DEMARCHE D'OPTIMISATION POUR DES PRODUITS ATTRACTIFS ET DURABLES

Octobre 2015

N° de contrat :
Étude réalisée pour le compte de l'ADEME par : *Non renseigné*[i]

Coordination technique ADEME : *BORTOLI PUIG Hélène* **– Direction\Service :** *ANGERS DECD SPEM*

ADEME
Agence de l'Environnement et de la Maîtrise de l'Energie

RAPPORT D'ETUDE

En partenariat avec :

Réseau Eco SD
Eco-conception de systèmes durables

REMERCIEMENTS

Citer les membres du Comité de pilotage et/ou du comité de suivi ou de relecture et/ou du consortium de recherche.

Prénom et Nom (Organisme)

CITATION DE CE RAPPORT

ADEME/EcoSD. Auteur(s). Année de publication. Titre – Rapport. Nombre de pages.
Si le rapport est en ligne ajouter : Cet ouvrage est disponible en ligne www.ademe.fr, rubrique Médiathèque (URL)

En français :

Toute représentation ou reproduction intégrale ou partielle faite sans le consentement de l'auteur ou de ses ayants droit ou ayants cause est illicite selon le Code de la propriété intellectuelle (art. L 122-4) et constitue une contrefaçon réprimée par le Code pénal. Seules sont autorisées (art. 122-5) les copies ou reproductions strictement réservées à l'usage privé de copiste et non destinées à une utilisation collective, ainsi que les analyses et courtes citations justifiées par le caractère critique, pédagogique ou d'information de l'œuvre à laquelle elles sont incorporées, sous réserve, toutefois, du respect des dispositions des articles L 122-10 à L 122-12 du même Code, relatives à la reproduction par reprographie.

En anglais :

Any representation or reproduction of the contents herein, in whole or in part, without the consent of the author(s) or their assignees or successors, is illicit under the French Intellectual Property Code (article L 122-4) and constitutes an infringement of copyright subject to penal sanctions. Authorised copying (article 122-5) is restricted to copies or reproductions for private use by the copier alone, excluding collective or group use, and to short citations and analyses integrated into works of a critical, pedagogical or informational nature, subject to compliance with the stipulations of articles L 122-10 – L 122-12 incl. of the Intellectual Property Code as regards reproduction by reprographic means.

TABLE DES MATIERES

Résumé 4
1. Contexte du projet 5
1.1. Contexte et positionnement du projet 5
1.2. Objectifs scientifiques et technologiques 7
2. Méthodologie 9
2.1. Méthodologie d'écoconception 9
2.2. Modélisation et résolution 11
2.2.1. Modélisation de la phase de conception 11
2.2.2. Modélisation du scénario de fin de vie 12
2.2.3. Modélisation des variables d'écoconception 12
2.2.4. Modélisation des variables d'écoconception 13
2.2.5. Résolution d'un modèle global 13
3. Bilan / Principaux résultats obtenus 14
3.1. Modélisation et recherche des Architectures de Produits 14
3.1.1. Description du modèle 15
3.1.2. Evaluation de l'attractivité des fonctions 19
3.1.3. Résultats obtenus 19
3.1.4. Résultats obtenus avec prise en compte de l'attractivité 24
3.2. Modélisation et recherche des Architectures de Cycle de Vie 26
3.2.1. Description des variables de conception 26
3.2.2. Description des variables de décision pour le scénario de fin de vie 26
3.2.3. Description des variables de performance 27
3.2.4. Paramètres utilisés 27
3.2.5. Résultats obtenus 28
3.3. Synthèse des résultats 29
4. Recommandations 29
5. Conclusions / Perspectives 29
Références bibliographiques 32
Index des tableaux et figures 33
Sigles et acronymes 34

Résumé

Expliquer en une vingtaine de lignes le contexte du projet, ses objectifs, ses enjeux, les principaux résultats et les suites à donner. Proposer un résumé en anglais pour valorisation à l'international.

Dans ce PRC nous proposons de repenser le processus de raisonnement d'écoconception, en agrégeant rigoureusement dès le départ l'ensemble des contraintes associées aux impacts environnementaux pour préciser le problème de conception de manière systémique et pouvoir lancer un processus d'exploration systématique et informatisé des solutions satisfaisant tous les objectifs à la fois. Techniquement, cette nouvelle approche d'écoconception prend appui sur une modélisation du problème environnemental par contrainte (CSP) et une résolution par une propagation des contraintes basée sur un algorithme d'optimisation adapté à la complexité de la problématique environnementale.

Ce changement de paradigme de méthode d'écoconception proposé se traduit par des objectifs ambitieux, novateurs et porteurs d'un gisement important :

- Dépasser les limites des approches actuelles en écoconception majoritairement basées sur l'Analyse de Cycle de Vie. L'approche systémique intégrant la modélisation des contraintes associées aux impacts environnementaux et l'exploration systématique via le raisonnement CSP permettent de travailler de manière proactive l'amélioration des impacts environnementaux.
- Révolutionner la considération de la fonctionnalité en conception pour optimiser le couple ajout de valeur et impacts environnementaux. Le caractère systématique de l'exploration d'un raisonnement CSP associé à la définition de fonctions « sensibles/négociables » permet de travailler sur l'optimisation des spécifications fonctionnelles d'un système (cahier des charges) et ainsi de véritablement rechercher les solutions compatibles avec un développement durable.

ABSTRACT

Résumé en anglais

Avertissement

Les éléments ci-dessous ont pour objectif de proposer une trame pour guider la rédaction. Ils ne sont en aucun cas obligatoires, ni ne traduisent un ordre de présentation des idées imposé.

1. Contexte du projet

Rappeler ici le contexte et l'objet de l'étude ou du projet de recherche, définir ou décrire le sujet, donner quelques éléments sur les enjeux liés au sujet

1.1. Contexte et positionnement du projet

Réduire drastiquement les impacts environnementaux dès la conception des produits, services et systèmes… tel est l'impératif absolu et urgent pour que la planète, avec sa surcharge démographique et son mode de production/consommation généralisé, reste viable à court terme.
Avec le recul, nous savons que les méthodes d'écoconception traditionnelles développées depuis 30 ans ne conviennent pas car elles raisonnent en figeant la définition fonctionnelle des systèmes considérés (performances, durée de vie) et appréhendent de façon simpliste la complexité de ces nouveaux systèmes.
Les systèmes à concevoir deviennent de plus en plus complexes car, structurellement, ils sont composés de nombreux sous-systèmes, associés dans une logique de plateforme et de gamme, qui sont souvent couplés et qui fonctionnent sur plusieurs modes de fonctionnement. Ensuite, les produits deviennent connectés, et plus généralement les interactions avec leur environnement deviennent un objet de conception, le système considéré n'étant plus le produit isolé mais le macro-système associé à la fonctionnalité. Enfin, la complexification provient des contraintes environnementales elles-mêmes.
L'évaluation environnementale qui découle d'une Analyse du Cycle de Vie (ACV) ne s'encapsule pas dans un indicateur unique mais nécessite la mesure de nombreuses catégories d'impacts distinctes et complémentaires [1]. En écoconception, l'intérêt n'est plus d'optimiser localement une dimension du cycle de vie du produit mais de considérer l'ensemble des dimensions du cycle de vie du produit ainsi que l'infrastructure support des services qui lui sont associés (Figure 1). Lorsque l'on considère de multiples cycles de vie, des améliorations fonctionnelles (upgrades) ou encore une évaluation dynamique des impacts environnementaux, les systèmes à concevoir doivent être considérés sur une période temporelle élargie. Plusieurs questions illustrent la complexité émanant des problématiques d'écoconception, comme par exemple les effets rebonds ou les effets dynamiques[2].

[1] toxicité pour l'homme, troubles respiratoires, rayonnements ionisants, appauvrissement de la couche d'ozone, oxydation photochimique, éco-toxicité aquatique, éco-toxicité terrestre, acidification du milieu aquatique, eutrophisation aquatique, acide terrestre, occupation des territoires, réchauffement climatique, énergie non renouvelable, extraction de minéraux

[2] comment gérer la prise en compte des effets rebonds (par exemple : réduction des impacts environnementaux provoquant un comportement décomplexé avec des usages abusifs) ? la définition a priori de la durée de vie optimale d'un produit ? les effets dynamiques des impacts (sur 100 ans, tenant compte des évolutions externe au système) ? la définition de la fin de vie optimale d'un système ? la prise en compte de la spatialité des impacts environnementaux (tenant compte de la fragilité de la zone géographique) ? l'optimisation multicritère d'une ACV ?

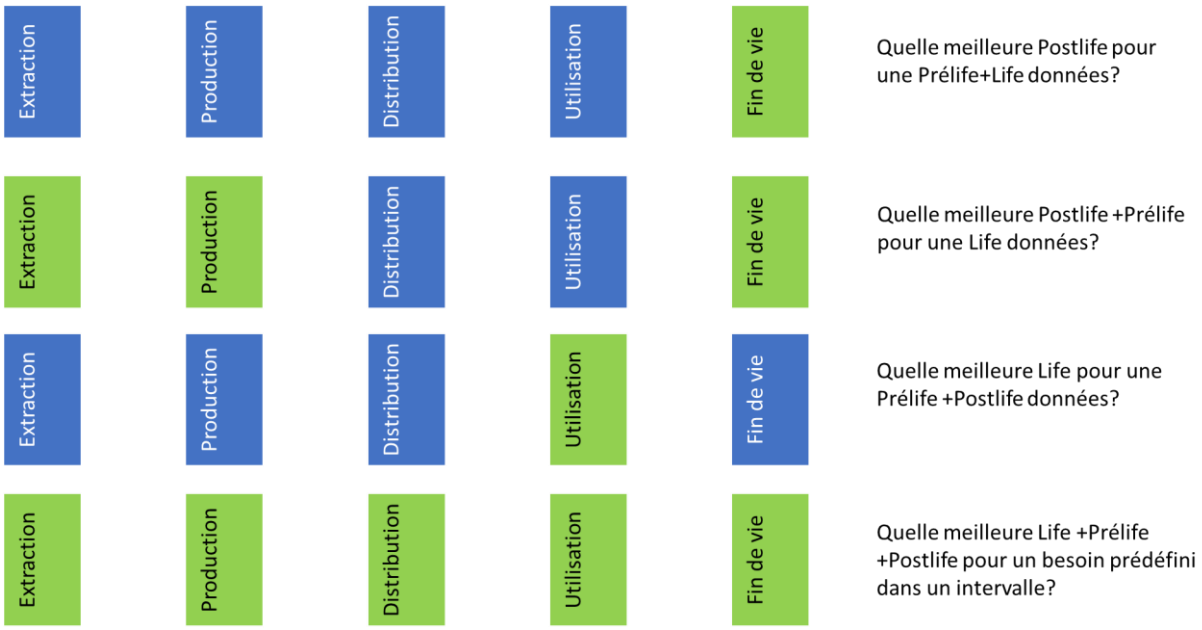

Figure 1 : complexité en écoconception

Au-delà de méthodes d'écoconception s'adressant majoritairement aux systèmes simples (méthodes génériques type ACV simplifiée, EcoPas, LIDS Wheel ou méthodes plus spécifiques type Design for (Df) recycling, Df remanufacturing, Df upgrading, Df low energy in use, Df eco-logistique, Df cleaner production), nous avons relevé 4 méthodes d'écoconception applicables aux systèmes complexes dont les avantages et inconvénients figurent dans la Figure 2.

