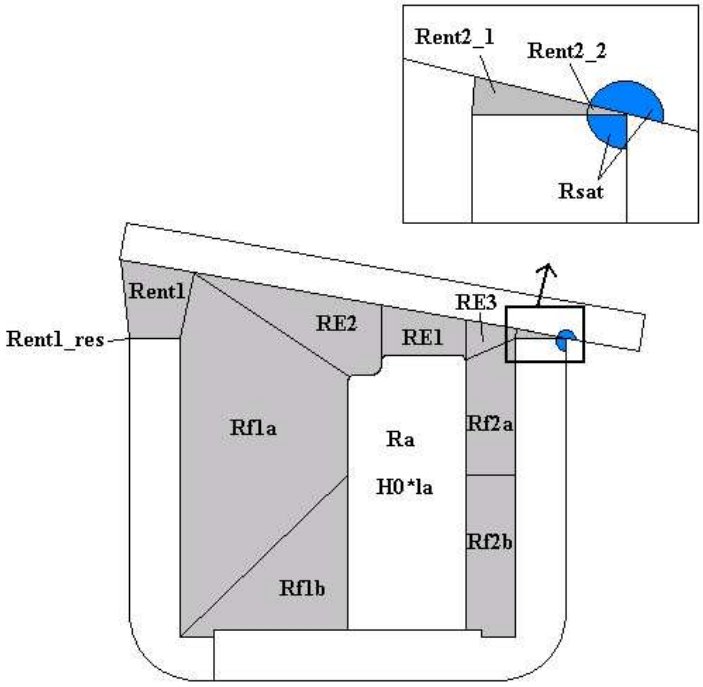


Figure 19 : Schéma réluctant du déclencheur pour angle d'ouverture >0.5

En revanche pour des angles d'ouverture supérieurs, nous avons dû considérer un autre schéma, en particulier pour la prise en compte de l'angle. En effet, en faisant l'hypothèse d'un fer idéal et non saturé, nous aurions obtenu un flux qui passe intégralement par le point de contact entre la palette et le noyau. Cette situation ne se peut absolument pas, et il faut donc ajouter la prise en compte de la saturation sur ce point de contact.

Le principe est simple, nous savons que la saturation, en induisant une non linéarité, crée une dépendance entre la réluctance de l'élément saturable, et le flux qui la traverse. Il faut donc conduire un calcul qui permet de connaître la valeur de la réluctance, connaissant le flux qui la traverse. Ainsi on pouvait partir de l'hypothèse qu'il y a un flux F qui traverse Rent2_1 et Rent2_2. De plus F se répartit entre Rent2_1 et Rent2_2 de sorte que Rent2_1 ne soit pas saturé. Nous avons fait l'hypothèse que le champ magnétique, dans la réluctance saturée, était constamment égal à Bsat, le niveau de saturation du matériau ferromagnétique employé. En posant comme inconnue x, la longueur de la réluctance Rent2_2, qui est saturée, on peut calculer le flux qui traverse Rent2_2 en fonction de x.

Pour l'entrefer Rent2_1, la géométrie de l'entrefer permet de prédire que la répartition du champ magnétique sera hyperbolique (en 1/n). La valeur au contact de la réluctance saturée est évidemment égale à Bsat (par continuité). On peut donc en déduire, en fonction de la longueur de cette réluctance (qui dépend de x), le flux qui passe dans cette réluctance.

On obtient ainsi le flux dans chacune des réluctances en fonction de x. Comme on connaît le flux total qui passe dans les deux réluctances (par hypothèse), on peut déterminer x, et donc la réluctance équivalente, en calculant simplement Ren2_2 avec la valeur de x trouvée.

IV.A.1.c- PRESENTATION DES RESULTATS

Les optimisations menées sur le modèle présenté ont conduit à des résultats prometteurs.

	Avant optimisation	**Après optimisation**
Puissance de déclenchement	0.56 W	0.23 W
Courant de déclenchement	23.5 mA	14.5 mA
Temps d'ouverture	4.5 ms	3.7 ms
Couple magnétique	6.52 N.mm	6.5 N.mm

Figure 20 : Résultat de l'optimisation

Cette étude illustre que dans un cadre industriel, la méthode fonctionne encore, et qu'elle peut répondre à des enjeux importants sans pour autant développer un modèle très lourd. La qualité de la personne qui réalise le modèle joue un rôle important dans la réussite de la méthode, néanmoins cette personne voit ses performances de concepteur amplifiées.

Trouver un système à courant propre qui déclenche à un seuil jusque là réservé au déclencheur avec source auxiliaire est un résultat remarquable. Le dimensionnement n'est pas une activité qui se cantonne à trouver des valeurs que le choix de structure n'a pas encore déterminée, non plus que d'optimiser une solution en vue de gagner quelques pour cents, mais bien une activité de la conception à part entière, qui peut apporter autant que la phase d'analyse, ou celle de choix de structure. L'exploration des dimensions d'une structure permet

de dégager des espaces pas toujours connus, sachant que l' on ne connaît bien une structure que lorsqu'on a pu explorer son espace de dimension pour avoir une idée de ses potentialités. Typiquement, dans ce cas, les systèmes à courant propres peuvent être employés pour des seuils de courants inférieurs à 15 mA.

IV.A.2- LE MINIMITOP

Dans le cadre d'un stage de D.E.A, effectué en 2000 par Cédric Osmond, en collaboration avec Schneider Electric, nous avons réalisé le dimensionnement d'un dispositif électrotechnique: le mini MITOP. Il s'agit d'un déclencheur pour disjoncteur. Au début de cette étude, nous ne connaissions pas le dispositif, et Cédric Osmond ne connaissait pas non plus la méthodologie de dimensionnement. Néanmoins, nous avons montré que dans un intervalle de temps aussi réduit qu'un stage, il était possible de réaliser la modélisation analytique, le dimensionnement, la vérification sous Flux2D, et la vérification sur un prototype. Cédric Osmond a même pu aller au-delà et fournir des résultats intéressants sur la répétition du cycle de conception.

IV.A.2.a-PRESENTATION DU DISPOSITIF

Le rôle de l'actionneur est de convertir une énergie électrique en une énergie mécanique, permettant le déclenchement d'un mécanisme de coupure. Il s'insère de la manière suivante dans la chaîne de coupure :

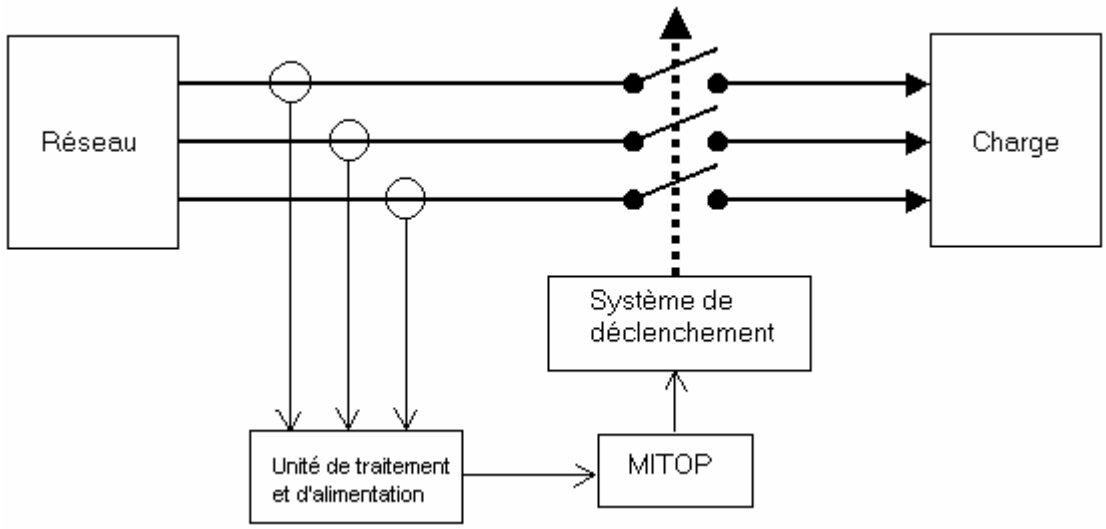


Figure 21 : Place du MITOP dans la chaîne de coupure

- Le MITOP est constitué des éléments suivants :
  - Un circuit magnétique constitué par :
  - la culasse.
  - le noyau ; celui-ci est mobile.
- la rondelle-shunt ; elle permet d'une part de concentrer le flux de l'aimant dans le noyau, et d'autre part de diminuer l'énergie électrique de déclenchement.

- Deux sources de flux : l'aimant, qui assure le collage du noyau à l'état repos, et la bobine qui permet la démagnétisation de l'entrefer en cas de défaut sur la ligne.
- Un ressort qui fournit l'énergie mécanique nécessaire au déclenchement du système de coupure.

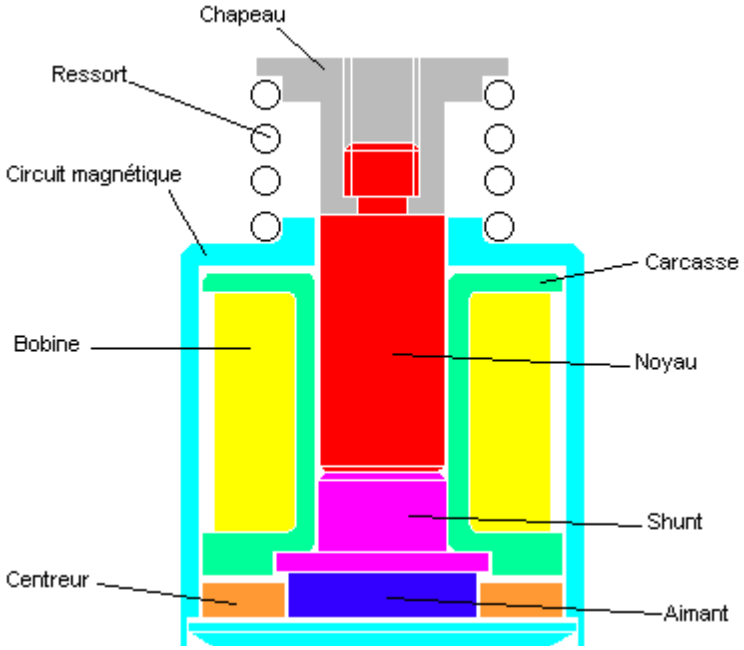


Figure 22 : Géométrie du MINIMITOP

Qualitativement, le fonctionnement du MITOP peut se décomposer en trois phases.

Phase 1 : A l'état repos, la bobine n'est pas alimentée. Le flux produit par l'aimant se répartit dans le circuit magnétique. L'énergie stockée dans l'entrefer crée une force qui compense celle du ressort et qui maintient donc le noyau collé (c'est à dire à entrefer résiduel). La différence entre la force magnétique et la force du ressort est appelée tenue au choc. Elle caractérise les chocs que peut subir l'actionneur sans qu'il ne déclenche de façon intempestive. C'est une phase statique du point de vue magnétique et mécanique.

Phase 2 : Lorsqu'un défaut est détecté sur la ligne, on alimente la bobine à l'aide d'une décharge capacitive. La bobine crée un flux qui s'oppose à celui de l'aimant. L'énergie stockée dans l'entrefer et donc la force magnétique diminuent jusqu'à ne plus compenser la force du ressort. Cette phase est statique du point de vue mécanique et dynamique du point de vue magnétique.

Phase 3 : La force du ressort n'étant plus compensée, le ressort se détend et met en mouvement le noyau qui va venir percuter le mécanisme d'ouverture.

Ce nouvel actionneur devra donc faire face à deux principales contraintes : réduction du volume et réduction de l'énergie de déclenchement.

Enfin, cet actionneur sera produit à quelques centaines de milliers d'exemplaires par an. Cela montre l'intérêt de l'optimiser pour en réduire le coût.

Les principaux points du cahier des charges qui ont été fournis sont les suivants :

Volume/encombrement	en position réarmée, environ 16x16x25 mm.

Course	5 mm (position déclenchée)
Force délivrée	10 Newtons à 2 mm
Tenue au choc	L'actionneur doit supporter une accélération de 1700 m/s^2 sans déclencher.
Energie de déclenchement	L'actionneur doit pouvoir déclencher à l'aide d'une décharge d'un condensateur d'une capacité de 22 µF chargée sous 15V.
Temps de déclenchement	Le temps de déclenchement mesuré entre le début de décharge du condensateur et une course de 3 mm, doit être inférieur à 2 ms.

Tableau 4 : Caractéristique du cahier des charges

IV.A.2.b-PRESENTATION DES RESULTATS

Les résultats trouvés par l'optimisation ont donné lieu à des prototypes. En effet une des craintes que l'on peut avoir est que les solutions proposées par l'optimiseur ne soient optimales que pour le modèle analytique, et que cela ne corresponde pas à une amélioration sensible du dispositif. La question sous-jacente est celle de la validité du modèle après optimisation. En pratique, les concepteurs deviennent de plus en plus sûrs de leur modélisation et donc sont confiants dans les résultats de l'optimiseur. En revanche, dans le cadre de ce stage, nous avons profité de la nécessité de faire des prototypes pour les comparer avant et après optimisation, et même pour les comparer à nos modélisations.

	Premier prototype (S300)	Prototype optimisé (S300)	Gain (en %)
Diamètre	18 mm	16 mm	11%
Volume	17.7 cm^3	13.6 cm^3	23%
$F_{collage}$	18.8	18.79 N	0%
$\Delta F - \Delta F_{théorique}$	3.0 N	4.1 N	37%
$I_{décrochage}$	147 AT	133 AT	10%

Tableau 5 : Comparatif des prototypes avant et après optimisation

Cette solution a permis de valider la faisabilité et une bonne idée de la valeur optimale de la solution axisymétrique telle que représentée sur le schéma du MITOP. Il a donc naturellement été demandé d'adapter l'étude à d'autres structures proches. La première adaptation a été de réaliser le même travail mais en n'utilisant pas une géométrie axisymétrique, mais une tôle pliée pour le retour de flux. Cette solution présente l'avantage d'être plus simple à réaliser, mais plus dure à modéliser. La deuxième adaptation a concerné

le calibre du disjoncteur que devra déclencher le MITOP ; les deux structures précédentes ont été dimensionnées pour un effort d'ouverture nécessaire environ deux fois plus important. Enfin la dernière étude a consisté à étendre le modèle du MITOP sur un autre déclencheur de géométrie proche, mais dont le circuit magnétique sature.

	Mitop de base	Mitop en tôle pliée	Mitop calibre au dessus	Mitop avec saturation
Durée	13 semaines avec prototype	6 semaines avec prototype	2 semaines sans prototype*	2 jours d'adaptation du modèle**

* le délai de fabrication d'un prototype est d'environ 2 semaines

** il restait à faire l'optimisation, et toutes les vérifications, estimées à 2 semaines.

Tableau 6: Durée d'études de dimensionnement pour différentes adaptations.

IV.A.2.c- CONCLUSION

L'étude qui a permis de dimensionner une première solution pour le MITOP est un résultat intéressant en soi. En effet elle permet déjà d'illustrer que la méthode proposée permet de conduire rapidement à une solution, et que cette solution est fiable, comme en atteste la validation vis à vis du prototype.

Par ailleurs cette première partie de l'étude montre la difficulté de mesurer les gains obtenus grâce à la méthode car il n'existe pas d'éléments de comparaison faciles. Les acteurs de la conception qui utiliseraient une méthode moins performante n'accepteraient pas facilement d'admettre qu'ils y ont gagné du temps. Néanmoins, lors ce stage, il a été demandé à Cédric Osmond de trouver une géométrie pour le MITOP, qui satisfasse le cahier des charges, mais aussi qui ait un certain encombrement. Dans le temps qui lui était imparti pour trouver cette géométrie à encombrement donné, il lui a fallu se familiariser avec le dispositif, la méthodologie de dimensionnement, et l'outil de dimensionnement. Il a fallu également faire le modèle approché, qui résulte de la bonne maîtrise de ce qui précède. Cédric Osmond a non seulement trouvé une solution en respectant les délais, mais il a également trouvé une solution avec un encombrement meilleur que celui qui lui était demandé (validation par prototype à l'appui).

On peut considérer qu'à la fin de cette première étape, Cédric Osmond maîtrisait son modèle, le dispositif, et la méthode. Il lui a été alors demandé un travail similaire, pour des dispositifs différents (soit par la taille, soit par la technologie), et on peut voir que la durée du cycle de conception a été fortement réduite, pour presque gagner un ordre de grandeur.

S'il était encore possible de contester les apports dus à la méthode lors de la première partie de l'étude, il n'y a plus de contestation possible dans la deuxième partie de cette étude.

Si cette expérience se révèle généralisable, on peut alors espérer que les logiciels de dimensionnements gagneront leur place dans les bureaux d'études industriels, et que la technologie de génération de code à partir de modèles analytiques sera la technologie de référence de ces outils.

IV.B Intégrations de nouvelles capacités de modélisation

Le logiciel-prototype que nous présentons a pour objet d'illustrer comment supporter certains processus. Dans ce cadre, il semble important de montrer qu'il est possible d'utiliser les extensions de langage dans une modélisation industrielle. En effet le composant gen-X a été pensé pour rendre possible l'intégration d'une capacité en génération de code de calcul, que ce soit pour de l'intégration d'un logiciel existant, ou pour de l'exploitation numérique de calcul. Dans tous les cas, le développeur de composant gen-X est une personne sensibilisée au problème de la génération de code, et de la programmation automatique (donc *a fortiori* de la programmation). En revanche l'utilisateur de Pro@DESIGN, est un ingénieur qui développe des compétences en modélisation approchée, et *a fortiori* dans son domaine physique. Il est alors impératif de tester si la complexité du composant gen-X est transparente pour l'utilisateur final.

La modélisation approchée par équations analytiques explicites, ne permet certainement pas de modéliser tous les dispositifs possibles. Si nous pensons qu'une approche plus complexe et plus précise n'est pas toujours souhaitable, que l'on peut gagner à utiliser une approche simple et rapide même si elle est imprécise, il y a des cas où les phénomènes dimensionnants ne peuvent être pris en compte sans avoir recours à une technique de modélisation plus précise.

Cette situation se présente souvent quand on cherche à modéliser un phénomène dû à la saturation de matériaux (qui requiert une modélisation avec des équations implicites), ou bien lorsque le dispositif présente un comportement transitoire critique (qui requiert cette fois des équations différentielles et des équations implicites).

C'est pourquoi nous avons développé deux composants gen-X de test, un qui traite les équations implicites, et l'autre les équations différentielles.

IV.B.1- EQUATIONS IMPLICITES

IV.B.1.a- INTRODUCTION

La modélisation fondée sur des circuits réluctants permet de calculer le champ électromagnétique en tout point du dispositif. Pour cela les lignes de flux sont considérées comme canalisées dans des tubes de flux, et chaque tube de flux peut être ramené à une réluctance moyennant une hypothèse sur la répartition du flux dans le tube même. Les tubes alors mis bout à bout forment un circuit semblable à un circuit électrique. Pour connaître la valeur du champ à l'intérieur de chaque réluctance il faut résoudre les équations du circuit (équations très semblables aux lois de Kirchoff), qui mettent en relation les Ampère-tours et le flux. Si le matériau magnétique employé dans le dispositif fonctionne en mode non saturé, alors les équations de circuit se ramènent à une dépendance multilinéaire entre les Ampère-tours et le flux, et peuvent par conséquent être résolues analytiquement. En revanche si le matériau est saturé, alors la réluctance dépend du flux qui la traverse, créant une relation de dépendance fortement non linéaire entre le flux et l'excitation.

Dans le cas linéaire :

$$nI = \Re \cdot \phi \qquad (1)$$

Dans le cas non linéaire on a en général quelque chose comme :

$$nI = \Re(\phi) \cdot \phi \qquad (2)$$

Ce sont ces deux équations qui doivent être résolues pour connaître le flux dans le dispositif. Dans le cas linéaire la résolution est symbolique :

$$\phi = \frac{nI}{\Re} \qquad (3)$$

Dans le cas non linéaire, une solution symbolique ne peut pas être trouvée sans connaître l'expression de la dépendance de la réluctance avec le flux, qui dépend elle-même de la modélisation de la courbe de saturation du matériau. Même dans les rares cas où cette solution analytique est trouvable, elle est tellement dépendante de l'expression de la courbe de saturation du fer que le modèle ainsi constitué est très difficile à maintenir. En effet si l'on souhaite effectuer le moindre changement dans les propriétés physiques du matériau, voire dans la technique de modélisation de la saturation dans les matériaux, la totalité du modèle est à revoir, et il faut procéder à nouveau à la résolution symbolique des équations de circuit. On a dit que ces équations étaient rarement solubles symboliquement, par conséquent le plus probable est que le moindre changement rende le modèle caduque.

C'est pourquoi il est souhaitable dans ce cas de résoudre les équations de circuit numériquement, et non symboliquement. De plus, pour tous les cas où elles ne sont pas solubles, cette résolution numérique est la seule voie de sortie.

Il faut donc chercher le flux :

$$\phi = x \mid \Re(x) \cdot x - nI = 0 \qquad (4)$$

Le composant gen-X de génération des équations implicites écrit par Loïg Allain, doctorant du LEG, prend tout simplement une description de cette dernière équation, mais aussi de sa généralisation de taille n.

$$\begin{array}{l} \forall i \in N, i < n \\ \begin{matrix} \phi_0 \\ \vdots \\ \phi_n \end{matrix} = \begin{matrix} x_0 \\ \vdots \\ x_n \end{matrix} \mid \begin{matrix} f_0(x_0 \cdots x_n) = 0 \\ \vdots \\ f_n(x_0 \cdots x_n) = 0 \end{matrix} \end{array} \qquad (5)$$

IV.B.1.b-MODELE

Dans le cadre d'un stage de DEA , effectué par Benoit Morel, en collaboration avec Schneider Electric, nous avons été confronté à un système implicite.

Le dispositif étudié était un composant d'un futur produit de la gamme Schneider Electric, qui devait être prochainement lancé en production. Il s'agit d'un contacteur-disjoncteur, assurant la protection et la commande de moteurs électriques, pour un courant nominal de 32 A par phase. Il permet de démarrer et d'arrêter des moteurs, tout en assurant leur protection vis à vis des courts-circuits et des surintensités. Ce produit est composé de quatre éléments principaux:

- Les contacts et leur chambre de coupure ;
- Un déclencheur électromagnétique ;
- Un électro-aimant pour la fermeture des contacts ;
- Le dispositif électronique de mesure, et de commande des deux électro-aimants.

Deux électro-aimants agissent donc sur les contacts dans ce dispositif. Cependant, tous deux ont des rôles très différents :

- Le rôle du déclencheur est de permettre une ouverture rapide des contacts, en venant percuter un mécanisme qui les retient. Il intervient donc en cas de court-circuit et remplit la fonction disjoncteur du produit.
- L'autre électro-aimant est l'objet de l'étude : alimenté, il a pour rôle de permettre la fermeture des contacts. Non alimenté, il doit les ouvrir même lentement. En régime normal, cet électro-aimant est donc alimenté, il assure la fonction alimentation du produit.

Le rôle de l'électro-aimant est donc de fermer les contacts lorsqu'on le commande. Inversement, pour des raisons de sécurité évidentes, il doit les ouvrir en l'absence de commande.

L'électro-aimant et l'ensemble de son mécanisme doivent donc être monostables en l'absence d'alimentation, la position stable (position de repos) étant la position contacts ouverts. A l'inverse, lorsqu'il est alimenté, l'électro-aimant se retrouve dans la position contacts fermés, stable avec une alimentation normale.

Le retour de la position de travail (contacts fermés) à la position de repos (contacts ouverts) ne fait intervenir que des phénomènes mécaniques, car ce retour se fait normalement alimentation coupée. Par contre, la fermeture de l'électro-aimant, de la position de repos à la position de travail, se fait lors de la mise sous tension de sa bobine et fait intervenir des phénomènes électromagnétiques.

La fermeture de l'électro-aimant se déroule en quatre étapes :

- Dans l'état initial, le courant dans la bobine est nul. L'électro-aimant est maintenu dans sa position de repos par la force de collage créée par l'aimant seul.
- On alimente la bobine en tension. On a alors une montée du courant classique lorsque l'on met une inductance sous tension. Le mobile reste en position de collage.
- Le courant atteint une valeur suffisante pour annuler la force de collage, le mobile décolle et prend de la vitesse. La variation rapide d'inductance faisant diminuer le courant, on a alors une évolution rapide du courant et de la position.
- Le mobile arrive en bout de course. Il y est maintenu par la force créée par le courant circulant dans la bobine. Il n'y a plus de mouvement, le courant reprend sa montée jusqu'à atteindre le régime permanent.

Pris seul, l'électro-aimant est bistable, il est donc nécessaire de lui adjoindre un ressort de rappel, poussant le mobile à l'ouverture, pour rendre l'ensemble monostable en l'absence de courant. Par ailleurs, le maintien des contacts en position fermée est assuré par d'autres ressorts, les ressorts de pôle, qui poussent à la fermeture.

Enfin, le mobile n'est pas fixé mécaniquement aux contacts, ils ne sont en contact qu'entre la position de repos et la position de fermeture des contacts.

La force appliquée sur le mobile par les ressorts de pôle s'oppose donc à celle du ressort de rappel, mais seulement entre la position de repos et la fermeture des contacts.

La modélisation par schéma réluctant d’un tel dispositif est une modélisation usuelle, à ceci près que le point de fonctionnement du dispositif est tel que de nombreuses parties du fer sont saturées. La modélisation à base de schéma réluctant, lorsque des réluctances sont saturables, pose des problèmes. Il faut tenir compte de la courbe de saturation du matériau et

adapter la réluctance au niveau du flux qui la traverse. Ceci induit comme on l'a vu des équations implicites. Mais cela pose un autre problème délicat : lorsque la valeur d'une réluctance varie suite à la saturation progressive du matériau, les réluctances de fuite prennent une importance de plus en plus grande, et leur modélisation doit être précise.

IV.B.1.c- RESULTAT

Les difficultés de modélisation n'ont pas permis de conduire à une optimisation, et à un pré-dimensionnement complet du dispositif. Contrairement aux autres DEA qui avaient conduit à un dimensionnement, l'objectif de ce stage était plus orienté modélisation. En effet la capacité à pré-dimensionner une solution qui est correctement modélisée analytiquement ne faisait pas l'objet de cette étude, mais plutôt la capacité à tenir compte de modèles fortement implicites dans l'outil. Pour cela il fallait mettre au point le modèle fortement saturé. Par ailleurs une partie importante du stage a été consacrée à la modélisation de la dynamique du système dans une version recherche de flux. Dans le temps restant le modèle a atteint une précision intéressante.

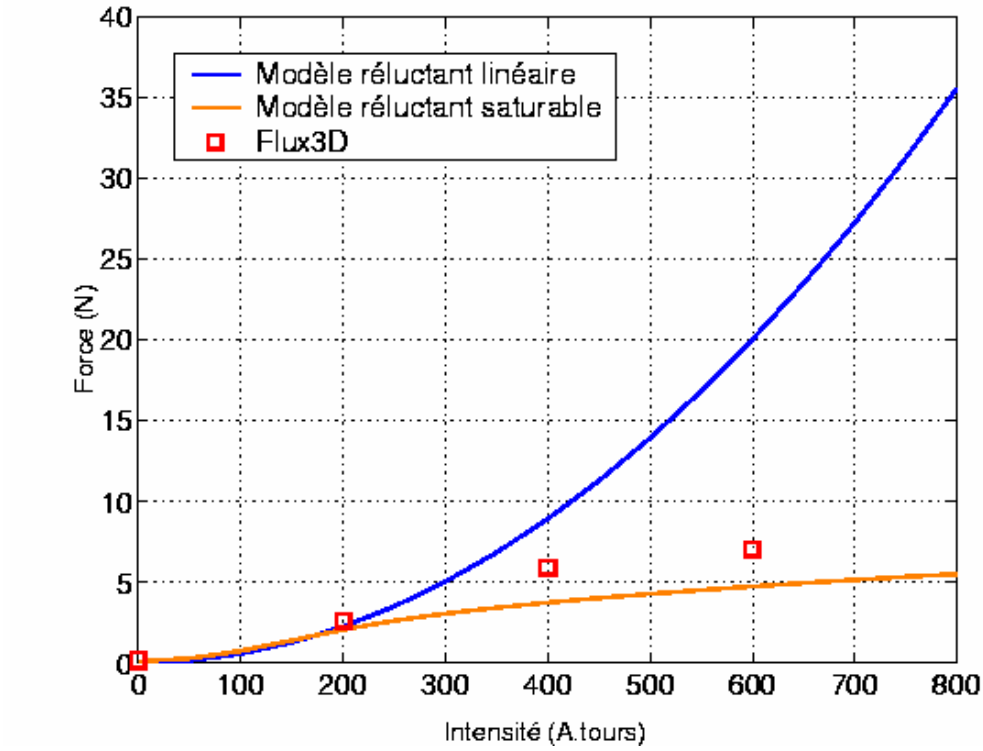


Figure 23 Courbe de force simulée par différentes méthodes pour une position donnée du noyau.