Outils et méthodes	Références	Points forts	Points faibles
Analyse de Cycle de Vie (ACV)	(Schmidt et al., 2004), (LCA-Ship, 2004)	**+ Evaluation multicritère**	**- Fonctionnalités et architectures figées**
Design of Experiments (DoE)	(Unal et al., 1993)	**+ Identification des paramètres influents**	**- Exploration du champs des solutions limitée**
Linear programming/ LCA	(Azapagic and Clift,1999)	**+ Approche multicritères**	**- Architectures figées**
Computational approach	(van der Ploeg, 2011)	**+ Eco-Optimisation**	**- Fonctionnalités et architectures figées**

Figure 2 : Méthodes d'écoconception applicables aux systèmes complexes

De manière synthétique, ces méthodes d'écoconception conventionnelles ne permettent pas d'explorer toutes les solutions envisageables, dans une temporalité acceptable. Elles s'inscrivent en effet dans le paradigme de conception « classique » issu de **Pahl et Beitz**, processus descendant dans lequel les choix de conception successifs permettent progressivement de compléter le jeu de contraintes jusqu'à figer la définition de l'objet ; il en résulte généralement une réduction marginale des impacts environnementaux sur le cycle de vie de ces systèmes.
En outre, ces méthodes présentent une faiblesse liée au fait que l'amélioration repose sur une non prise en compte de l'attractivité pour les utilisateurs, conduisant à une optimisation orientée vers la réduction fonctionnelle (Figure 3) (difficilement acceptable pour le client).

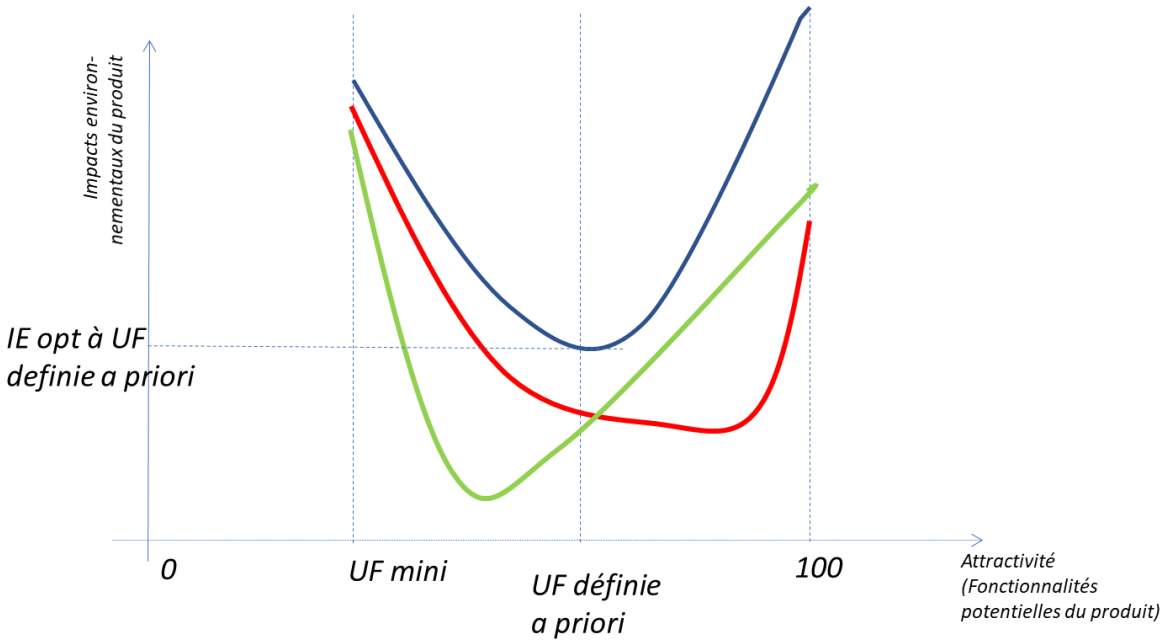

Figure 3 : Optimisation fonctionnelle

Il est de ce fait nécessaire de changer d'approche en considérant (1) que la complexité des systèmes à développer pour répondre à la crise environnementale actuelle nécessite une modélisation aussi exhaustive que possible du problème et des solutions techniques possibles, (2) que l'identification des solutions ne peut plus relever d'approches empiriques mais doit être basée sur un algorithme de recherche d'optima rendant compte d'un compromis entre performances d'usage et performances environnementales.

Pour éviter de longues et coûteuses itérations entre conception de systèmes et évaluation environnementale par ACV, nous avons la conviction de la nécessité d'un nouveau paradigme d'écoconception. Nous proposons d'hybrider un raisonnement CSP (Constraint Satisfaction Problem) à l'écoconception (cf. Figure 4).

1.2. Objectifs scientifiques et technologiques

Nous proposons dans ce projet de renverser le processus de raisonnement d'écoconception, en agrégeant rigoureusement dès le départ l'ensemble des contraintes associées aux impacts environnementaux pour préciser le problème de conception de manière systémique et pouvoir lancer un processus d'exploration systématique et informatisé des solutions satisfaisant tous les objectifs à la fois. Techniquement, cette nouvelle approche d'écoconception est basée sur une modélisation du problème par contraintes (CSP) et une résolution par une propagation des contraintes complétée par un algorithme d'optimisation. Cet outil de résolution par satisfaction de contraintes, bien établi scientifiquement aujourd'hui, comporte 2 temps :

- Une formulation par un ensemble de variables X = {x1, x2,..., xn}, chacune parcourant un domaine D= {d1, d2,..., dn}, tout en respectant un ensemble de contraintes C = {c1, c2,..., ck}. Résoudre un CSP revient à instancier chacune des variables de X dans le domaine D tout en satisfaisant l'ensemble C des contraintes du problème.
- Un algorithme d'exploration/optimisation permet ensuite de chercher le ou les optima parmi le champ restreint de solutions possibles.

L'hybridation d'une approche CSP et de l'écoconception est inédite. La Figure 4 présente l'originalité d'une structuration d'un problème d'écoconception en CSP vis-à-vis d'une approche classique.

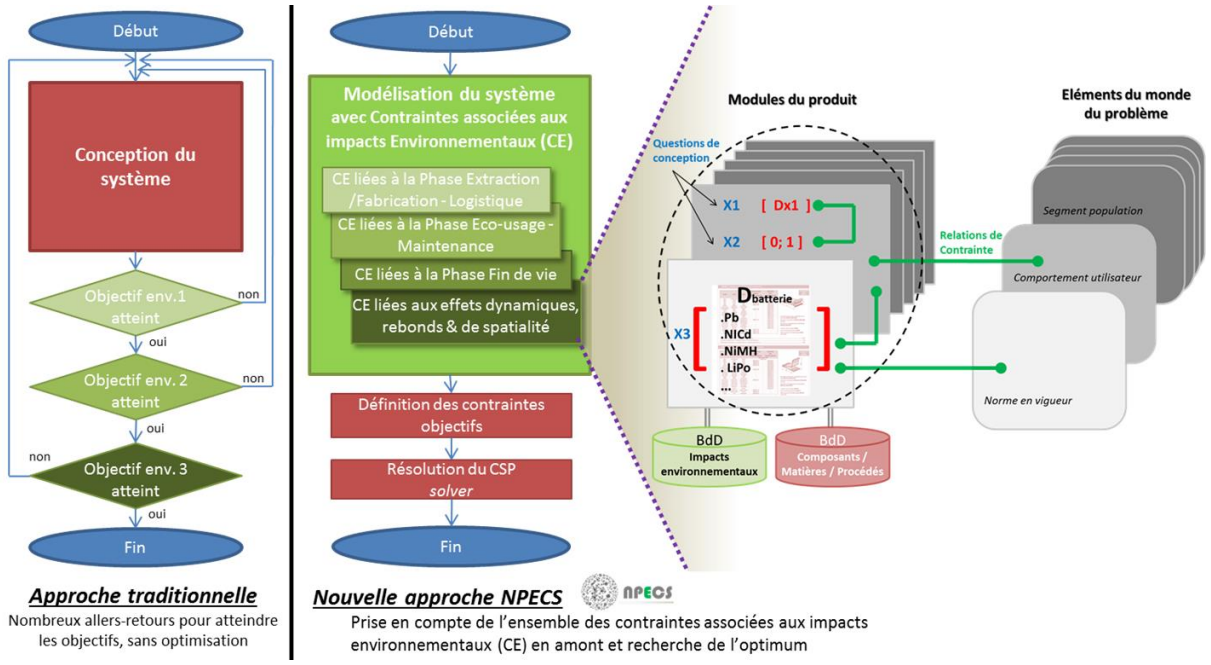

Figure 4 : Différences entre l'approche traditionnelle et la nouvelle approche NPECS