On constate que le modèle réluctant avec prise en compte de la saturation donne une précision uniforme satisfaisante.

L'effort de modélisation nécessaire à la prise en compte de saturations dans plusieurs parties du dispositif, est tel que le stage de DEA ne suffit pas à conduire l'étude jusqu'à une optimisation et validation par un prototype. Le modèle obtenu est un modèle implicite de taille importante (la taille du modèle est égale au nombre de réluctances susceptibles de saturer). Nous avons quand même effectué une optimisation du modèle sur un cahier des charges qui nous semble industriel. Cette optimisation a convergé vers une solution. Cette partie du travail n'ayant pas été suivie par une personne du service dans lequel s'est effectué le stage, nous ne pouvons pas considérer qu'il s'agit d'un résultat industriel.

Le résultat majeur que l'on peut dégager de cette étude est que la modélisation approchée de dispositif tridimensionnel, et fortement saturé, est certes plus difficile à mener, mais néanmoins réalisable avec une précision uniforme acceptable.

IV.B.2- EQUATIONS DIFFERENTIELLES

A l'instar des équations implicites, les équations différentielles se situent à mi-chemin entre le tout analytique, et le tout numérique. On a en effet d'une part un problème physique, qui s'exprime à travers des équations différentielles (prise en compte d'un mouvement, d'une pénétration de champ dans un ferromagnétique), et d'autre part un besoin en calcul qui exploite ce résultat. Si l'on essaie de faire une topologie des équations différentielles, on se rend compte que bien peu d'entre elles sont solubles analytiquement. Physiquement, on peut estimer qu'en dehors de régimes permanents sinusoïdaux, on utilise peu souvent une résolution analytique d'équation différentielle. Néanmoins ces équations interviennent souvent, par exemple dans presque tous les calculs de phénomènes transitoires.

Le problème peut sembler simple, mais pour prendre en compte les équations différentielles, il faut introduire dans la modélisation analytique fondée sur des équations et le calcul de paramètres, un espace plus complexe, la modélisation fondée sur les fonctionnelles, et le calcul sur les fonctions.

IV.B.2.a- MODELE : UN CAS SIMPLE

La complexité de la prise en compte des équations différentielles dans un outil de dimensionnement nous a conduit à ne disposer de cette fonctionnalité que très tard. Par conséquent, nous n'avons pas eu l'occasion de tester la fonctionnalité sur un cas industriel. C'est pourquoi nous allons présenter un cas simple, mais symbolique d'une utilisation des équations différentielles en conception de dispositif électromécanique.

La portée d'un canon est un modèle qui représente bien les modèles dynamiques des électro-aimants. Une des grandeurs clefs de calcul d'un déclencheur électromagnétique est le temps de fermeture. Celui-ci s'obtient en calculant la trajectoire de la partie mobile, et en résolvant l'instant où cette partie mobile atteint la butée. Le calcul de la trajectoire fait appel à une équation différentielle, tandis que la résolution de l'instant de fermeture fait appel à une fonctionnelle simple, la racine de la fonction.

La portée de canon représente bien cette typologie de modèle car le projectile est soumis à une équation différentielle simple (que l'on peut complexifier), et l'exploitation qui permet de mesurer la portée se fait en résolvant d'abord le temps de vol.

Soit un projectile P de masse m, soumis à une condition initiale de vitesse de module Vo et d'angle α par rapport à l'horizontale. En l'absence de frottement f on sait que la portée est maximale pour un angle α de 45°.

Le projectile est soumis à l'équation différentielle suivante, issue du principe fondamental de la dynamique :

$$\begin{cases} \frac{d^2x}{dt^2} = -\frac{f}{m} * \frac{dx}{dt} \\ \frac{d^2y}{dt^2} = g + \frac{f}{m} * \frac{dy}{dt} \end{cases}$$

Équation 35: Système différentiel d'un projectile

Dans un contacteur on a sensiblement le même système différentiel, auquel il faut ajouter les équations électriques. Les solutions de ce système x(t) et y(t) sont soumises à une extraction de racine pour trouver le temps de vol.

$$t_{vol}=t\,|\,y(t)=0$$

Équation 36: Equation implicite du calcul de temps de vol

On voit bien que l'équation implicite précédente a deux solutions, dont une triviale à t=0. C'est pourquoi il est important de pouvoir choisir parmi les racines d'une fonction, celle qui nous intéresse.

On en déduit simplement la portée du canon en calculant la distance parcourue pendant ce temps.

$$portée=x(t_{vol})$$

Équation 37 : Calcul de la portée

Le modèle est très simple, et il est très simple à écrire sous forme d'équation.

```
//fonctions de l'équation
ypp(yp)=-g-f/m*yp;
xpp(xp)=-f/m*xp;

//condition initiales de l'Equation
vox=vo*cos(alpha);
voy=vo*sin(alpha);

@ode
//resolution de l'équation
[y   ,yp   ,x    ,xp     ]  // fonctions solutions
     =ODE(t,
[Id(yp),ypp(yp),Id(xp),xpp(xp)], // fonctions de l'équation
diff (y'=f(y,t) )
[0   ,voy  ,0    ,vox    ]); // conditions initiales

@functional
// resolution de y(t)=0
tsol=root(y(u),u=a);
@eden

//utilisation des resultats,
porte=x(tsol);
```

Figure 24 : Modèle saisi dans le Logiciel Pro@DESIGN

IV.B.2.b-RESULTATS

Les résultats n'ont rien de remarquable. Si l'on pose comme coefficient de frottement f=0, alors l'angle qui maximise la portée, est bien 45°. Si l'on adopte d'autres valeurs pour ce coefficient de frottement, l'optimum n'est plus de 45°. On peut même exploiter ce modèle pour tracer la portée en fonction de l'angle de tir.

Même un modèle aussi simple que la portée de canon, pose des problèmes que l'on n'avait pas envisagé au premier abord : la nécessité de disposer des fonctions implicites et plus généralement d'autre fonctionnelles pour pouvoir exploiter le résultat d'une équation différentielle.

Un des problèmes qui apparaît avec la génération des équations différentielles est le calcul des dérivées, et plus fondamentalement la gestion de la mémoire de l'ordinateur, pour pouvoir réaliser un compromis entre temps de simulation et mémoire occupée. En effet, lorsque l'on exploite une fonction issue de la résolution d'une équation différentielle, on est confronté au dilemme suivant : soit on resimule à chaque appel de la fonction, soit on mémorise tous les points simulés et on se contente de renvoyer ces points à chaque appel. Bien entendu la réalité est un compromis entre ces deux stratégies, mais nous avons vu lors de nos premiers tests sur la portée de canon qu'une implémentation simpliste de la stratégie pouvait nécessiter des quantités de mémoire très supérieures à celles disponibles sur une machine classique.

CONCLUSION

L‘équipe de recherche dans laquelle ces travaux ont été effectués s’est construite, il y a une dizaine d’années, avec pour objectifs l’automatisation et le développement de la conception intégrée. En dix ans de recherche une technologie et un paradigme complet ont émergé. Les travaux sur les systèmes à base de connaissance ont abouti à définir un formalisme qui réalise un bon compromis entre l’efficacité et la généricité. Il s’agit des équations mathématiques, et il permet trois axes fondamentaux.

D’une part les connaissances mathématiquement formalisées existent depuis longtemps, dans un état exceptionnellement stable. Par exemple des travaux datant de 1941 [Roters-41] ont été récupérés par l’équipe [Gentilhomme-91, Trichon-91]. Quel système expert peut se prévaloir de récupérer en l’état des règles d’expertise écrites 50 ans plus tôt ?

D’autre part cette formalisation mathématique des connaissances est effectuée par des experts métier. Elle subit ensuite automatiquement toute une série de transformations symboliques, jusqu’aux instructions machine. Ces transformations sont indépendantes du contenu, et par là même bénéficient de tout progrès dans leur traitement (compilateurs, calcul numérique, « loi » de Moore). Notre technologie fait le lien entre l’expert en dimensionnement et les formalismes informatiques.

Ces dernières décennies ont vu l’essor de méthodes automatiques de calcul symbolique. Le traitement formel d’un modèle en équations peut donner bien plus que le simple calcul de résolution (par exemple dérivation, intégration, dépendances, simplification, factorisation, résolution d’équations, extraction de racines…). Notre technologie utilise déjà le calcul, la dérivation et la détection des dépendances.

Le prototype que nous avons implémenté pour valider la méthodologie fonctionne. La méthodologie est facilement appréhendée par des stagiaires ingénieurs qui ne connaissent pas le dimensionnement. Elle l’est également par des experts de leur domaine qui se familiarisent vite avec la notion de COB.

De plus, notre architecture composant a supporté la montée en charge. Nous avons ajouté des fonctionnalités telles que les équations différentielles, ou les équations implicites, nécessitant la génération d’algorithmes numériques. Elle a supporté jusqu’à quatre algorithmes d’optimisation différents. L’architecture a montré sa robustesse face aux évolutions des besoins, et sa simplicité face à l’utilisation par des experts, ou des gens étrangers à l’implémentation.

La technologie à base de composants logiciels que nous avons mise au point doit maintenant s’insérer dans un produit utilisé par les services de conception. Elle est suffisamment mûre pour sortir d’un laboratoire, et s’engager sur la voie d’une industrialisation. En effet elle est implémentée dans les technologies logicielles les plus récentes (java, XML). De plus les expériences d’utilisation menées avec des stagiaires ingénieurs et des chercheurs universitaires, ont contribuées à la valider.

Enfin, il ne reste plus alors qu’à la décliner en un produit industriel destiné aux services de conception..

A partir de la technologie composant mise au point dans cette thèse, il semble que deux axes d'améliorations se dessinent.

D'une part, le calcul numérique est une science qui fait partie aujourd'hui de l'enseignement de base d'un ingénieur, et par conséquent de ses compétences élémentaires. Or, il y a dans ce domaine des familles de problèmes difficiles à résoudre ; le marché offre plusieurs algorithmes permettant de résoudre le même problème (c'est le cas notamment de l'optimisation sous contraintes). Chacun ayant ses spécificités, il appartiendra alors à l'utilisateur de choisir judicieusement l'algorithme approprié à son besoin.

La perspective d'évolution peut être de parvenir à définir des algorithmes comme des composants.

D'autre part il semble important de surveiller les progrès en calcul formel pour pouvoir les intégrer dans une méthodologie de dimensionnement. On peut d'ores et déjà envisager de détecter les linéarités des modèles, ou bien calculer les unités des paramètres de sortie en fonction des unités des paramètres d'entrée. Il reste néanmoins d'autres traitements symboliques connus (factorisation, simplification, résolution, intégration …) dont on n'a pas étudié l'application au dimensionnement.

ANNEXES

V- Etude de cas : Dispositif de déclenchement

V.A Introduction

L'objectif était d'étudier un dispositif industriel en appliquant la méthodologie de dimensionnement que nous proposons. Ce cas permet d'illustrer :

- la démarche de conception et d'optimisation,
- la nécessité d'une approche dynamique (équations différentielles) par rapport à une approche statique,
- l'intérêt du logiciel PRO@DESIGN qui supporte l'approche dynamique, et qui fournit des outils pour celle-ci.

Cette étude a été menée dans le cadre d'un contrat entre la société Schneider et le LEG, au cours de laquelle j'ai mené l'étude en modélisation du dispositif, puis une optimisation « statique » du dispositif.

Une seconde partie a été menée sur la fin de ma thèse, car nous disposions dans Pro@DESIGN du support des équations différentielles. Elle a été faite par Regis Darnault, qui a écrit les équations différentielles et résolu les problèmes soulevés par celle-ci.

C'est la synthèse de ces deux études qui est montrée ici.

V.B Présentation du dispositif et de l'objectif de dimensionnement

V.B.1- DESCRIPTION DU DECLENCHEUR

Le dispositif est un contacteur. Il est constitué d'une bobine et d'un circuit magnétique lui même constitué d'une partie mobile et d'une partie fixe, comme indiqué sur le schéma.

Le circuit magnétique est constitué de fer homogène dont on connaît la perméabilité relative, il est fait de deux entrefers : celui à droite sur le schéma qui est l'entrefer variable (celui qui permet la mise en mouvement) et d'un entrefer fixe (en pointillé sur le schéma). Le noyau du dispositif referme le déclencheur quand la bobine est alimentée.