Les intérêts majeurs de l'approche avec un raisonnement CSP sont les suivants :

- L'approche par contraintes permet d'obtenir des solutions qui valident à coup sûr l'ensemble des objectifs fixés, ce qui n'est pas le cas des approches d'écoconception classiques où le respect des objectifs fixés est seulement vérifié au terme du processus de conception.
- L'approche systémique de la modélisation du problème de conception (le « monde » du problème et l'espace des solutions acceptables) rend réutilisable le modèle établi (les CSP sont des modèles déclaratifs acausaux permettant de facilement constituer des bibliothèques de modèles génériques et flexibles (inversion des entrées/sorties)). C'est une rupture vis-à-vis des processus d'écoconception conventionnels qui se déroulent de manière descendante vis-à-vis de spécifications initiales figées, ce qui a pour conséquence d'imposer une nouvelle itération de conception si les solutions ne sont pas satisfaisantes. Dans un raisonnement CSP, il devient aisé de modifier une spécification, de mettre en œuvre une étude de sensibilité, ou d'enrichir la modélisation ou d'en préciser une partie. Tout élément modélisé avec un niveau de granularité et un nombre de variantes possibles pourra être précisé et/ou complété de sorte que la modélisation globale s'enrichisse continuellement. Ainsi, il devient possible d'explorer en profondeur de nouveaux espaces de conception, en simulant un grand nombre d'alternatives de solutions dès la conception architecturale (choix et principes structuraux du système).
- L'approche systémique vise à modéliser les composantes du produit à concevoir mais aussi les éléments du problème à considérer, notamment l'ensemble des contraintes associées aux impacts environnementaux sur l'ensemble du cycle de vie. Il devient possible d'explorer différentes configurations de problèmes, ce qui augmente grandement la capacité d'éco-innovations. En effet, au-delà des modules/blocs fonctionnels qui constituent le produit, pourront être modélisés les phases de cycle de vie (éco-usage, éco-maintenance …), les parties prenantes du système (techniques, économiques, sociales …) et les configurations de problèmes (type d'usage, nombre de personnes…). Si le monde du problème se traduit par des contraintes de conception que l'on peut faire varier, c'est tout un pan d'explorations nouvelles qui peut être mené. Il devient possible de considérer différents comportements d'utilisateurs, différents segments de consommateurs, différents contextes d'usage, différents modes de consommation (PSS/ servicialisation/ mutualisation) … et donc plus largement de remettre en cause le niveau des performances fonctionnelles défini classiquement dans le cahier des charges. Cette « négociation fonctionnelle » est un levier pour améliorer les performances environnementales : faire varier certaines fonctions (négociables) et comparer les performances environnementales par rapport au

scénario actuel permet d'ajuster l'unité fonctionnelle « nominale », de ne pas reconduire les mêmes schémas et donc de décupler la capacité d'éco-innovations.

Le changement de paradigme de méthode d'écoconception proposé se traduit par des objectifs ambitieux, novateurs et porteurs d'un gisement important tant du point de vue de l'amélioration environnementale des systèmes que de la conception efficiente de systèmes complexes :

- Dépasser les limites de l'Analyse de Cycle de Vie telle qu'elle est pratiquée classiquement pour optimiser les impacts environnementaux lors de la conception de systèmes complexes. L'approche systémique intégrant la modélisation des contraintes associées aux impacts environnementaux (exemple pour la Phase de Fin de Vie illustré par la Figure 5) et l'exploration systématique via le raisonnement CSP permettent de travailler de manière proactive l'amélioration des impacts environnementaux, et de traiter des systèmes complexes.
- Révolutionner la considération de la fonctionnalité en conception pour optimiser le couple ajout de valeur et impacts environnementaux. En associant aux fonctions leurs contributions aux impacts environnementaux et en considérant l'évolution au cours de la vie du produit de la fonctionnalité (via les upgrades notamment), de nouvelles questions émergent dont le principal intérêt est la redéfinition de l'unité fonctionnelle. Le caractère systématique de l'exploration d'un raisonnement CSP associé à la définition de fonctions « sensibles/flexibles » permet de travailler sur l'optimisation des spécifications fonctionnelles d'un système (Cahier des charges) : c'est ce que nous appelons la « négociation fonctionnelle ». Cet axe de travail est particulièrement déterminant pour un système complexe.
- Démocratiser la pratique du raisonnement par contraintes pour le concepteur. Le but est de rendre actionnable et utilisable cette nouvelle génération de méthode de conception, qui nécessite un travail de modélisation spécifique. Le travail sur la fonctionnalité, notion bien connue du concepteur, est un des vecteurs possibles. En termes de processus d'obtention des solutions, il existe des solveurs plus appropriés pour des variables continues et d'autres pour des variables discrètes. Les problèmes d'écoconception faisant intervenir une hétérogénéité de variables, un des défis sera de les traiter de manière pertinente (adaptation ou hybridation de solveurs).

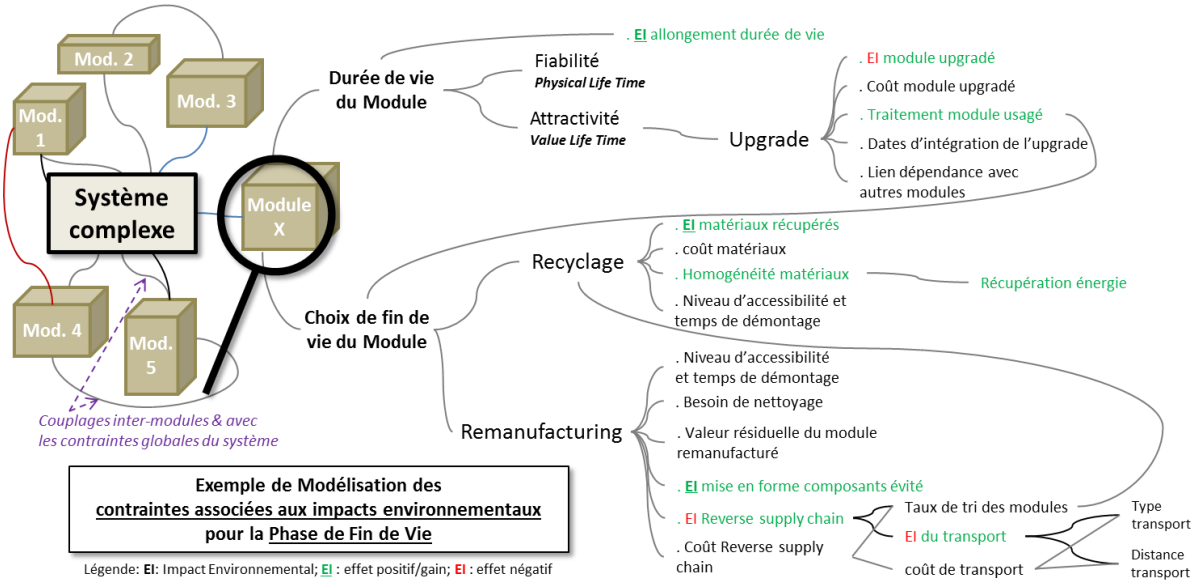

Figure 5 : Exemple de modélisation pour la phase de fin de vie

2. Méthodologie

2.1. Méthodologie d'écoconception

La méthodologie d'écoconception s'appuie sur une combinaison « Problème de satisfaction de contraintes » / ACV et s'articule en 6 phases consécutives (cf. flowchart Figure 6) [Tchertchian, 13].
Le besoin lié à la modélisation physique de système complexe en conception architecturale, consiste à pouvoir caractériser plusieurs configurations de conception les unes par rapport aux autres en termes de performances environnementales. Plus précisément, ce sont les valeurs des variables qui déterminent le système, qui constituent le besoin en conception. En conception préliminaire la satisfaction des contraintes numériques du problème de conception n'est pas facile, en effet les données dont disposent le concepteur sont souvent imprécises et les paramètres du système mal définis. Ces problèmes sont généralement sous-contraints, i.e., il existe un très grand nombre de solutions, voire un nombre infini si les domaines sont continus. Ce type de problème peut s'exprimer en termes de CSP [Montanary, 74].
Un CSP (Constraint Satisfaction Problem) est défini par :

- X = {x1, x2, x3, …, xn}, un ensemble de variables, *n* étant le nombre de variables du problème. Ces variables peuvent être de conception, de performance ou d'état. Les variables de conception structurent la conception et permettent de distinguer par leurs valeurs deux configurations de conception. C'est l'instanciation de toutes les variables de conception qui définit totalement une solution de conception potentielle. Les variables de performance permettent de traduire l'état ou la qualité d'une alternative de conception et de la comparer par rapport à une référence définie par le cahier des charges.
- D = {d1, d2, d3, …, dn}, un ensemble de domaines. Chaque domaine, associé à une variable, peut être discret ou continu.
- C = {c1, c2, c3, …, cp}, un ensemble de contraintes. Les contraintes traduisent comment les fonctions structurantes sont réalisées par le système pendant la situation de vie de référence. Les contraintes sont sous forme de relations explicites entre plusieurs variables.

Chacune des contraintes porte sur une partie de l'ensemble des variables dont elle restreint les domaines D. Résoudre un CSP revient à instancier chacune des variables de X dans son domaine de valeurs tout en satisfaisant l'ensemble des contraintes C du problème.

Au cours des deux dernières décennies, certains membres de la communauté des sciences de l'ingénieur ont utilisé cet outil dans le cadre de leurs recherches. Ils ont démontré que l'usage des CSP pouvait être intéressant dans les domaines tels que la préconception [Scaravetti, 04], la résolution de problèmes de dimensionnement [Yvars, 09], d'optimisation [Hao, 04], [Yvars, 10], de configuration de plateforme de produit [Xie, 05].

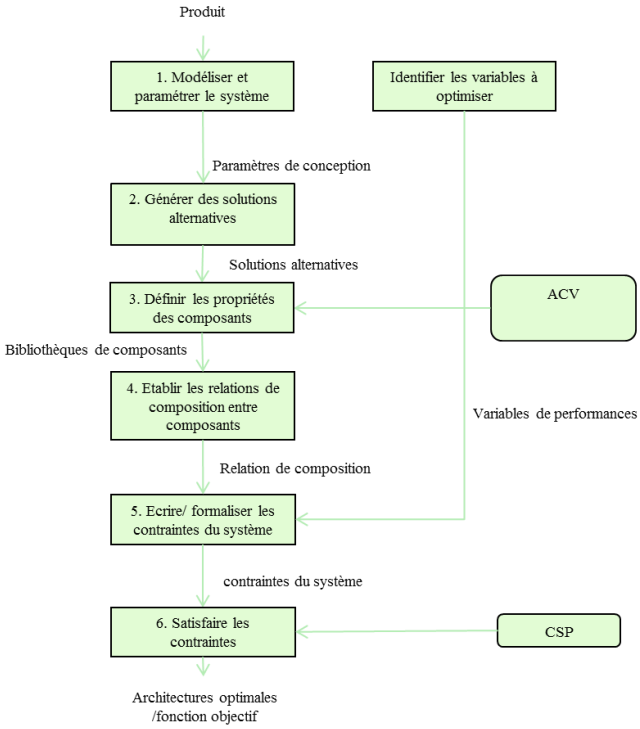

Figure 6 : Diagramme de la méthodologie mise en œuvre

La mise en œuvre de l'approche CSP/ LCA suit successivement les 6 phases :

- Modéliser et paramétrer le système et son cycle de vie et identifier les variables à optimiser.
- Pour chacun des composants, créer des bibliothèques pour définir l'ensemble des solutions disponibles pour le système (ça peut être des matériaux ou des composants). Une bibliothèque prend la forme d'une table de contrainte. Une table de contrainte est une contrainte globale qui représente les valeurs de combinaison possibles d'un jeu de variables de contrainte.
- Le processus de modélisation comprend la déclaration des variables de chaque sous-ensemble du système sur un intervalle fini ou un ensemble de valeurs discrètes. Pour l'évaluation environnementale il est nécessaire de déclarer des variables matériaux rendant compte de la composition en matériaux de chaque sous-ensemble ou composant. Par exemple : {ABS, PC, PP, PVC}
- Etablir les relations de composition entre composants. Un graphe de structure permet de modéliser les différentes configurations existantes. Il permet de décrire le produit par une décomposition de ses parties constituantes : en Ensembles, sous-ensembles, composants.
- Ecrire/ formaliser les contraintes du système. Une fois toutes les variables déclarées (phase 3) les contraintes sont ajoutées. Les contraintes sont des relations entre les variables (équations linéaires et non linéaires), appels de fonction, inéquations, tables de contraintes, par conséquent, en programmation par contraintes, les connexions entre les éléments du modèle sont faites en ajoutant les contraintes d'égalité entre les différentes variables du système.
- L'objectif est de déterminer l'architecture du système ainsi que son cycle de vie optimal par propagation des contraintes décrites dans la phase précédente.

2.2. Modélisation et résolution

Cette section décrit les éléments de modélisation mis en œuvre pour mener à bien la méthodologie décrite dans la section précédente. La modélisation permet de représenter une architecture de composants avec une modélisation et un choix spécifique à chacun d'eux. Ces choix concernent aussi bien l'étape de fin de vie du cycle de vie que la phase de conception. La phase d'utilisation n'est pas traitée.

2.2.1. Modélisation de la phase de conception

Cette partie de la modélisation vise à définir les valeurs de certaines variables de conception. Ces variables de conception ont un lien majeur avec les impacts environnementaux via la variation de la durée de vie de l'élément concerné. Elles correspondent par exemple aux choix de matériaux, aux dimensions de certains composants ou encore au nombre d'éléments constituants des composants.
L'analyse des bases de données environnementales permet d'établir des relations entre ces choix de conception et la durée de vie obtenue :

- Les composants dits passifs, comme les carter, ont comme variables de conception les épaisseurs de matériaux et le type de matériaux utilisés. Les autres variables dimensionnantes ne sont pas prises en compte comme le focus est mis sur l'écoconception et l'évaluation des impacts environnementaux. A partir de ces deux variables, la masse de matière est aisément obtenu et utilisée pour évaluer l'impact environnemental local à cet élément. Les analyses effectuées sur le cas applicatif ont permis d'établir une relation linéaire.
- Les composants dits actifs continus, comme les moteurs ou les cartes électroniques, ont comme variables de conception des variables continu permettant d'établir un lien avec leur durée de vie. Pour les moteurs, la puissance traduit d'une certaine manière la robustesse du composant et dans le cas d'un aspirateur peut impacter le temps d'usage. En effet, plus l'aspirateur est puissant, moins on devrait mettre de temps à aspirer, ce qui devrait accroitre la durée de vie du moteur.
- Les composants dits actifs discrets, comme les batteries, ont comme variables de conception des variables discrètes telles que le nombre de cellules dans une batterie ou encore le type de technologie des batteries (NiMH, NiCd, etc.).

2.2.2. Modélisation du scénario de fin de vie

Cette partie de la modélisation vise à choisir les scénarios de fin de vie des composants d'une architecture produit. Après avoir identifié les principales alternatives pour chacune des matières et/ou composants de l'architecture, des variables de décision continues sont utilisées pour formaliser ces choix comme des pourcentages associées à ces alternatives. Dans notre exemple sur l'aspirateur, nous considérons les 3 scénarios possibles que sont l'incinération, le recyclage et la réutilisation. Ainsi, nous avons trois fois plus de variables de décision que d'éléments pour lesquels il faut déterminer le scénario de fin de vie.
Cette modélisation permet de relier les scénarios de fin de vie à une évaluation de leur impact environnemental (IE). Pour se faire, les modèles de calcul utilisés reposent sur une définition massique des composants et de leurs matières constitutives. Le paramétrage de ces modèles est effectué à partir de bases de données environnementales servant de référence pour les données relatives aux matières et processus sur tout le cycle de vie.

La modélisation proposée permet de prendre en compte des décisions au niveau des assemblages, des composants et des matières. L'approche est systémique et une modélisation similaire est appliquée à ces trois échelles. Pour un élément X donné, nous définissons alors trois variables :

- X_inc dans l'intervalle [0,1] comme le pourcentage de x qui est incinéré
- X_rec dans l'intervalle [0,1] comme le pourcentage de x qui est recycler
- X_rem dans l'intervalle [0,1] comme le pourcentage de x qui est réutiliser ou remanufacturer.

A partir de ces 3 variables pour chaque élément de l'architecture nous assurons qu'aucune matière n'est perdue ou ajoutée avec la contrainte de conservation de la matière suivante : X_inc+X_rec+X_rem = 1

A l'échelle d'un composant ces trois variables sont une agrégation des variables des matières le composant. Pour une un composant A composé de N matières B1,…,BN, nous définissons les contraintes d'agrégation suivantes :

- A_inc = $(\sum_{i=1}^{N} \mathrm{A_Bi_m} * \mathrm{A_Bi_inc})/\mathrm{A_m}$
- A_rec = $(\sum_{i=1}^{N} \mathrm{A_Bi_m} * \mathrm{A_Bi_rec})/\mathrm{A_m}$
- A_rem = $(\sum_{i=1}^{N} \mathrm{A_Bi_m} * \mathrm{A_Bi_rem})/\mathrm{A_m}$

Les masses de matières A_Bi_m et la masse du composant A_m sont des paramètres dans cette partie de la modélisation. Les mêmes contraintes sont formulées au niveau des assemblages et de leurs composants.

L'évaluation des impacts environnementaux peut être simplement faite en agrégeant les masses au niveau des matières en fonction du scénario de fin de vie. Pour une matière B donnée constituant M composants A1,…,AM, nous pouvons formuler l'expression de l'impact environnemental comme une fonction linéaire de cette masse pour chaque scénario :

- IE_inc = coeff_B_inc*$\sum_{j=1}^{M} \mathrm{Aj_B_m} * \mathrm{Aj_B_inc}$
- IE_rec = coeff_B_rec*$\sum_{j=1}^{M} \mathrm{Aj_B_m} * \mathrm{Aj_B_inc}$
- IE_rem = coeff_B_rem*$\sum_{j=1}^{M} \mathrm{Aj_B_m} * \mathrm{Aj_B_inc}$

Les coefficients de ces fonctions linéaires sont issus de l'analyse des bases de données environnementales sur ces matières. Afin d'obtenir un impact environnemental global, les impacts précédents sont agrégés à chaque niveau systémique. Il est à noter, que la prise en compte du remanufacturing permet in fine de prendre en compte une réduction des impacts environnementaux sur l'élément concerné, puisqu'il évite la fabrication d'un élément neuf. La phase de remanufacturing génère malgré tout des impacts environnementaux (transport, reconditionnement, etc.).

2.2.3. Modélisation des variables d'écoconception

A partir de la modélisation locale des durées de vie de chacun des composants du système, l'agrégation à chaque niveau systémique se fait simplement en remontant la plus petite durée de vie des composants.
D'autres variables de performances peuvent être évaluées comme les coûts ou les prix, mais l'objectif est de mettre en lien la durée de vie avec les impacts environnementaux. Dans ce contexte, la fonction objectif à minimiser est l'impact environnemental moyen par an, soit la fonction :

$$f = IE/DV0,$$

où DV0 est la durée de vie agrégée au niveau du système et IE sont les impacts environnementaux exprimés dans la section 2.2.1 auxquels il faut rajouter les impacts environnementaux des étapes précédentes du cycle de vie. Ces derniers obtenus à l'aide de relations entre les variables de conception. Les coefficients de ces relations sont extraits

des bases de données environnementales. Ainsi, nous pouvons établir les IE environnementaux d'une matière ou d'un composant à partir de sa masse.
Dans la partie 2.2.1, nous avons modélisé la possibilité de remanufacturer un élément. Nous devons donc mettre ce choix en cohérence avec la durée de vie globale de l'élément par rapport à la durée de vie du module. Nous considérons donc la contrainte suivante pour un élément X dans un module S : $X_rem * X_DV \geq X_rem * coeff_DV * S_DV$.