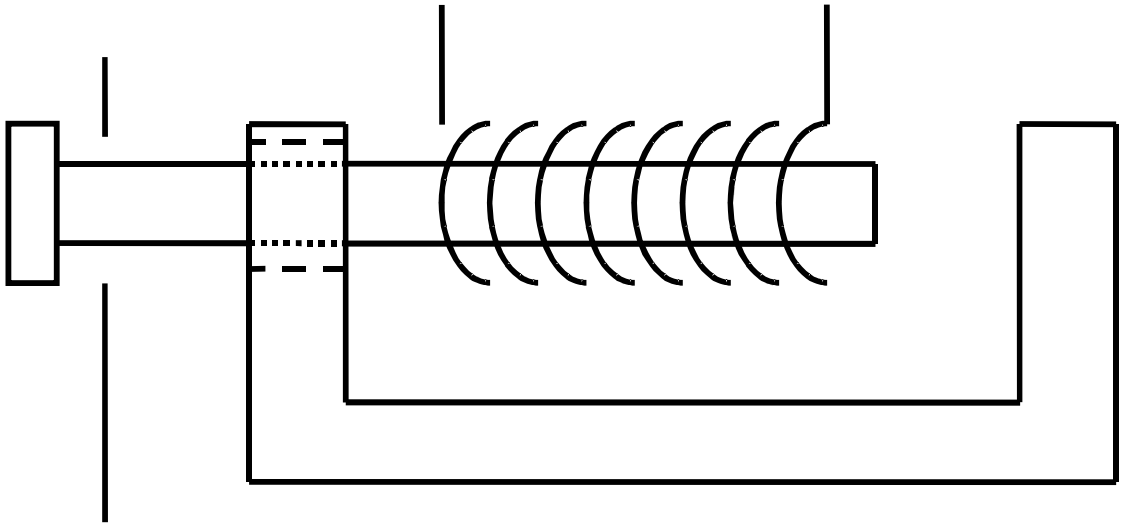


Figure 25 : Schéma du déclencheur étudié.

La bobine est construite en deux galettes. Elle est alimentée par un circuit électrique comme indiqué sur le schéma.

Le thyristor pouvant être commandé à chaque instant on ne peut prédire la forme du signal qui alimentera la bobine. On considère que la résistance propre de la bobine n'est pas négligeable, et il faut donc la rajouter au circuit électrique équivalent.

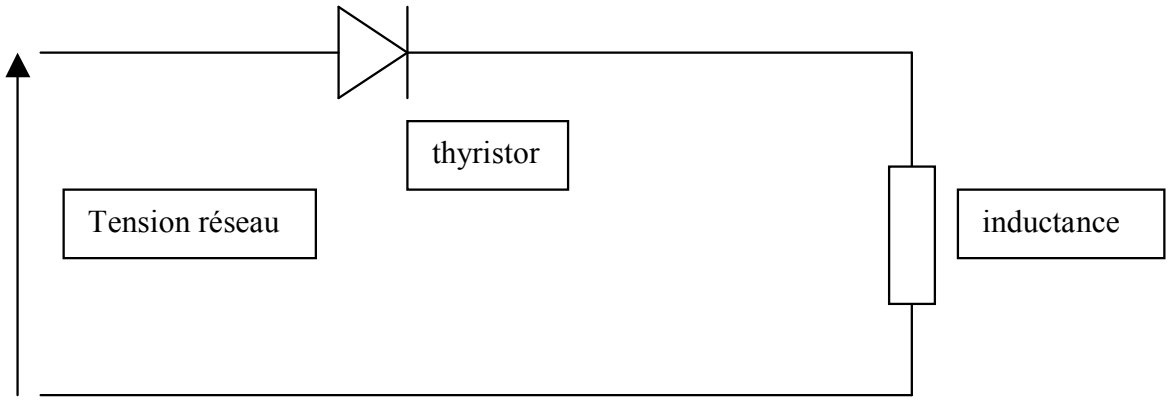


Figure 26 : Schéma Electrique d'alimentation du dispositif

V.B.2- OBJECTIFS DE DIMENSIONNEMENT ET CONTRAINTES

L'objectif est de minimiser l'encombrement compte tenu des contraintes suivantes :

- Le dispositif ne doit en aucun cas se détruire.
- fonctionner à n'importe quelle tension entre 50 et 400V.
- fermer en moins de 30ms
- Résister à la foudre.

Remarque : Pour des raisons de normes, il faut que le déclencheur ferme en moins de 30 ms. Mais l'alimentation du déclencheur supprimant l'alternance négative (diode), le

fonctionnement du déclencheur peut n'avoir lieu que sur 10ms (dans le pire des cas). Il faut donc que le déclencheur soit fermé en moins de 10ms.

V.B.3- EQUATIONS PHYSIQUES

La physique du phénomène en jeu s'exprime dans notre cas assez simplement par deux équations différentielles à coefficients non constants :

$$\begin{cases} U(t) = R*i(t) + \dfrac{d}{dt}(L(i,z)*i(t)) \\ m\dfrac{d^2z}{dt^2} = F(i,z) \end{cases}$$

Avec : i, le courant dans la spire.

z, la longueur de l'entrefer

R, la résistance de la bobine

L, l'inductance de la bobine

m, la masse du noyau.

F, la force globale exercée sur le noyau (force magnétique plus charge, plus frottement…)

La première équation représente la loi de comportement électrique, la seconde la loi de comportement mécanique. En toute rigueur il manque la loi de comportement magnétique. Celle ci est résolue lorsque l'on est capable de calculer l'inductance et la force en fonction du courant dans la spire, et de la longueur de l'entrefer. Cela revient à faire l'hypothèse que les phénomènes magnétiques sont rapides devant les constantes de temps mécanique et électrique.

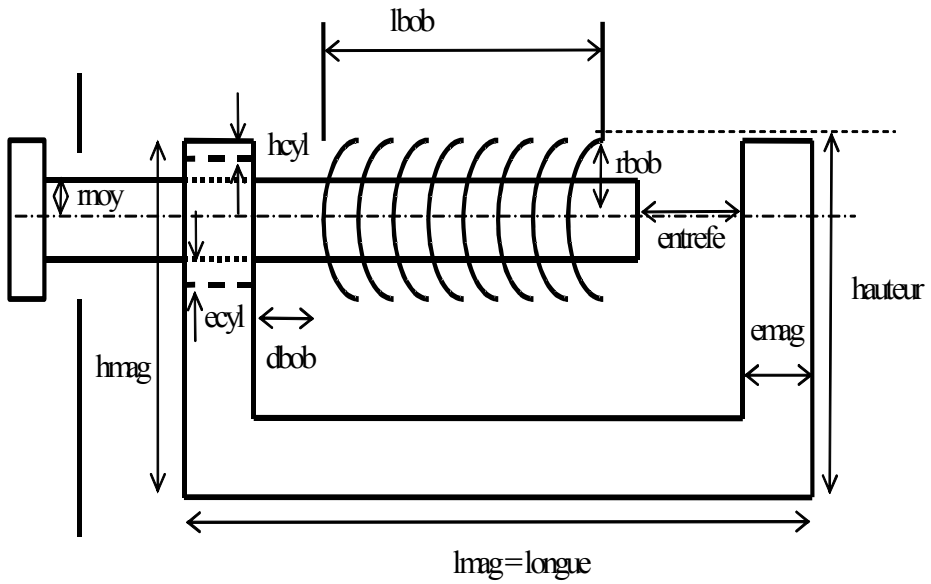
V.B.4- DEFINITIONS

Nous définissons les paramètres géométriques, électriques et magnétiques suivants.

Paramètres géométriques de définitions :

Nom	Définition	Unité
Entrefe	Valeur de l'entrefer initial coté course.	m
Hmag	Hauteur globale du circuit magnétique.	m
Lmag	Longueur du circuit magnétique.	m

Pmag	Profondeur du circuit magnétique.	m
Emag	Epaisseur du circuit magnétique.	m
Rnoy	Rayon du noyau.	m
Ecyl	Entrefer fixe (situé dans la partie cylindrique).	m
Hcyl	Hauteur partie circuit magnétique située au-dessus trou de guidage du noyau.	m
Dbob	Distance entre le début de la bobine et la fin du circuit magnétique coté fer.	m
Lbob	Longueur de la bobine.	m
Rbob	Rayon de la bobine.	m
N	Nombre de spire de la bobine.	-
Dcond	Diamètre des conducteurs.	m



Paramètres géométriques qui seront calculés pour ajouter des contraintes géométriques :

Nom	Définition	Unité
Longue	Longueur totale du dispositif	m
Largeur	Largeur totale du dispositif	m
hauteur	Hauteur totale du dispositif	m
Volume	Volume total du dispositif	m^3
Seff	Section efficace du Cuivre	m^2
Stot	Section totale disponible pour le cuivre	m^2
Cun	Espace entre bobine et circuit magnétique dans axe horizontal coté entrefer mobile (sera utilisé en tant que contrainte géométrique n°1)	m
Cdeux	Espace entre bobine et circuit magnétique dans axe vertical (sera utilisé en tant que contrainte géométrique n°2)	m
Ctrois	Différence entre section efficace cuivre et section disponible (sera utilisé en tant que contrainte géométrique n°3)	m^2

Paramètres électriques de définition :

Nom	Définition	Unité
Eeffmax	Tension efficace maximale d'utilisation	V
Eeffmin	Tension efficace minimale d'utilisation	V
Efoudre	Tension de foudre	V
Rho	Résistivité du cuivre.	Ω.m
Frequence	Fréquence	Hz

Paramètres électriques qui seront calculés :

Nom	Définition	Unité
ibobstat	Courant max dans la bobine à eeffmin	A
inbob	Inductance de la bobine.	H
rebob	Résistance de la bobine.	Ω

imax	Courant maximum, donc à eeffmax	A
dens	Densité de courant correspondante à imax	A/m²
vmax	Tension maximale entre 2 spires adjacentes quand bobine soumise à tension de foudre	V

Paramètres mécaniques de définition :

Nom	Définition	Unité
mfer	Masse volumique du fer.	kg/m³
freac	Force qu'exerce la charge sur le noyau.	N

Paramètres mécaniques qui seront calculés :

Nom	Définition	Unité
masse	masse du noyau	Kg
forcedecol	Force de décollage : force magnétique appliquée lorsque la tension est minimum (eeffmin) et l'entrefer est ouvert moins la force qu'exerce la charge sur le noyau (freac).	N

Paramètres magnétiques de définition :

Nom	Définition	Unité
muz	Perméabilité du vide.	T.m/A
murel	Perméabilité relative du fer.	-
Bsat	Induction de saturation.	T
taux	Coefficient d'approximation des fuites dans l'entrefer principal (1 = aucune fuite, 1.2 = 20% de fuites).	-
refui	Reluctance de fuite.	A/Wb

Paramètres magnétiques qui seront calculés :

Nom	Définition	Unité

mufer	Perméabilité du fer (= murel x muz).	T.m/A
muequ	Permittivité équivalente du système en dynamique (= mufer en modélisation statique)	-
renoy	Reluctance noyau.	A/Wb
recyl	Reluctance entrefer cylindrique.	A/Wb
remaga	Reluctance partie verticale du coté entrefer fixe du circuit magnétique.	A/Wb
remagb	Reluctance partie horizontale du circuit magnétique.	A/Wb
remagc	Reluctance partie verticale cote entrefer mobile du circuit magnétique.	A/Wb
reent	Reluctance entrefer mobile pour position ouverte entrefer.	A/Wb
refer	Reluctance globale du fer.	A/Wb
reequ	Reluctance équivalente de tout le circuit.	A/Wb
phi	Flux magnétique dans circuit magnétique	Wb
force	Force magnétique	N
bnoystat	Induction dans la modélisation statique	T

Il faut aussi définir le paramètre tferm (en s) qui correspond au temps de fermeture du dispositif.

Remarque liée à l'utilisation du logiciel Pro@DESIGN : certains des paramètres ci-dessous sont en réalité des fonctions dépendant du courant et de l'entrefer (valeurs figées lors de l'étude en statique). Elles ne sont donc pas visibles parmi les paramètres de sortie.

Pour pouvoir visualiser des résultats à z et i donné ou encore tracer des courbes en fonction de ces paramètres, nous avons donc introduit les paramètres calc_i et calc_z en entrée auxquels font appel les équations suivantes :

calc_muequ = muequ(calc_i,calc_z);
calc_renoy = renoy(calc_i,calc_z);
calc_recyl = recyl(calc_i,calc_z);
calc_remaga = remaga(calc_i,calc_z);
calc_remagb = remagb(calc_i,calc_z);
calc_remagc = remagc(calc_i,calc_z);
calc_reent = reent(calc_z);

calc_refer= refer(calc_i,calc_z);

calc_reequ = reequ(calc_i,calc_z);

calc_inbob = inbob(calc_i,calc_z);

calc_phi = phi(calc_i,calc_z);

calc_force = force(calc_i,calc_z);

Pour tracer des courbes en fonction du temps dans la modélisation dynamique, nous avons également décidé de faire entièrement appel au temps et donc avons défini :

calc_z = z(t);

calc_i = i(t);

calc_v = v(t); (v est la vitesse de la masselotte, la dérivée de z)

Ces paramètres ne jouent en rien sur le processus d'optimisation.

Convention typographique : dans les chapitres suivants, les équations saisies dans Pro@DESIGN ont été écrites en italique.

V.C Modélisation statique

Les équations qui suivent dépendent du courant (i) et de la position de l'entrefer (z). Dans le cas d'une approche statique, le courant n'intervient plus (on se place au courant max dans le cas de tension le plus défavorable effmin) et la valeur de l'entrefer est considérée comme égale à sa valeur initiale.

Cependant, afin de garder un seul fichier d'équation entre statique et dynamique, nous les avons écrites en tant que fonction tout en figeant la valeur de z (z = *entrefe)*. Le courant, quant à lui n'apparaît plus, il n'est donc plus nécessaire de figer sa valeur.

V.C.1- EQUATIONS PHYSIQUES

V.C.1.a- COURANT MAXIMUM DANS LA BOBINE

Il s'agit du courant maximum dans la bobine dans les conditions de tensions les plus défavorables (pour l'ouverture), donc à eeffmin (50V dans notre exemple).

Nous supposons l'inductance négligeable devant la résistance. Nous obtenons donc :

*ibobstat = effmin * sqrt(2) / rebob*

V.C.1.b- PERMEABILITE

On écrit directement l'équation :

*mufer = murel * muz*

On définit la perméabilité équivalente du système (égale à mufer en statique) :

muequ(i,z) = mufer

V.C.1.c- RÉLUCTANCES

Pour plus de clarté, nous rappelons la fonction de calcul de la réluctance :

$$\mathrm{Re}(\mu, L0, S) = \frac{L0}{\mu S}$$

ou L0 est la longueur de cheminement et S sa section.