2.2.4. Modélisation des variables d'écoconception

Cette partie de la modélisation vise à introduire les exigences des clients/ utilisateurs comme variables de décision dans les spécifications des variables de conception, correspondant par exemple aux choix de matériaux, aux dimensions de certains composants ou encore au nombre d'éléments constituants des composants. Les exigences que nous avons identifiées comme importantes pour les clients qui utilisent un aspirateur sont :

- La légèreté, caractérisée par la variable de performance Masse du système,
- L'exigence de rapidité de passage de l'aspirateur est caractérisée par la variable de performance Puissance du moteur
- L'exigence d'autonomie est caractérisée par la variable de performance Energie stockée de la batterie et de la Puissance du moteur.

Pour chacune des exigences un score d'attractivité est défini par une note de 0 à 100Pts (Figure 7) :

- 0 lorsque la variable de performance caractérisant l'exigence est minimale
- 100 lorsque la variable de performance caractérisant l'exigence est maximale

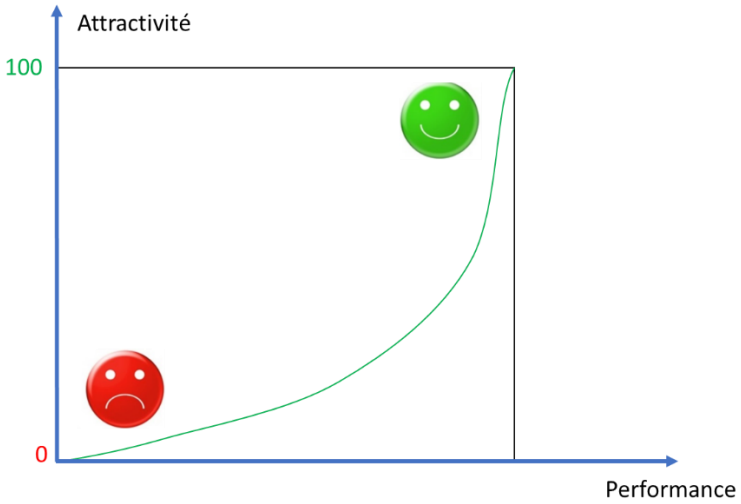

Figure 7 : Evaluation de l'attractivité

2.2.5. Résolution d'un modèle global

La résolution du modèle global revient à traiter dans un même modèle les variables de conception du produit et les variables de décision concernant les scénarios de fin avec toutes leurs contraintes associées. Afin d'obtenir la meilleure solution, la fonction objectif f décrite en 2.2.2 est minimisée. Le modèle obtenu est un modèle mixte non linéaire (MINLP). Le modèle peut être reformulé sous la forme d'un problème non-linéaire continu (NLP). De tels problèmes peuvent être traités à l'aide de solveur d'optimisation globale.
La structure de ce modèle global est présentée dans la Figure 8, qui se focalise sur un seul composant de l'architecture dont la variable de conception est l'épaisseur du matériau utilisé :

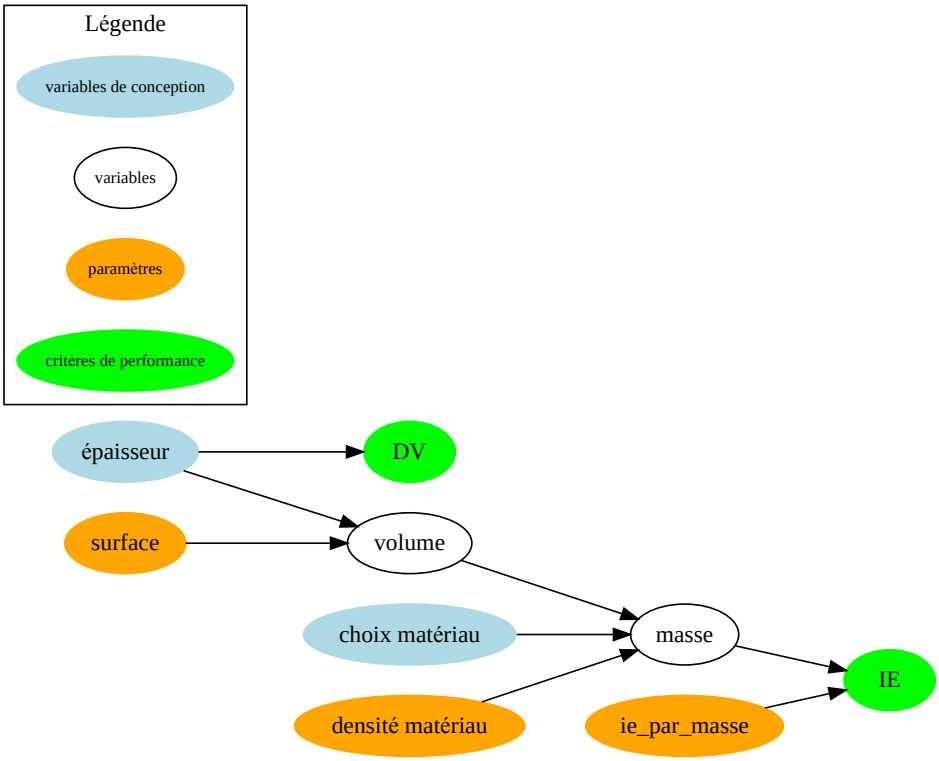

Figure 8: Schéma de la structure du modèle de calcul des impacts environnementaux par composant ou matière

En modélisant ainsi les différents composants de l'aspirateur (carters, moteur, batterie, carte électronique, etc.) nous obtenons un modèle permettant d'agréger et d'optimiser les impacts environnementaux à l'échelle globale du système.

3. Bilan / Principaux résultats obtenus

Cette section présente les résultats obtenus en mettant en œuvre la méthodologie décrite dans le paragraphe précédent. Deux applications complémentaires sont proposées utilisant deux solveurs différents, pour écoconcevoir un aspirateur. Dans la première application le focus est mis sur la recherche d'architecture de produit et sur l'attractivité des fonctions de l'aspirateur. Dans la seconde application ce sont les architectures de cycle de vie qui sont mis en avant.

3.1. Modélisation et recherche des Architectures de Produits

La méthode proposée a été appliquée à l'écoconception d'un aspirateur. Cet aspirateur a été modélisé de façon simplifiée comme étant un produit composé d'un carter de protection, d'un moteur pour l'aspiration, d'une batterie pour l'énergie et d'une carte électronique pour la partie commande. Nous avons ensuite modélisé chacun des composants à l'aide de contraintes afin d'explorer en profondeur de grand nombre d'alternatives de solutions. Par simplicité, les impacts environnementaux modélisés sont basés sur « l'Eco-Indicator » (Recipe) exprimé en milli-point (mPt).

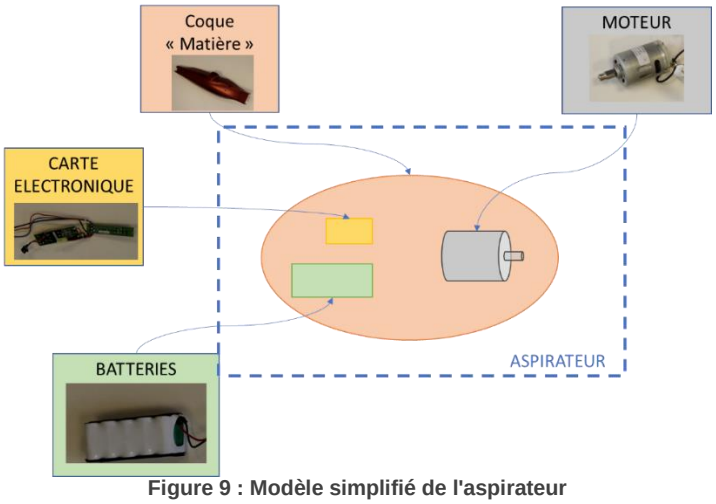

Figure 9 : Modèle simplifié de l'aspirateur

3.1.1. Description du modèle

- **Le module Carter de protection**

Le module « carter » est un module passif, l'impact environnemental du module est relié au type de matériau choisi et à la quantité (masse) spécifiée. Ces deux données constituent nos variables de conception. La durée de vie (DV_carter) est fonction de la masse de matériaux ou de l'épaisseur de la coque (Figure 10).

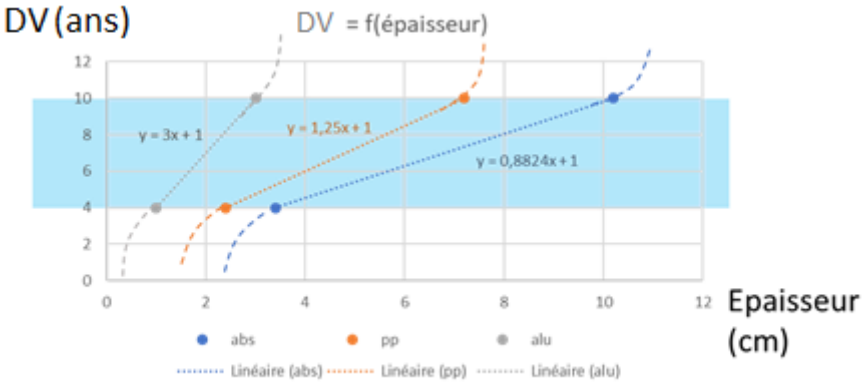

Figure 10 : caractérisation de la durée de vie d'un module passif

Le modèle composant, créé à l'aide des CSP, est constitué d'un tableau de matériaux possibles pour le module, de variables de conception (épaisseur, densité du matériau, résistance), de variables critères (durée de vie physique, masse, prix, impact environnemental) et de contraintes.

Tableau 1 : Tableau de Matériau

Type	Densité (Kg/ m3)	Prix (€/kg)	EI[3] (mPt/kg)	a
1 (ABS)	1060	2,1	223	3
2 (PP)	920	1,2	99,3	1,25
3 (Aluminium)	2700	1,2	782	0,8824

[3] Impact environnemental du matériau

Les variables de conception sont :

- Type de matériaux {ABS, PP, Aluminium}
- Épaisseur de la coque [1 ; 10] cm
- Surface à protéger 5000cm2
- DV = [4 ; 10] (ans)

Les contraintes sont :

(1) DV_{carter} **= a.** ***Epaisseur*** **+1**

(2) $EI_{carter} == M_{carter} \times EI_{mat}$

La contrainte **(1)** reliant la durée de vie à l'épaisseur est une hypothèse que nous avons faite afin d'illustrer notre démarche.

- **Le module Moteur**

Le module « Moteur » est un module actif car énergivore en phase d'utilisation. L'impact environnemental du module est relié à la masse du composant, aux matériaux utilisés et à la puissance à la phase d'utilisation. Ces trois données constituent nos variables de conception. La durée de vie « physique » du moteur dépend du type de technologie utilisé (moteur à balais ou sans balais) ainsi que de la puissance (Figure 11).

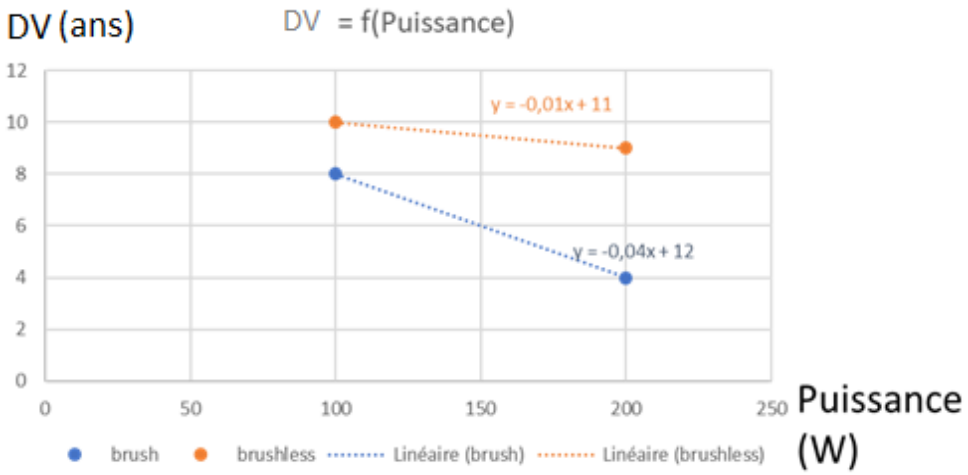

Figure 11 : Caractérisation de la durée de vie du module moteur

Le modèle composant, créé à l'aide des CSP, est constitué d'un tableau de composant possible pour le module, de variables de conception (type de technologie, Puissance, fiabilité), de variables critères (masse, prix, impact environnemental) et de contraintes.

Les variables de Conception

- Type de technologies {Brushless, Brushed}
- Puissance = [100 ; 200] Watts
- DV = [4 ; 10] ans

Tableau 2 : Tableau de Moteur

Type	Prix (€/W)	EI* (mPt/kg)	a	b	c	d
1 (brushed)	0,1	551	0,04	12	0,625	237,5
2 (brushless)	0,12	551	0,01	11	0,625	237,5

Les contraintes sont :

(3) $DV_{moteur} = - a.\ Puissance + b$
(4) $M = c.\ Puissance + d$
(5) $EI_{moteur} == M_{moteur} \times EI_{mat}$
(6) If type ==2 then add PCB

La contrainte (3) reliant la durée de vie à la puissance est une hypothèse que nous avons faite afin d'illustrer notre démarche.
La contrainte (4) *caractérise la masse en fonction de la puissance*
Une contrainte de type booléenne (6) doit être spécifier pour modéliser la carte de commande du moteur brushless.

- **Le module carte de commande (PCB)**

Le module « PCB » est un module actif continue car elle contient des composants électroniques énergivores en phase d'utilisation. L'impact environnemental du module est relié à surface de la carte, aux matériaux utilisés et aux composants électroniques qui la composent. Par hypothèse la durée de vie « physique » de la carte dépend du type de technologie utilisé et diminue lorsque le nombre de composants électroniques augmente (donc la surface de la carte augmente) Figure 12.

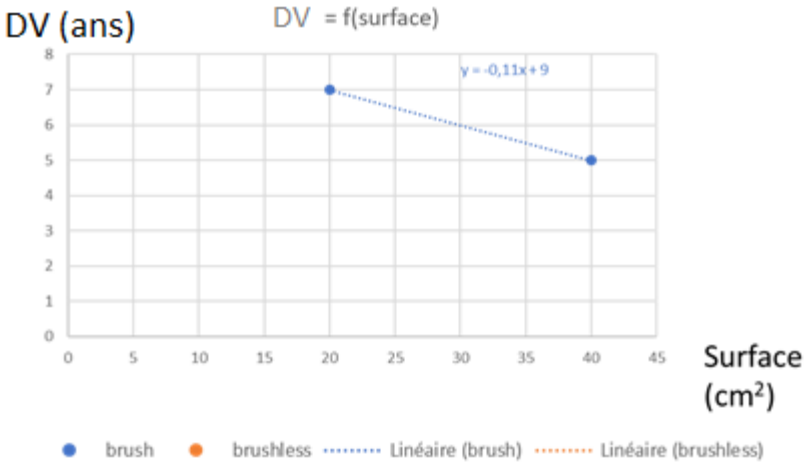

Figure 12 : Caractérisation de la durée de vie du module carte de commande

Les variables de Conception sont :

- Type de technologies {…}
- Surface (nombre de composants) = [10 ; 40] cm2
- Épaisseur = 0,2cm
- Carte de contrôle moteur « brushless » {0 ; 1}
- DV = [4 ; 10]

Tableau 3 : Tableau de PCB

Type	Densité (g/ cm3)	Prix (€/kg)	EI* (mPt/kg)	a	b
1	1,850	2	229	0,11	9
2	…				

Les contraintes sont :
(7) $DV_{PCB} = -a.\ Surface + b$
(8) $EI_{PCB} == M_{PCB} \times EI_{mat}$

La contrainte (7) reliant la durée de vie à la surface est une hypothèse qui stipule que la fiabilité de la carte diminue lorsque le nombre de composants électroniques augmente (donc avec la surface).

- **Le module batterie**

Le module batterie est un module actif discret. C'est d'abord un module actif car il permet le stockage de l'énergie pendant la phase d'utilisation. C'est aussi un module discret car la quantité d'énergie stockée dépend d'un nombre discret de cellules. Les batteries sont caractérisées par une énergie spécifique (Wh/kg) qui dicte l'autonomie de l'appareil. Une batterie est constituée de plusieurs cellules dont le format est défini par une tension (V) et une capacité (Ah). La durée de vie d'une batterie dépend du nombre de cycle de charge/ décharge Figure 13. Par exemple une batterie Lithium-ions ne possède plus, après 500cycles de charge/ décharge, que de 80% de sa capacité initiale.
L'impact environnemental du module dépend des matériaux utilisés (Nickel, Lithium, Plomb) et du nombre de cycle des cellules utilisées.

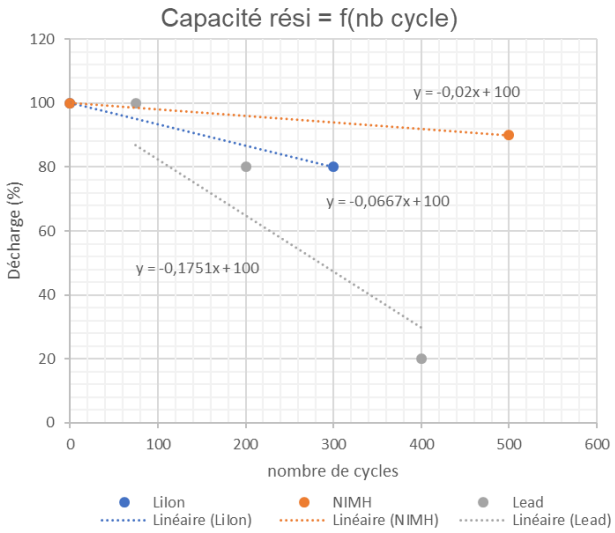

Figure 13 : Caractérisation de la durée de vie du module batterie

Les Variables de Conception sont :

- Type de technologies {Pb, Li-ion, NIMH}
- Nombre de cellules [0 ; 100]
- Capacité Energie stockée
- PLT = [4 ; 10]
- Nombre de cycles [100 ; 1500]

Tableau 4 : Tableau de batterie

Type	U (Volts)	Capacité (Ah)	Densité (Wh/kg)	a	EI (Pt/kg)	Prix (€/kWh)
1 (Li-ion)	3,6	2	170	0,067	1,1	780
2 (Ni-MH)	1,2	0,8	180	0,020	2,6	600
3 (Plomb)	6	1,3	31	0,175	4,2	130

Les contraintes sont :
(9) ***Capacité résiduelle = -a. Nb de cycle +100***
(10) $EI_{bat} == M_{bat} \times EI_{mat}$

La contrainte (9) *Capacité résiduelle = -a. Nb de cycle +100*(9) reliant la capacité résiduelle de la batterie au nombre de cycle est une hypothèse que nous avons faite afin d'illustrer notre démarche.

3.1.2. Evaluation de l'attractivité des fonctions

Les exigences que nous avons identifiées comme importantes pour les clients qui utilisent un aspirateur sont la légèreté, la rapidité de passage et l'autonomie.
Pour chacune des exigences un score d'attractivité est défini par une note de 0 à 100Pts.