V.C.1.i- RELUCTANCE DU NOYAU AVANT LE TROU

$$\mathrm{Renoy}(i,z) = \mathrm{Re}(\mu_{equ}(i,z),\ \mathrm{lmag} - 2\ \mathrm{emag} - z,\ \pi\ \mathrm{rnoy}^2)$$

Ce qui donne dans Pro@DESIGN :

*renoy(i,z) = (lmag - 2 * emag - z) / (muequ(i,z) * Math.PI * pow(rnoy,2))*

V.C.1.ii- RELUCTANCE DE L'ENTREFER CYCLIQUE

Hypothèse : l'essentiel des lignes de flux sont dans le demi cylindre inférieur

recyl = Re(μ0, ecyl, π rnoy emag)

Ce qui donne dans Pro@DESIGN :

*recyl(i,z) = ecyl / (muz * Math.PI * rnoy * emag);*

V.C.1.iii- RELUCTANCE DE LA 1ERE PARTIE VERTICALE DE L'ARMATURE

remaga(i,z) = Re(μ(i,z),hmag – hcyl – 2 ecyl – 2 rnoy, emag pmag)

Ce qui donne dans Pro@DESIGN :

*remaga(i,z) = (hmag - hcyl - 2 * ecyl - 2 * rnoy) / (muequ(i,z) * emag * pmag);*

V.C.1.iv- RELUCTANCE DE LA PARTIE HORIZONTALE DU CIRCUIT MAGNETIQUE

remagb(i,z) = R(μ(i,z), lmag – 2 emag, emag, pmag)

Ce qui donne dans Pro@DESIGN :

*remagb(i,z) = (lmag - 2 * emag) / (muequ(i,z) * emag * pmag);*

V.C.1.v- RELUCTANCE DE LA 2EME PARTIE VERTICALE DU CIRCUIT MAGNETIQUE

Hypothèse très simplificatrice : la réluctance est en forme de U légèrement plus large au niveau de l'entrefer afin de prendre en compte les fuites. Le trajet moyen est donc un quart de Oblong.

Pour simplifier encore le calcul on considère que l'on a une pyramide comme tube de flux.

Le taux de fuite est le facteur d'évasement du champ dans l'entrefer. "Pas de fuite" est équivalent à "taux=1". Dans notre cas nous travaillerons avec taux = 1,2.

Pour calculer le chemin moyen du flux dans un virage :

s1 est le rayon de départ, s2 est le rayon d'arrivée, et α est l'angle du virage

On fait l'hypothèse d'un cercle continûment déformé.

$$Chem(s1, s2, \alpha) = \frac{(5s1 - 2s2)\alpha}{6}$$

Réluctance d'un cône de base rectangulaire :

h1,p1 : dimension de la première face

h2,p2 : dimension de la seconde face

Long : longueur du cône

μ : permittivité du matériau

$$Rc(h1, p1, h2, p2, Long, \mu) = Long \frac{\ln\left(\frac{h2\,p1}{p2\,h1}\right)}{\mu(p1h2 - h1p2)}$$

Longueur approché du trajet moyen d'une ligne de flux.

Lon = Chem(rnoy taux, emag, π/2) + (hmag – hcyl – 2 rnoy taux –emag) + Chem(emag, emag, π/2)

Nous obtenons donc :

remagc(i,z) = Rc(emag, pmag, 2 rnoy taux, 2 rnoy taux, Lon, μ(i,z))

Ce qui donne dans Pro@DESIGN :

```
remagc(i,z)     = (
                  (5 * rnoy * taux - 2 * emag) * Math.PI / 2 / 6
                  + hmag - hcyl - 2 * rnoy * taux - emag
                  + (5 * emag - 2 * emag)  * Math.PI / 2 / 6
                )
                * log(2 * rnoy * taux * pmag / (2 * rnoy * taux * emag))
                / (muequ(i,z) * (2 * rnoy * taux * pmag - 2 * rnoy * taux * emag));
```

V.C.1.vi- RELUCTANCE DE L'ENTREFER.

On considère le tube de flux comme un cône.

$$\mathrm{Re}\,nt(z) = \frac{z}{\mu 0 \pi 4 rnoy^2 taux}$$

Ce qui donne dans Pro@DESIGN :

*reent(z) = z / (muz * Math.PI * 4 * pow(rnoy,2) * taux);*

V.C.1.vii- RELUCTANCE DU FER

refer(i,z) = rnoy(i,z) + rcyl + rmaga(i,z) + rmagb(i,z) + rmagc(i,z) + reent(z)

Ce qui donne dans Pro@DESIGN :

refer(i,z) = renoy(i,z) + recyl(i,z) + remaga(i,z) + remagb(i,z) + remagc(i,z) + reent(z);

V.C.1.viii- RELUCTANCE GLOBALE

La prise en compte de l'inductance de fuite n'est pas évidente. Nous proposons (mais cela reste à valider) que la réluctance de fuite ait pour valeur la valeur de la réluctance propre d'une bobine dans le vide. Logiquement, quand il y a peu de courant l'essentiel du flux passe par les réluctances fer (donc Rfuite>Rfer(1 A)) quand le fer est totalement saturé, l'essentiel des lignes de flux passe par la réluctance de fuite (donc Rfuite < Rfer(infini A))

Cette méthode reste à valider.

Rfuite = 10 10^8 A/Wb

Donc la réluctance globale équivalente vaut :

Requ(i,z) = p(Rfer(i,z), Rfuite)

Nous obtenons la définition des fonctions suivantes :

$$reequ(i,z) = \frac{refer(i,z)\,refui}{refer(i,z) + refui}$$

Ce qui donne dans Pro@DESIGN :

*reequ(i,z) = refer(i,z) * refui / (refer(i,z) + refui);*

V.C.1.d- INDUCTANCE DE LA BOBINE

$$inbob(i,z) = \frac{n^2}{reequ(i,z)}$$

Ce qui donne dans Pro@DESIGN :

inbob(i,z) = pow(n,2) / reequ(i,z);

V.C.1.e- RESISTANCE DE LA BOBINE

$$rebob = \frac{\rho 8n \frac{rnoy + rbob}{2}}{dcond^2}$$

Ce qui donne dans Pro@DESIGN :

*rebob = 8 * rho * n * (rnoy + rbob)/(2 * pow(dcond,2));*

V.C.1.f- FORCE EXERCEE PAR LE NOYAU

Cela passe par le calcul du flux : il s'agit du flux total dans l'entrefer, c'est à dire pour nous, dans la réluctance fer.

$$\Phi(i,z) = \frac{ni}{refer(i,z)}$$

$$Force(i,z) = \frac{(\Phi(i,z))^2}{2\mu 0\pi rnoy^2}$$

Ce qui donne dans Pro@DESIGN :

*phi(i,z) = n * i / refer(i,z);*

*force(i,z) = pow(phi(i,z),2) / (2 * muz * Math.PI * pow(rnoy,2));*

Nous introduisons la force de décollage qui est la force magnétique moins la force exercée par la charge. Nous nous plaçons à courant max dans le cas de tension le plus défavorable (ibobstat). Cette force n'est pas rigoureusement la force de décollage, c'est la force qui contribuerait au décollage à l'instant où le courant serait maximum si le dispositif n'avait toujours pas décollé. Si cette force est négative, on peut affirmer que le noyau ne décollera jamais. Si cette force est trop faible, le noyau décolle à l'instant où le courant est maximum, puis se referme car la force magnétique redescendrait en dessous de la force

mécanique. Pour garantir une fermeture il faut imposer la force de décollage à être supérieure à une valeur minimum qu'il est difficile d'estimer.

Ce qui donne dans Pro@DESIGN :

Forcedecol = force(ibobstat, entrefe) - freac;

V.C.1.g- MASSE DU NOYAU

Masse = mfer π rnoy² lmag

Nous obtenons donc la définition des fonctions suivantes :

*masse = mfer * Math.PI * pow(rnoy,2) * lmag;*

V.C.2-EQUATIONS PERMETTANT D'EXPRIMER DES CONTRAINTES

V.C.2.a- CONTRAINTES GEOMETRIQUES

Calcul de la section efficace de cuivre, c'est à dire le nombre de spires multiplié par la section d'une spire.

$$\text{seff} = \frac{\text{n} \cdot \pi \cdot dcond^2}{4}$$

Calcul de la section dont dispose la bobine c'est à dire en fonction des paramètres géométriques :

$$\text{stot} = (\text{rbob} - \text{rnoy}) \cdot \text{lbob}$$

Calcule d'une première contrainte géométrique qui doit être positive. Elle représente la distance entre la bobine et le circuit magnétique du coté de l'entrefer mobile :

$$\text{cun} = \text{lmag} - \text{lbob}$$

Calcul d'une deuxième contrainte géométrique qui doit être positive. Elle représente la distance entre la bobine et le circuit magnétique dans le sens de la hauteur.

$$\text{cdeux} = \text{hmag} - \text{hcyl} - \text{ecyl} - \text{rnoy} - \text{rbob}$$

Calcul d'une troisième contrainte géométrique qui doit être positive. Elle représente la différence entre la section efficace de cuivre et la section disponible.

$$\text{ctrois} = \text{stot} - \text{seff}$$

Ce qui donne dans Pro@DESIGN :

```
cun         = lmag - lbob;
cdeux       = hmag - hcyl - ecyl - rnoy - rbob;
seff        = n * Math.PI * pow(dcond,2) / 4;
stot        = (rbob - rnoy) * lbob;
ctrois      = stot - seff;
```

V.C.2.b- CONTRAINTES ELECTRIQUES

Calcul du courant maximum dans la configuration maximale pour la tenue en courant (thermique). Nous considérons que le courant est maximum lorsque la tension est maximum (nous considérons ici l'inductance négligeable) :

$$\text{imax} = \frac{\text{eeffmax}}{\text{rebob}}$$

On calcule la densité de courant correspondante que l'on va contraindre en dessous d'un seuil admissible. Il s'agit d'un fonctionnement en régime transitoire

$$\text{dens} = \frac{4 \cdot \text{imax}}{\text{dcond}^2 \cdot \pi}$$

On calcule la tension maximale entre deux spires adjacentes quand la bobine est soumise à la tension de foudre :

$$\text{vmax} = \frac{\text{lbob} \cdot \text{efoudre}}{\text{dcond} \cdot \text{n}}$$

Ce qui donne dans Pro@DESIGN :

```
imax        = eeffmax / rebob;
dens        = 4 * imax / (pow(dcond,2) * Math.PI);
vmax        = lbob * efoudre / (dcond * n);
```

V.C.2.c- CONTRAINTES MAGNETIQUES

On calcule l'induction magnétique pour empêcher le modèle d'aller en saturation

```
bnoystat = phi(ibobstat, entrefe) / (Math.PI * pow(rnoy,2)
```

V.C.2.d- OBJECTIF DU DIMENSIONNEMENT

On récapitule les dimensions critiques pour définir l'encombrement du dispositif :

longue = lmag

largeur = pmag

hauteur = hmag - hcyl - ecyl - rnoy + rbob

Ce qui conduit facilement à la fonction objectif :

volume = longue . largeur . hauteur

Nous obtenons donc les équations suivantes :

longue = lmag;

largeur = pmag;

hauteur = hmag - hcyl - ecyl - rnoy + rbob;

*volume = longue * largeur * hauteur;*

V.D Modélisation dynamique

V.D.1-EQUATIONS PHYSIQUES

La plupart des équations décrites en modélisation statique restent valables, mais en dépendant maintenant du courant et de l'entrefer. La différence réside principalement dans le calcul de la permittivité équivalente (qui ne vaut plus mufer maintenant mais dépend du courant et de l'entrefer) et l'introduction des équations différentielles.

V.D.1.a- CALCUL DE LA PERMITTIVITE

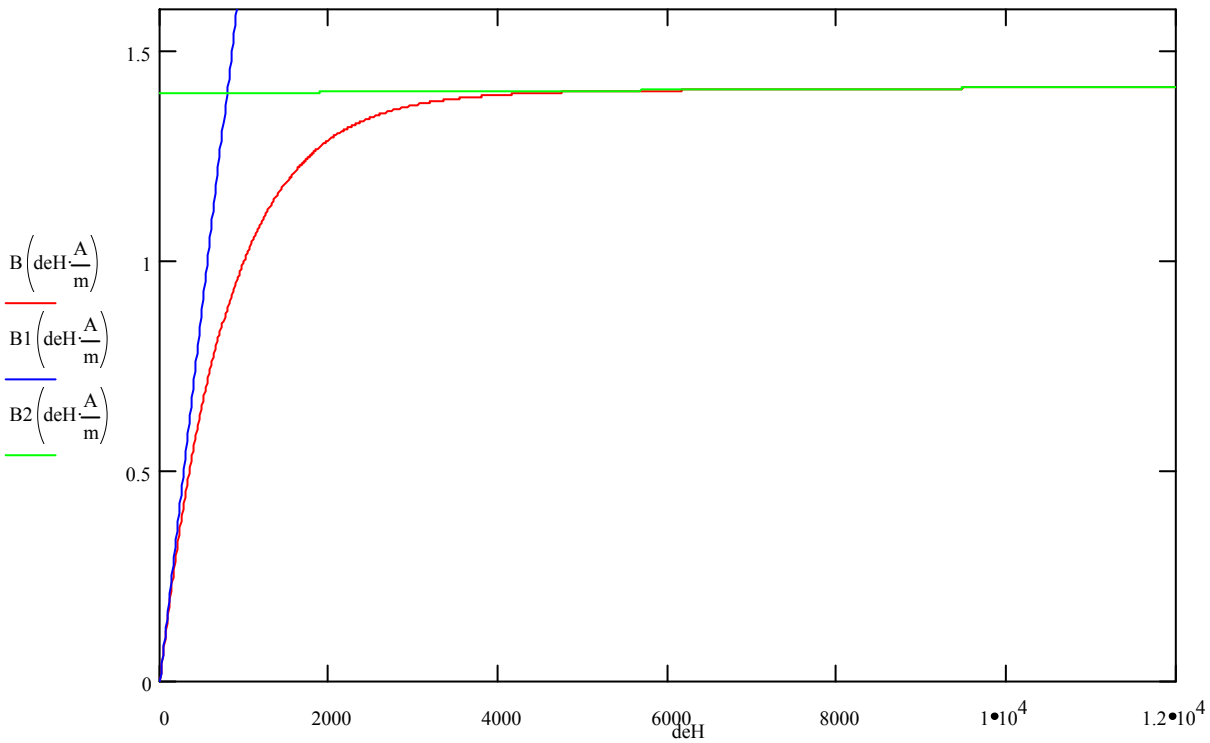
L'idée est de ramener le problème à un problème électrique avec une inductance qui dépend des paramètres courant et longueur de l'entrefer.