Modélisation de l'attractivité de l'exigence « Légèreté »

La légèreté est quantifiée par la masse totale de l'aspirateur.
- La recherche de la configuration produit minimisant la masse donne un produit de 1,83 kg
- La recherche de la configuration produit maximisant la masse donne un produit de 4 kg

Par hypothèse nous considérerons qu'entre ces 2 valeurs l'attractivité varient linéairement selon la loi :

(11) ATT_masse = - 30.1 * Masse + 155.2

Modélisation de l'attractivité de l'exigence « rapidité de passage »

Pour un système d'aspiration fixé, le temps de passage est fonction de la puissance.
Par hypothèse le temps de passage est réduit de 10% tous les 100watts de puissance d'aspiration en plus.
- La recherche de la configuration produit minimisant le temps de passage donne un temps de passage de 0,27h (16min)
- La recherche de la configuration produit maximisant le temps de passage donne un temps de passage de 0,3h (18min)

Par hypothèse nous considérerons qu'entre ces 2 valeurs l'attractivité varie linéairement selon la loi :

(12) ATT_Puissance = -3333*durée + 1000

Modélisation de l'attractivité de l'exigence « autonomie »

L'autonomie est égale à l'énergie stockée par la batterie divisée par la puissance de l'aspirateur.

- La recherche de la configuration produit maximisant l'autonomie donne une autonomie de 0,41h (24,6min)
- La recherche de la configuration produit minimisant l'autonomie donne une autonomie de 0,32h (19,2min)

Par hypothèse nous considérerons qu'entre ces 2 valeurs l'attractivité varie linéairement selon la loi :

(13) ATT_autonomie = 1111*d_auto – 355

3.1.3. Résultats obtenus

La mise en œuvre de cet exemple a été réalisée en C++ à l'aide de la bibliothèque de programmation par contraintes (CP) IlogCP[4].
La plupart des bibliothèques CP sont basées sur un paradigme orienté objet. L'approche de modélisation est composée de trois étapes : tout d'abord, les classes doivent être créées et implémentées. Le carter, moteur, carte de contrôle et la batterie sont les classes dans notre exemple. Deuxièmement, les objets doivent être instanciés dans un modèle. Éventuellement, ces objets doivent être reliés entre eux. Par exemple un carter en abs rouge de 5mm d'épaisseur est une instance de la classe Carter.
En CP, la forme de déclaration des variables et des contraintes dépend fortement du langage et de l'environnement de développement. Néanmoins, le processus de modélisation reste toujours le même. Les variables sont déclarées avec

[4] https://www.ibm.com/fr-fr/analytics/cplex-cp-optimizer

un domaine ou un ensemble de valeurs défini, puis des contraintes sont ajoutées. Les contraintes sont des relations entre variables (équations linéaires et non linéaires), appels de fonctions, inéquations, tables de contraintes, etc.
Par conséquent, en CP, les connexions entre les objets du modèle sont réalisées en ajoutant des contraintes d'égalité entre les données membres des objets. Lorsque toutes les contraintes sont affichées, le solveur de contraintes est capable de les propager, c'est-à-dire de réduire le domaine des variables en fonction des contraintes. Il est également capable de rechercher des solutions disponibles dans les domaines réduits. Les modèles complets sont donnés en Annexes. Nous avons réalisé différentes simulations avec différents niveaux de granularité des modèles. Dans un premier temps nous avons recherché des « Architectures produits » sans fin de vie puis avec une fin de vie de base. Dans un second temps nous avons intégré l'attractivité dans la recherche d'architecture produits. Enfin nous avons recherché des architecture produit avec fin de vie améliorée et en intégrant l'attractivité.

- Scénario 1 : Sans fin de vie

Nous allons rechercher toutes les « architectures produits » qui satisfont les contraintes du modèle Aspirateur. Toutes les solutions (Figure 14) sont hiérarchisées selon le score environnemental annuel. Ce score est calculé suivant l'équation (14).

(14) ∑impacts / DV0

DV0 est la durée (minimale) du produit, correspond à la durée de vie du module le moins fiable. Dans ce scénario les impacts issus de la fin de vie ne sont pas pris en considération.
Le nombre de solutions possibles, respectant les contraintes, est de 9240.
L'architecture produit la plus impactante à un score (annuel) de 3.27Pts. La solution optimisée a un score de 0.73596pts.

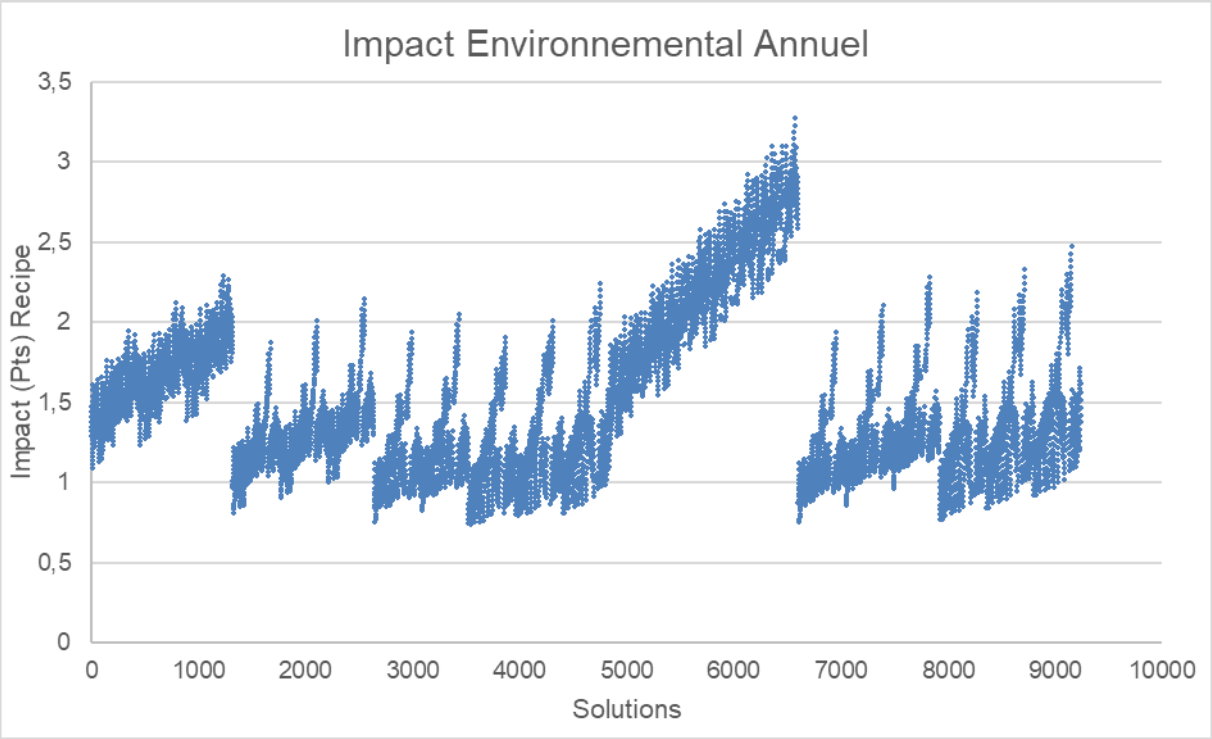

Figure 14 : Ensemble des solutions scénario 1 : Sans fin de vie

La solution optimale, Figure 15, a un impact environnemental annuel de 0.73596pts et une durée de vie DV0 de 7.9ans. L'aspirateur tombe en panne lorsque la carte électronique est défaillante (au bout de 7.9ans). Le carter choisi est en PP et pèse 3kg. Le moteur à une puissance de 116W de type « brushless ». La batterie est composée de 50 cellules NI-MH, a une durée de vie de 9.18ans correspondant à 1433 cycles et à une capacité de stockage de 48wh

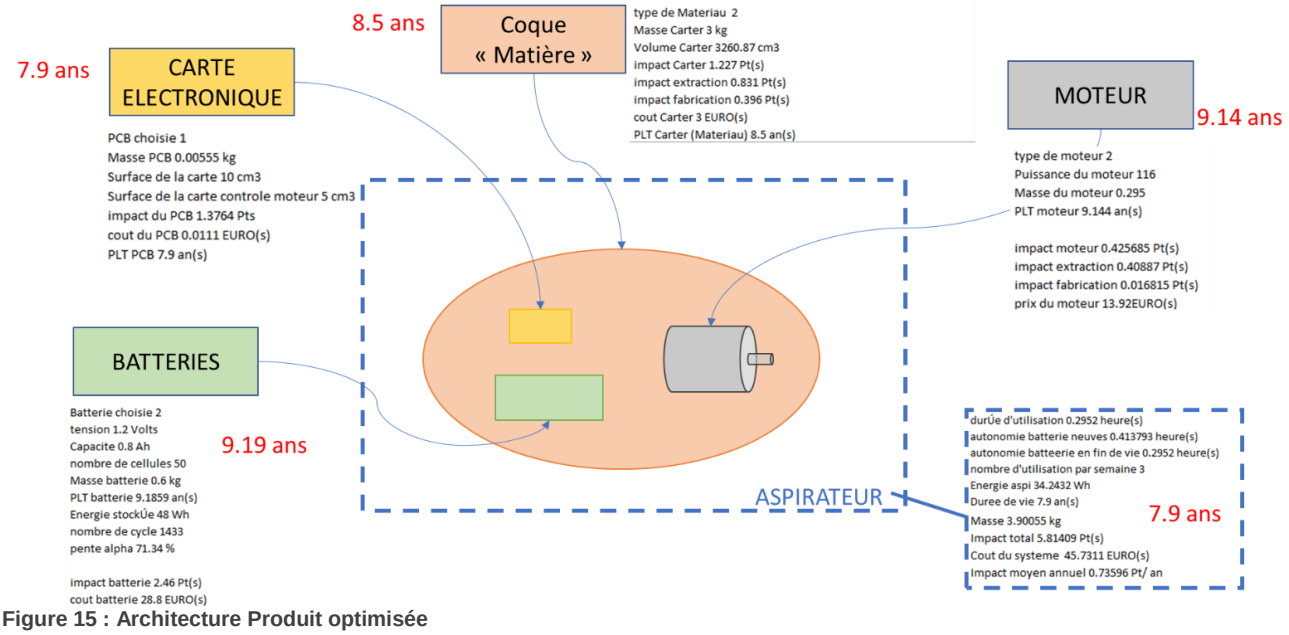

Figure 15 : Architecture Produit optimisée

Scénario 2 : Avec fin de vie de base

Nous allons rechercher toutes les « architectures produits » qui satisfont les contraintes du modèle Aspirateur. En prenant en compte la fin de vie, la durée de vie de l'aspirateur peut être allongée en changeant les composants qui ne fonctionnent plus par des modules fabriqués avec des matériaux recyclés, on définit alors un score environnemental annuel « étendu » calculé par l'équation (15).