Calcul de µequ=B(H(i))/H(i) avec une interpolation du type droite atténuée par une exponentielle Cette modélisation de la perméabilité du fer, n'est pas très fidèle, mais nous ne disposons pas de données concernant les caractéristiques magnétiques du dispositif. Mieux adapter la modélisation de la courbe B(H) doit permettre d'obtenir des résultats plus précis sur le dimensionnement, mais ne remet pas en cause le principe de l'optimisation dynamique qui va suivre.

$$B1(Hf) := \mu fer \cdot Hf$$

$$B2(Hf) := \mu 0 \cdot Hf + Bsat$$

$$B(Hf) := (\mu 0 \cdot Hf + Bsat) \cdot \left(1 - e^{-\mu fer \frac{Hf}{Bsat}}\right)$$



H dépend de i d'après l'équation de Maxwell rot(H)=J que l'on intègre sur un contour : la ligne médiane du circuit magnétique.

$$Lmoy(z) := lbob + dbob + emag + 2 \cdot hmag - 2 \cdot hcyl - 2 \cdot rnoy + lmag + z$$

$$Hcm(i, z) := n \cdot \frac{i}{Lmoy(z)}$$

Nous obtenons donc :

muequ(i,z) = B(Hcm(i,z)) / Hcm(i,z) si i > 0,

muequ(i,z) = mufer sinon.

Ce qui donne dans Pro@DESIGN :

*H(i,z) = max(n * i / (lbob + dbob + emag + 2 * hmag - 2 * hcyl - 2 * rnoy + lmag + z),1E-10)*

*B(Hf)= (muz * Hf + Bsat) * (1 - exp(- mufer * Hf / Bsat))*

*muequ(i,z) = heaviside(-1e-10+i) * B(H(i,z)) / H(i,z) + heaviside(1e-10-i) * mufer*

Remarque :

- Nous avons introduit la fonction heaviside définie comme suit :

  heaviside(x) = 0 si x <0

  heaviside(x) = 1 sinon.

- Pour le calcul de H, on prend le max avec une valeur faible pour éviter de diviser par zéro ensuite si i = 0

V.D.1.b- INTRODUCTION DES EQUATIONS DIFFERENTIELLES

Rappel des équations :

$$\begin{cases} U(t) = R*i(t) + \dfrac{d}{dt}(L(i,z)*i(t)) \\ m\dfrac{d^2 z}{dt^2} = F(i,z) \end{cases}$$

Avec : i, le courant dans la spire.

z, la longueur de l'entrefer

R, la résistance de la bobine

L, l'inductance de la bobine

m, la masse du noyau.

F, la force globale exercée sur le noyau (force magnétique plus charge, plus frottement…)

Ces équations ne sont pas utilisables en l'état, un retraitement est nécessaire.

1ere équation :

Moyennant un calcul mathématique sur les dérivées, elle peut être transformée de la manière suivante :

$$\frac{di}{dt} = \frac{U(t) - Ri(t)}{i\frac{\partial L}{\partial i} + L(i,z)}$$

Nous introduisons la fonction w(i,z) qui représente ∂L/∂i. Nous l'approximons de la manière suivante :

$$w(i,z) = \frac{L(i+\delta,z) - L(i,z)}{\delta}$$

avec δ = 1E-6 (valeur arbitraire)

2ème équation :

Nous introduisons la vitesse qui correspond à la dérivée de l'entrefer :

$$v = \frac{dz}{dt}$$

Ce qui donne dans Pro@DESIGN :

```
w(i,z)= (inbob(i+1E-6,z) - inbob(i,z)) / 1E-6;
deriv_i(i,z,t)  = heaviside(i) * (eeffmin * sqrt(2) * sin (2 * Math.PI * 50 * t) - rebob * i)
                  / (i * w(i,z) + inbob(i,z));
deriv_v(i,z)= heaviside(z) * (- heaviside(force(i,z) - freac) * (force(i,z) - freac) / masse);
deriv_z(v,z)= v;
```

L'introduction du terme heaviside(i) dans deriv_i permet d'éviter au courant de devenir négatif, représente le fonctionnement du thyristor.

L'introduction du terme heavisite(z) dans deriv_v se justifie comme suit. Quand le dispositif a décollé, le noyau suit une trajectoire régie par l'équation différentielle. Une fois qu'il a atteint la fin de sa course le modèle que nous utilisons est complètement faux. En effet un entrefer négatif, n'est pas physiquement possible. Néanmoins cela est mathématiquement possible. Nous devions donc veiller à donner un sens au modèle pour après la fermeture.

La première hypothèse consiste à forcer z à rester à zéro une fois qu'il a atteint le zéro. Cela peut se faire en supprimant toute vitesse dès que z atteint zéro, c'est à dire mettre un heaviside sur la vitesse. Ce modèle semble plus proche de la réalité mais il est valable dans un domaine qui ne nous intéresse pas ! Après la fermeture il y a un rebond, une déformation élastique. Ce qui nous intéresse pour l'instant c'est exclusivement l'instant de fermeture, et nous avons préféré modéliser un projectile qui continue sur sa lancée à vitesse constante. Pourquoi ?

Avant l'impact, z est positif, mais on ne peut pas déterminer l'instant de l'impact à partir de cette seule donnée. En revanche si on connaît la position du projectile à un instant donné quelconque après l'impact, on peut déterminer rigoureusement l'instant de l'impact à partir de cette seule donnée. Cette modélisation correspond encore moins à la réalité, mais elle présente l'avantage de nous donner des informations sur un élément qui nous intéresse : l'instant de fermeture. Pour cela nous avons « coupé » l'accélération du dispositif après l'impact en mettant un heaviside sur la force.

Nous exprimons le système des dérivées en utilisant la syntaxe Pro@DESIGN (sachant que les valeurs initiales sont : courant nul, entrefer initial, vitesse nulle) :

[i,z,v] = ode([deriv_i(i,z,t), deriv_z(v,z), deriv_v(i,z)],[i=0, z=entrefe, v=0],t);

On introduit également dans Pro@DESIGN les fonctions ci-dessous dépendant du temps afin de tracer des courbes :

calc_z = z(t);

calc_i = i(t);

calc_v = v(t);;

V.D.2- EQUATIONS PERMETTANT D'EXPRIMER DES CONTRAINTES

V.D.2.a- CONTRAINTE SUR LE TEMPS DE FERMETURE

L'outil Pro@DESIGN permet de définir ce qui est appelé des fonctions implicites. En clair cela signifie que l'outil cherche les valeurs permettant d'annuler une fonction.

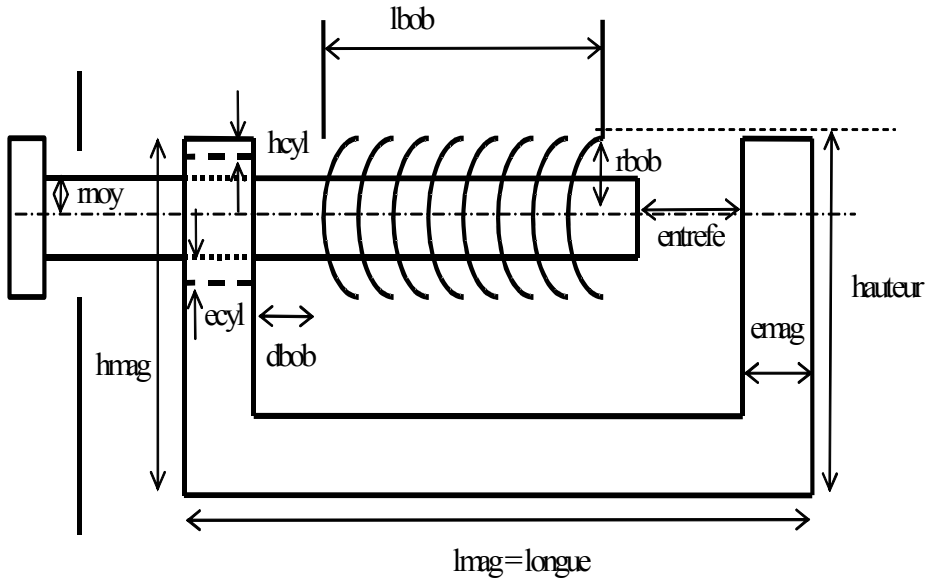
Nous définissons donc le système implicite suivant qui revient à chercher le temps tferm qui résout l'équation z(tferm) = 0 :

[tferm] = impl([z(t)],[t=0.01]);

La valeur initiale t = 0,01 correspond à l'initialisation de l'algorithme, c'est à dire l'instant à partir duquel on va rechercher la solution. On a vu que notre modèle post-impact donnait une information très précise sur l'instant de cet impact, c'est pourquoi nous commençons les recherches sur un instant qui majore presque à coup sûr l'instant de fermeture.

V.E Optimisations

Afin de permettre l'optimisation du dimensionnent, nous libérons une partie des paramètres d'entrée. Les valeurs correspondent à peu près au dispositif existant.



Plages de variation des paramètres d'entrée « libérés » et valeurs initiales

Paramètre	**Valeur mini**	**Valeur maxi**	**Valeur initiale**
dcond	1^{E}-4	5^{e}-4	1.2^{E}-4
emag	6E-4	1E-3	8.0E-4
hcyl	0.001	0.008	0.0025
hmag	0.001	0.05	0.013
lbob	0.001	0.05	0.02
lmag	0.001	0.05	0.0205
n	100.0	8000.0	2100.0
pmag	0.005	0.05	0.012
rbob	0.003	0.01	0.006
rnoy	0.0010	0.0050	0.0020

Paramètres qui restent figés :

Paramètre	**Valeur fixe**
Bsat	1.4
dbob	1^{E}-3
ecyl	6^{E}-4
eeffmax	400.0
eeffmin	50.0
efoudre	6000.0
entrefe	0.0020
freac	0.75
mfer	12000.0
murel	1400.0
muz	1.257E-6
refui	1.0E9
rho	1.7241E-8
taux	1.2

V.E.1- RESULTATS EN UTILISANT LA MODELISATION STATIQUE

Nous définissons les contraintes suivantes :

Paramètre	Type contrainte	Valeur min	Valeur max
bnoystat	Intervalle	0	1.4
cun	Intervalle	0.0001	0.01
cdeux	Intervalle	1^{e}-6	1^{E}-4
ctrois	Intervalle	0.0001	0.01

vmax	Intervalle	100	1750
dens	Intervalle	1E8	2e8
volume	Fonction objectif	0	5E-6

Nous obtenons le résultats suivants après 10 itérations :

Entrée/sortie	Paramètre	Valeur initiale (itération 1)	Valeur optimum (itération 10)	Ratio
Entrée	dcond	1,20E-04	1,00E-04	-17%
Entrée	emag	8,00E-04	8,01E-04	0%
Entrée	hcyl	2,50E-03	2,40E-03	-4%
Entrée	hmag	1,30E-02	1,81E-02	39%
Entrée	lbob	2,00E-02	4,11E-03	-79%
Entrée	lmag	2,05E-02	4,21E-03	-79%
Entrée	n	2,10E+03	2,47E+03	18%
Entrée	pmag	1,20E-02	5,00E-03	-58%
Entrée	rbob	6,00E-03	9,96E-03	66%
Entrée	rnoy	2,00E-03	5,00E-03	150%
Sortie	dens	4,40E+08	2,00E+08	-55%
Sortie	Forcedecol	6,29E+00	5,13E-01	-92%
Sortie	hauteur	1,39E-02	2,00E-02	44%
Sortie	largeur	1,20E-02	5,00E-03	-58%
Sortie	longue	2,05E-02	4,21E-03	-79%
Sortie	vmax	4,76E+02	1,00E+02	-79%
Sortie	volume	3,42E-06	4,22E-07	-88%

Il est important de rappeler ici que nous obtenons un dimensionnement d'après une modélisation statique. Le temps de fermeture n'est donc pas pris en compte. Si nous ne disposions pas d'un outil permettant de faire une modélisation dynamique, l'étape suivante consisterait à valider *a posteriori* les temps de fermeture. Nous testons le dimensionnement obtenu dans une simulation en dynamique en utilisant le modèle dynamique présenté au-dessus.

V.E.2- INVALIDATION EN DYNAMIQUE DU RESULTAT DU CALCUL STATIQUE

Nous introduisons les équations physiques du modèle dynamique. Par contre nous n'introduisons pas de contraintes supplémentaires

Après simulations, nous découvrons qu'en réalité la force magnétique est insuffisante pour faire décoller la masselotte ! De ce fait le système ne ferme pas. L'optimum du modèle statique n'est pas une solution valable pour le dispositif.

V.E.3- RESULTAT EN DYNAMIQUE SUITE INTRODUCTION CONTRAINTE SUR FORCE DE DECOLLAGE

Avant d'exprimer une contrainte sur le temps de fermeture sur le modèle dynamique, il semble intéressant de passer par cette étape intermédiaire qui consiste à introduire une contrainte qui va « obliger » la fermeture du dispositif. Il peut sembler plus facile de rester dans un modèle statique en se garantissant un « bon fonctionnement » du dispositif.

En effet, dans le cas précédent, le dispositif ne ferme pas car à courant maximum la force magnétique n'est pas suffisante pour vaincre la force de réaction du dispositif. Un raisonnement peut donc être d'ajouter une contrainte sur la force de décollage.

Nous imposons donc Forcedecol a être compris entre 0,1 et 10N. Nous introduisons également la fonction implicite qui nous donnera le temps de fermeture du dispositif ainsi optimisé.

Nous obtenons la solution suivante :

MODELE DYNAMIQUE

Cas D1

Sans contrainte sur tferm

Contrainte Forcedécol entre 0,1 et 10N

		Itération 1	**Itération 11**
Type	**Paramètre**	**Initiale**	**Optimum**
Entrée	dcond	1.20E-04	1.00E-04
Entrée	emag	8.00E-04	1.00E-03

Entrée	hcyl	2.50E-03	2.44E-03
Entrée	hmag	1.30E-02	1.61E-02
Entrée	lbob	2.00E-02	4.74E-03
Entrée	lmag	2.05E-02	4.84E-03
Entrée	n	2.10E+03	2.84E+03
Entrée	pmag	1.20E-02	5.00E-03
Entrée	rbob	6.00E-03	8.96E-03
Entrée	rnoy	2.00E-03	4.04E-03
Sortie	dens	4.40E+08	2.00E+08
Sortie	*Forcedecol*	*1.67E+00*	*1.00E-01*
Sortie	**hauteur**	**1.39E-02**	**1.80E-02**
Sortie	**largeur**	**1.20E-02**	**5.00E-03**
Sortie	**longue**	**2.05E-02**	**4.84E-03**
Sortie	tferm	5.44E-03	3.84E-02
Sortie	vmax	4.76E+02	1.00E+02
Sortie	**volume**	**3.42E-06**	**4.36E-07**

Nous obtenons maintenant un dispositif qui ferme, mais son temps de fermeture est d'environ 38ms.

Nous traçons la courbe z(t) pour information :

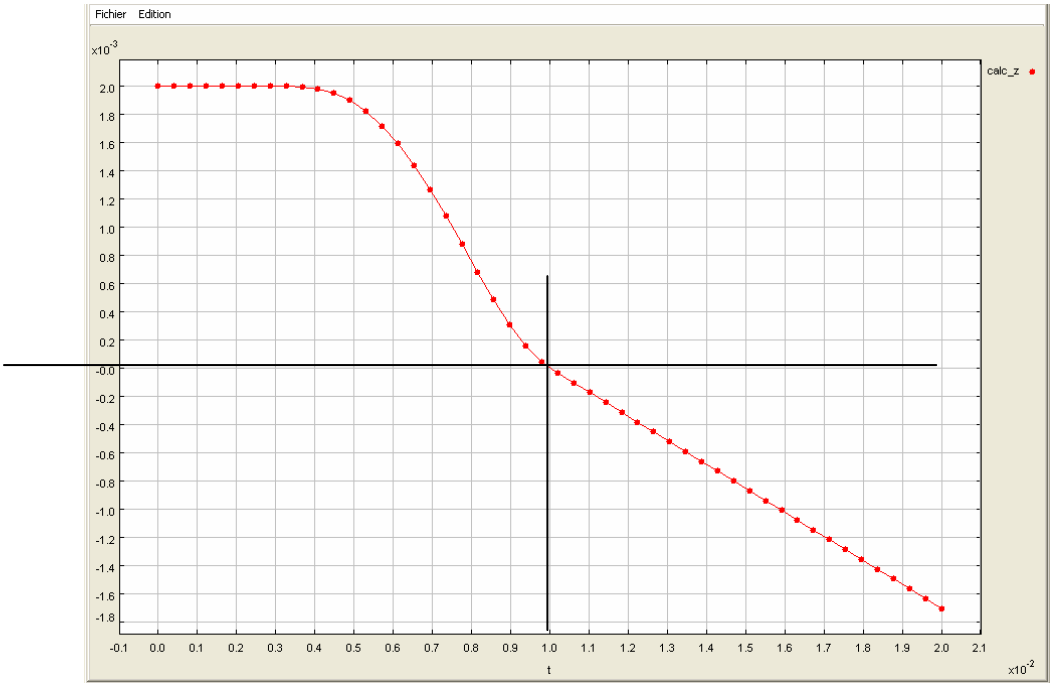


Figure 27 : Courbe de l'entrefer en fonction du temps

Cette courbe montre que le dispositif décolle, puis se referme. En effet la force appliquée sur la masselotte est égale à la différence entre la force magnétique et la force du ressort tant que la force magnétique est supérieure à la force du ressort. Dans le cas contraire, la force est nulle et le modèle est faux.

Par cette méthode, nous pourrions ainsi intervenir sur la valeur minimale de la force de décollage pour diminuer le temps de fermeture et donc obtenir un dispositif optimisé par rebouclages successifs.

Il reste cependant plus facile et plus naturel d'introduire directement une contrainte sur le temps de fermeture.

V.E.4- RESULTAT EN DYNAMIQUE AVEC CONTRAINTE SUR LE TEMPS DE FERMETURE

Nous contraignons le temps de fermeture à fermer en moins de 10 ms en relâchant la contrainte sur la force de décollage qui n'est plus nécessaire ici.

Nous obtenons les résultats suivants :

MODELE DYNAMIQUE

Cas D2

Contrainte Tferm entre 0 et 0,01s

Sans contrainte sur Forcedecol

Itération 10

Type	**Paramètre**	**Optimum**
Entrée	dcond	1.00E-04
Entrée	emag	1.00E-03
Entrée	Hcyl	2.47E-03
Entrée	hmag	1.45E-02
Entrée	Lbob	5.44E-03
Entrée	Lmag	5.54E-03
Entrée	N	3.26E+03
Entrée	Pmag	5.00E-03
Entrée	Rbob	8.11E-03
Entrée	Rnoy	3.21E-03
Sortie	Dens	2.00E+08
Sortie	*Forcedecol*	*3.01E-01*
Sortie	**Hauteur**	**1.63E-02**
Sortie	**Largeur**	**5.00E-03**
Sortie	**Longue**	**5.54E-03**
Sortie	Tferm	1.00E-02
Sortie	Vmax	1.00E+02
Sortie	**Volume**	**4.52E-07**

Nous remarquons que la force de décollage est d'environ 0,3N et que le volume est légèrement supérieur que dans le cas précédent.

Nous profitons des possibilités de l'outil Pro@design pour tracer un certain nombre de courbes qui semblent intéressantes :

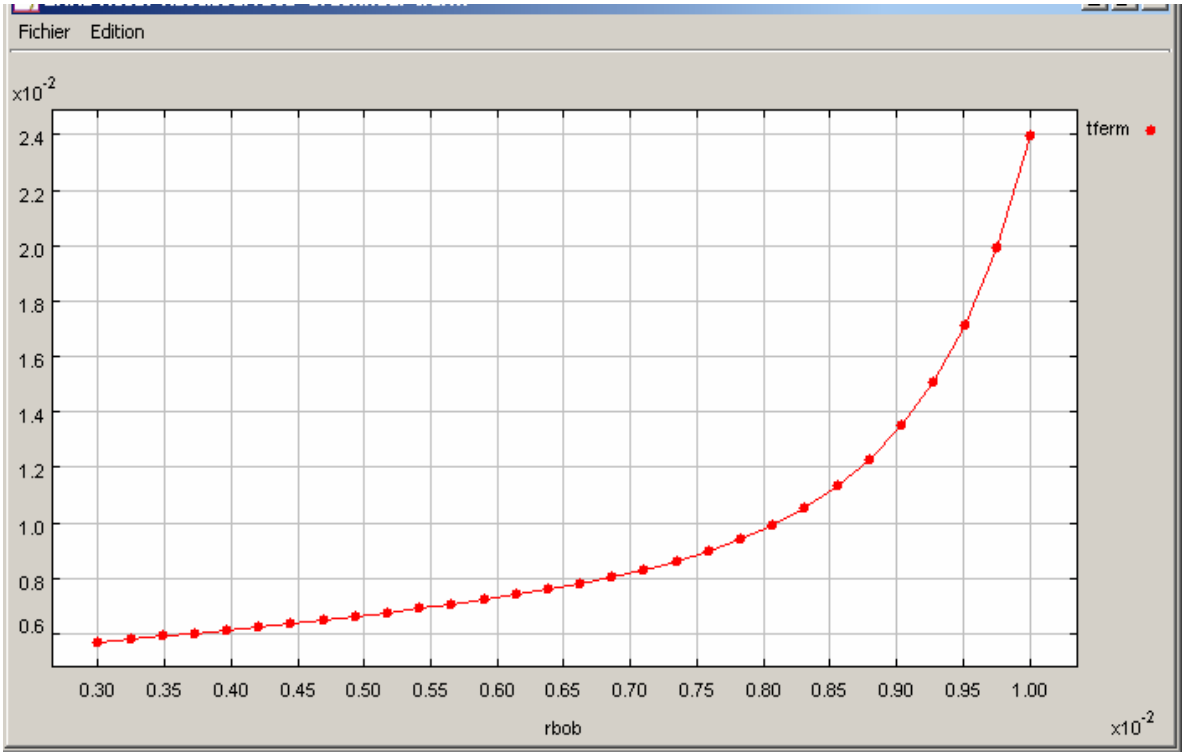


Figure 28 : Temps de fermeture en fonction du rayon de la bobine

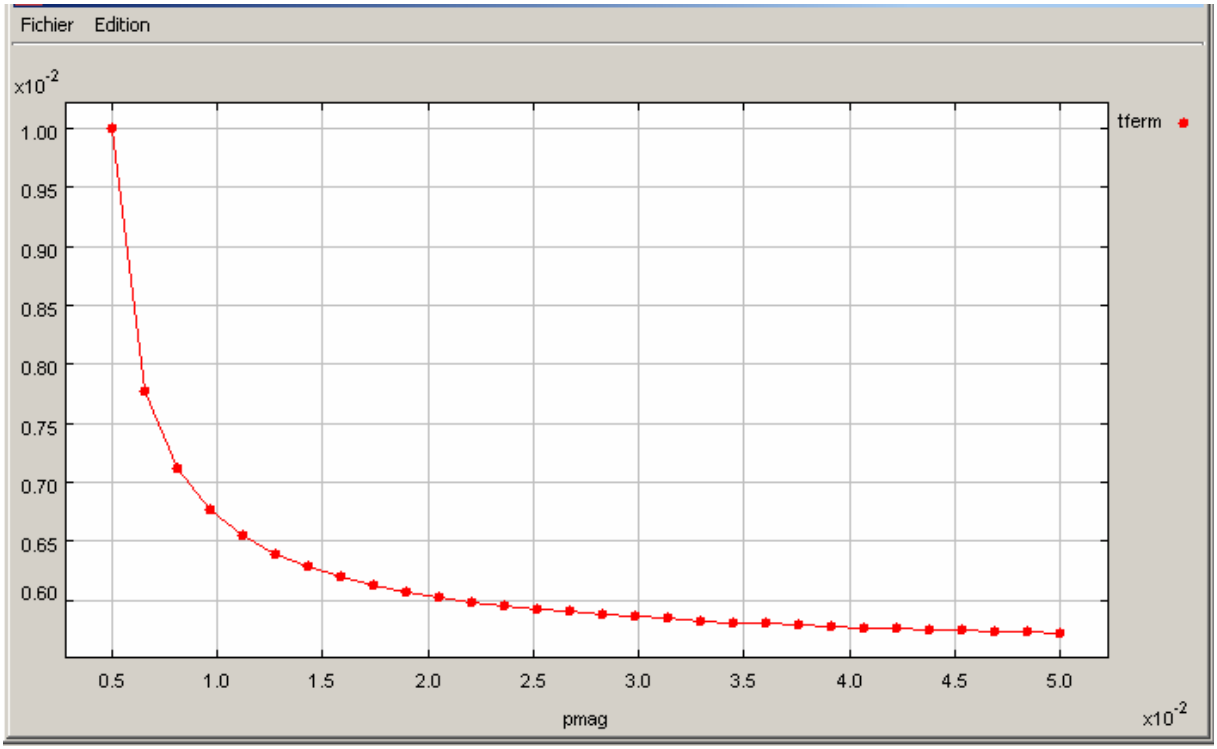


Figure 29 : Temps de fermeture en fonction de la profondeur du circuit magnétique

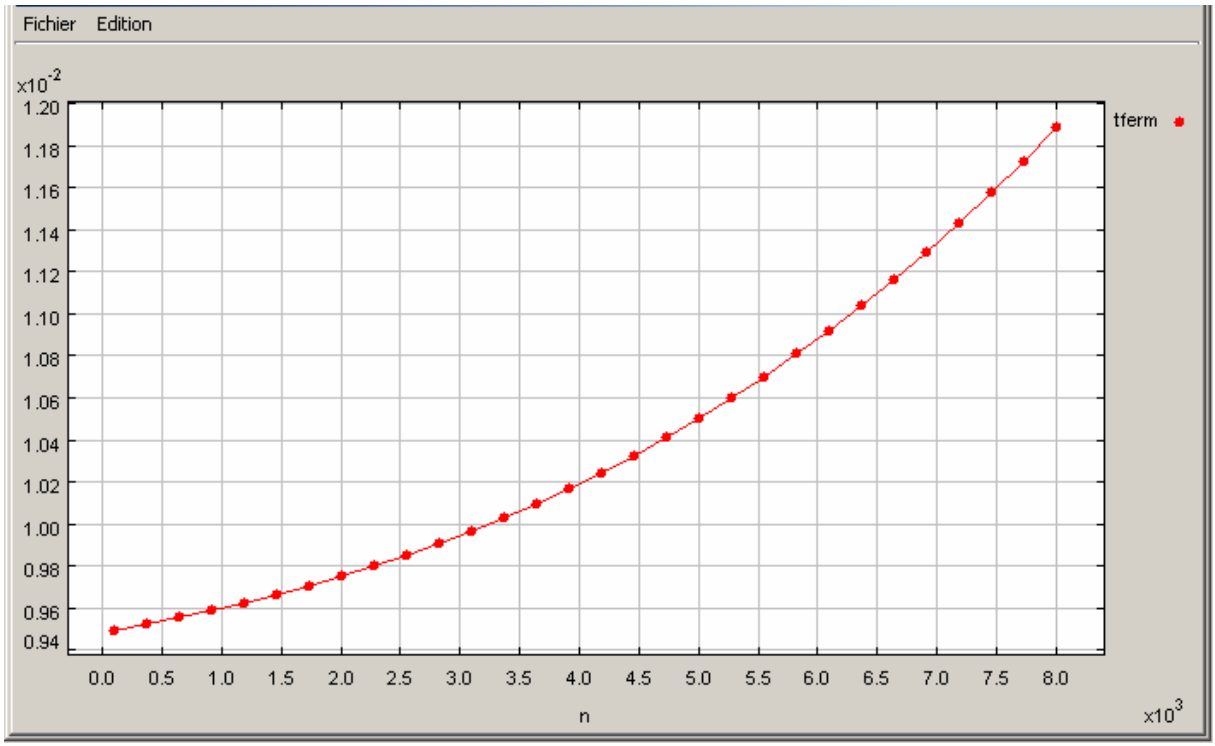


Figure 30 : Temps de fermeture en fonction du nombre de spires

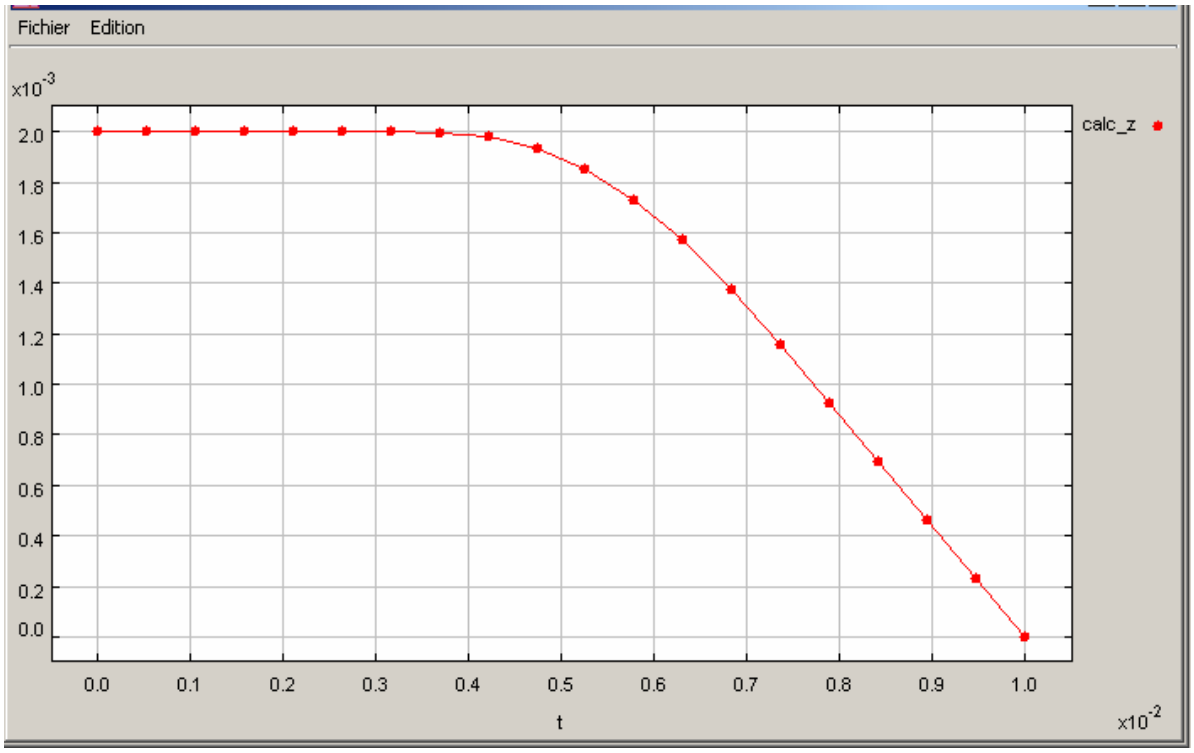


Figure 31 : Entrefer en fonction du temps

Cette dernière courbe montre que le « temps de vol » est d'environ 6 ms et n'est donc pas du tout négligeable.

V.F Conclusion

Dans cette étude nous avons présenté une démarche qui nous a semblé significative de l'utilisation avancée d'un outil de dimensionnement. En effet la démarche présentée a consisté à dimensionner le dispositif en tenant compte de considérations statiques. On voit alors que si l'on ne prend pas des marges de sécurité colossales sur les grandeurs qui influent sur la dynamique ultérieure du dispositif (en contraignant la force de décollage à 0.1 N) on risque de proposer des solutions qui ne satisfont pas les contraintes dynamiques. La puissance d'un outil de dimensionnement automatique comme Pro@DESIGN réside dans la possibilité d'exprimer directement les contraintes plutôt que de prendre de marges de sécurité qui sur-dimensionnent les dispositifs. Les phénomènes transitoires n'échappent pas à cette règle.

TABLE DES SYMBOLES

La Nature est un temple où de vivants piliers
Laissent parfois sortir de confuses paroles ;
L'homme y passe à travers des forêts de symboles
Qui l'observent avec des regards familiers.

Baudelaire (Charles)

.NET	Nom de la technologie de serveur d'application de Microsoft.
CAD	Computer Aided Design. Essentiellement le marché du logiciel de dessin industriel Catia, Pro/Engineer …
CAE	Computer Aided Engineering. Marché industriel du calcul pour l'ingénierie. Essentiellement des logiciels de simulation fine, type élément fini.
CdiCC	Nom d'un langage développé au cours de cette thèse qui sert de générateur de programme générateur.
COB	Computational OBject. Composant logiciel contenant la version calculable d'un modèle analytique, développé au cours de cette thèse
COM	Nom de la technologie de composant de Microsoft
CORBA	Technologie d'interopérabilité, et de distribution de programmes en langage orienté objet.
EJB	Entreprise JavaBean. Composant d'un serveur d'application J2EE.
Gen-X	Composant logiciel pour le générateur développé durant cette thèse.
J2EE	Java 2 Entrepise Environment. Plateforme de serveur d'application de la nébuleuse Java
JavaBean	Nom de la technologie de composant de Java
OMG	Object Management Group. Organisme de normalisation de CORBA entre autres.
PIDO	Process Integration Design Optimisation. Marché émergent de gestion de processus de calcul de logiciel du CAE. Le concept est flou, et il n'est pas garanti que le nom subsiste longtemps. Dans la thèse nous appelons logiciels du PIDO les logiciels qui exploitent, et répètent le calcul de un ou plusieurs logiciels éléments finis, en vue d'une optimisation sous contraintes.
SQP	Sequential Quadratic Programming. Nom de la technique d'optimisation sous contraintes.
UML	Unified Modelling Language. Ensemble de symboles et de conventions pour décrire un programme informatique. Sous tend une méthode unifiée de développement logiciel.

BIBLIOGRAPHIE

[Ahrens-93] Sigurd Ahrens, “Système expert d’aide au diagnostic et à la maintenance de grands entraînements électriques”, Thèse INPG, Génie Electrique, Grenoble, 1993

[Bel Habib-00]* Bel Habib Basma, “Méthodologie pour le Développement de Plates Formes Intégrées dédiées à la Conception en Génie Electrique”, Thèse INPG, Génie Electrique, Grenoble, juillet 2000

[Bergeon-98]* Stéphane Bergeon, “Contribution à une méthodologie de dimensionnement des convertisseurs statiques”, Thèse INPG, Génie Electrique, Grenoble, 1998

[Blanco-98]* Eric Blanco, « l’émergence du produit dans la conception distribuée », Thèse INPG, Génie Industriel, décembre 98

[Bolopion-75] Alain Bolopion, “Mise en oeuvre et expérimentation pédagogique d’un système d’enseignement assisté par Ordinateur (E.S.P.A.C.E)”, Thèse INPG, Génie Electrique, Grenoble, 1975

[Delinchant-00]* Benoit Delinchant, “Contribution à l’approche composant pour la conception en genie électrique”, rapport de D.E.A. INPG, Génie Electrique, Grenoble, 2000

[Gamma-95]* Erich Gamma, Richard Helm, Ralph Johnson, John Vlissidès, “Design Patterns”, Addison Wesley, Reading, MA, 1995

[Ganascia-96]* Jean-Gabriel Ganascia, “Les sciences cognitives”, flammarion, 1996

[Gentilhomme-91]* Alain Gentilhomme, “C.O.C.A.S.E Un système expert d’aide à la conception d’appareillages electriques”, Thèse INPG, Génie Electrique, 1991

[Gerbaud]* Laurent Gerbaud “Aide à la conception des ensembles machine - convertisseur - commande : apport d’une approche système expert”, Thèse INPG, Génie Electrique, Grenoble,

[Gosling-95]* James Gosling, H. McGilton, “The Java Language Environment”, Sun Microsystems Computer Company, May 1995

[Guichon-01]* Jean-Michel Guichon, “Modélisation, caractérisation et dimensionnement de jeux de barres”, Thèse INPG, Génie Electrique, Grenoble, novembre 2001

[Harani-97]* Yasmina Harani, “Une Approche Multi-Modèles pour la capitalisation des connaissances dans le domaine de la conception”, Thèse INPG, Génie Industriel, Grenoble, Novembre 1997

[Harwell-87]* Harwell, “Harwell subroutine specification”, October 1987

[Haton-91] Jean-Paul Haton, Buizid Najet, François Charpillet, “Le raisonnement en Intelligence Artificielle”, InterEditions, Paris, 1991

[Jufer] Marcel Jufer, “Traité d’électricité”, Volume IX, électromécanique, Presse Polytechnique et Universitaire Romandes.

[Karnopp-90] Dean C. Karnopp, Donald L. Margolis, Ronald C. Rosenberg, “System dynamics: A Unified Approach”, Second Edition, John Wiley, New York, 1990

[Kone-93]* A.D Kone, B. Nogarede, M. Lajoie-Mazenc, “Le dimensionnement des actionneurs électriques, un problème de programmation non linéaire”, Journal de Physique III, Février 1993, pp. 285-301

[Larouci-02]* Chérif Larouci, “Conception et optimisation de convertisseurs statiques pour l’électronique de puissance. Application aux structures à absorption sinusoïdale”, Thèse INPG, Génie Electrique, Grenoble, mai 2002

[Lawrence] C.T. Lawrence, J.L.Zou, A.L. Tits, “User’s guide for CFSQP version 2.5”, Electrical Engineering Department and Institue for System Research, University of Maryland

[Lechevalier-97]* Christophe Lechevalier, Laurent Gerbaud, Jean Bigeon, “A functional description for static converter structures design”, EPE’97, Trondheim, Nrway, 8-10 September 1997, pp. 2 794-2 799

[Macsyma-95]* Macsyma, “user’s guide”, USA, 1995

[Maple-96]* Maple V, “Learning guide”, USA 1996

[Mathcad-98]* Mathcad, “user’s guide”, Mathsoft Inc, 1998

[Matlab-96]* Matlab, “User’s guide”, The Math works Inc, 1996

[Meijler-95]* Theo D. Meijler, Oscar Nierstrasz, “Beyond Objects: Components”, Software Composition Group, Université de Berne, 1995

[Mer-98]* Stéphane Mer, « les mondes et les outils de la conception pour une approche socio-technique de la conception de produit », Thèse INPG, Génie Industriel, novembre 1998

[Nierstrasz-93]* Oscar Nierstrasz, “Composing Active Objects”, Research Directions in Concurrent Object-Oriented Programming, ed. G. Agha, P. Wegner, A. Yonezawa, MIT Press, Cambridge Mass., 1993, pp. 151-171

[Pitrat-95]* Jacques Pitrat, “De la machine à l’intelligence”, Hermès, Paris , 1995

[Press-92] W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery, “Numerical recipes in C : the art of scientific computing”, Second Edition, Cambridge University Press, ISBN 0-521-43108-5, 1992

[Roters-41]* H.C. Roters, “Electromagnetic devices”, John Wiley and Sons Inc, New-York, 1941

[Singh-92]* C. Singh, D. Sarkar, “Practical consideration in the optimisation of induction motor design”, IEE Proceedings-B, Vol. 139, No 4, July 1992

[Stewart-99] G.W. Stewart, “JAMPACK, a Java package for matrix computations”, ftp://math.nist.gov/pub/Jampack/Jampack/AboutJampack.html, University of

Maryland, février 1999

[Taton-95-1]* René Taton, “La science Antique et Médievale”, Quadrige, Presse Universitaire de France, 1995

[Taton-95-2]* René Taton, “La science Moderne”, Quadrige, Presse Universitaire de France, 1995

[Taton-95-3]* René Taton, “La science Contemporaine - 1”, Quadrige, Presse Universitaire de France, 1995

[Taton-95-4]* René Taton, “La science Contemporaine - 2”, Quadrige, Presse Universitaire de France, 1995

[Trichon-91]* François Trichon, “Modélisation du processus de conception des machines électriques” ,Thèse INPG, Génie Electrique, Grenoble, mars 1991

[Uml]* http://www.uml.org

[Udell-94]* James Udell, “Componentware”, Byte, vol. 19, no. 5, May 1994, pp. 46-56

[Vignaux-91]* Georges Vignaux, “Les sciences cognitives: une introduction”, La découverte, Paris, 1991

[Wurtz-96]* Frédéric Wurtz, “Une nouvelle approche pour la conception sous contraintes de machines électriques”, Thèse INPG, Génie Electrique, Grenoble, Mai 1996

Résumé

Les équations mathématiques constituent un formalisme essentiel pour la science. Le dimensionnement doit pouvoir s'appuyer sur ce formalisme, et disposer d'une méthode performante.

En se fondant sur une analyse de la place du dimensionnement au sein des entreprises, des outils existants et des méthodes de développement logiciel, cette thèse présente une méthodologie et une technologie à base de composants d'aide au dimensionnement automatisé en ingénierie.

Quelques cas industriels de dimensionnement de dispositifs électromécaniques servent à illustrer la méthode utilisée ainsi que le traitement de modèles à l'aide d'équations différentielles et implicites.

Les expériences menées au cours de ce travail conduisent à estimer que le dimensionnement, dans les services de conception, peut se faire à partir de modèles mathématiquement formalisés en utilisant cette méthodologie. La technologie développée semble mûre pour fournir l'aide adéquate.

Abstract

Mathematical equations are an essential language in science. We propose a methodology that deals with them in an efficient way to size industrial devices.

By analyzing the place of sizing among the company's services, and the existing tools used nowadays for sizing, and the software engineering methods, mainly on component based methods, this document presents a methodology and a component based technology for automated sizing in engineering.

The method is illustrated using some industrial in electro mechanics, dealing with differential, and implicit equations.

The experiments carried out have let us be confident that sizing industrial device can be based on mathematical equations operating such a methodology. The technology is now ready to satisfy the needs.