(15) ∑impacts / DV_max

DV_max est la durée de vie maximale du produit, correspond à la durée de vie du module le plus fiable. Dans la suite on fixera une valeur (1000 ans par exemple) pour éviter les effets de bord. Toutes les solutions (Figure 16) sont hiérarchisées selon le score environnemental annuel.
La fin de vie de base consiste à recycler selon un taux de recyclage (t_rec) de 50% les matériaux plastiques, le reste étant incinérés, le moteur est recyclé à 50%, le reste est mise en décharge, les batteries sont placées en décharge à 100%, enfin les cartes de contrôle sont incinérées à 100%.

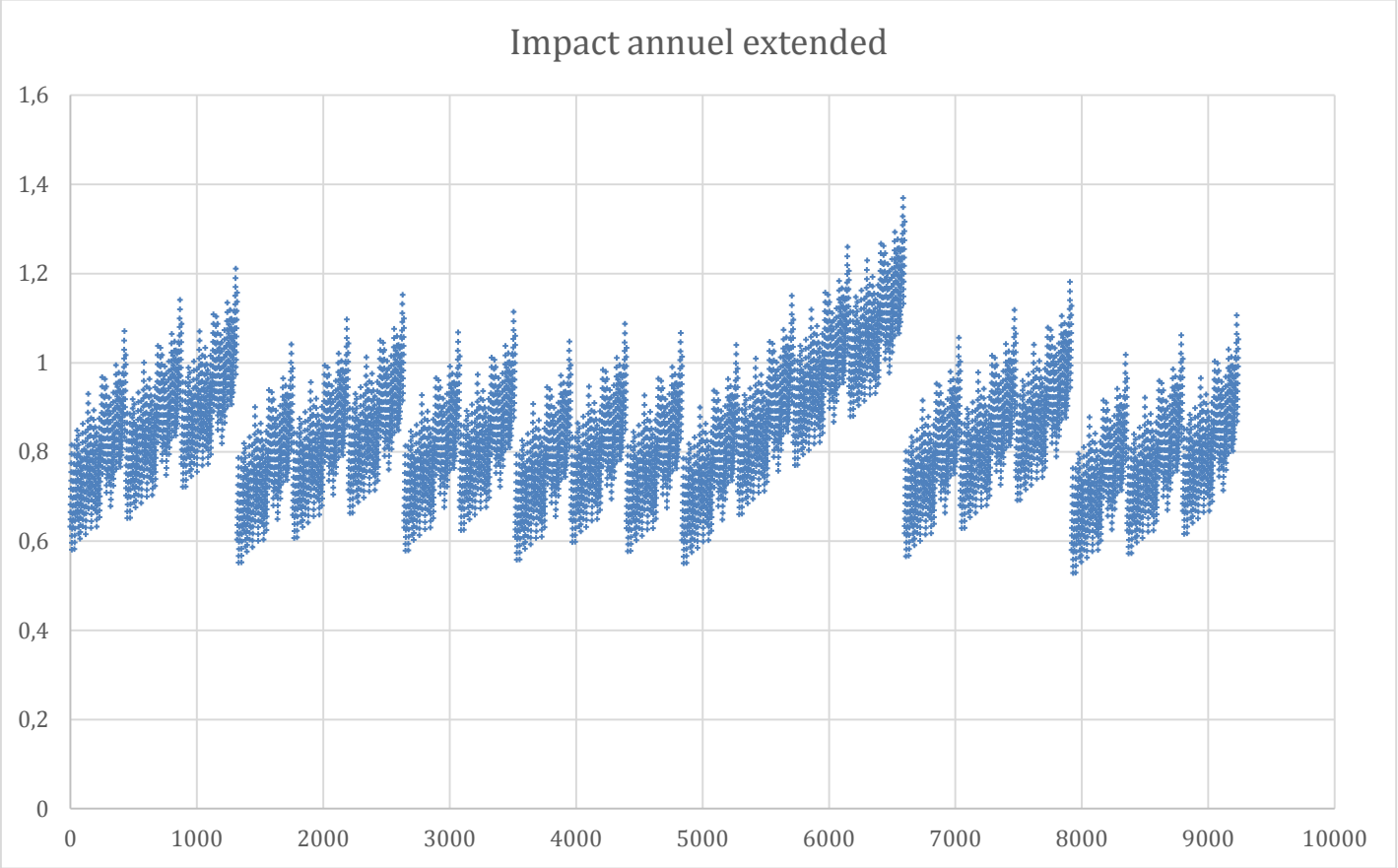

Figure 16 : Ensemble des solutions scénario 2 : fin de vie de base

Le nombre de solutions possibles, respectant les contraintes, est de 9240.
L'architecture produit la plus impactante à un score (annuel) de 1.37Pts. La solution optimisée a un score de 0.53pts.
La solution optimale, qui minimise le ratio impact/ durée de vie « étendue », a une durée de vie de 6.8ans. L'aspirateur tombe en panne lorsque le moteur de type « balais » d'une puissance de 108 Watt tombe en panne. Le carter choisi est en ABS et pèse 2kg. La batterie est composée de 45 cellules NI-MH, a une durée de vie de 8.2ans correspondant à 1280 cycles et à une capacité de stockage de 43.2wh

Scénario 3 : Avec fin de vie améliorée

Dans cette section le modèle a été modifié pour prendre en compte une fin de vie dite « améliorée ». Cette fin de vie améliorée permet de remanufacturer les modules moteur et carte de contrôle. On déclare une variable taux de remanufacturing (t_rem_X, X = module moteur ou carte de contrôle) pour chacun des modules avec les contraintes **(16)** et **(17)**

(16) t_rem_X = 0.8
(17) t_rem_X + t_recy_X <=1

De plus un impact environnemental est attribué au processus de remis en état, équivalent à 0.4kwh d'électricité.
Le nombre de solution évaluée est d'environ 100 000.

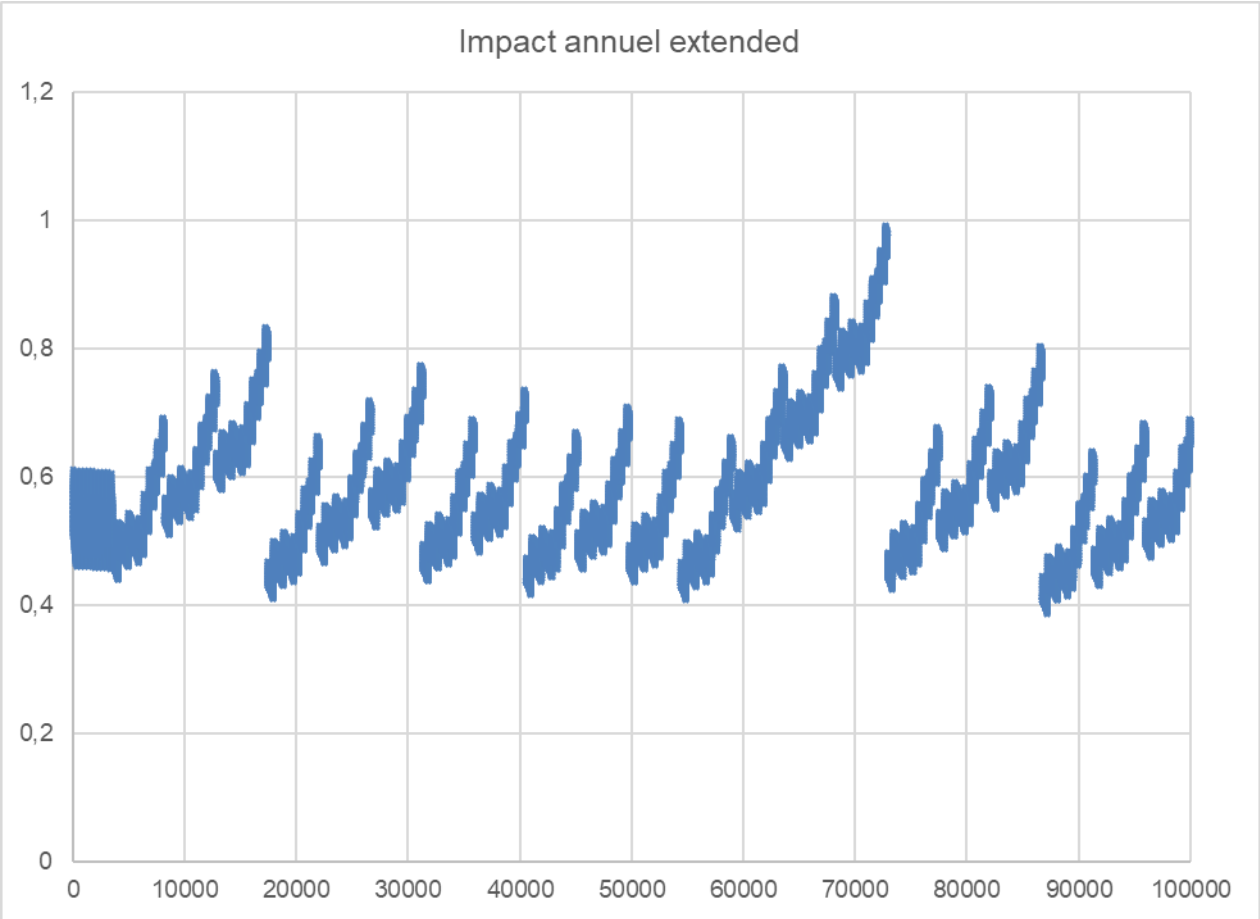

Figure 17 : Ensemble des solutions 3 : scénario fin de vie améliorée

Ce nombre important s'explique du fait que dans le modèle on fait varier le taux de recyclage de 0 à 20%, dans la suite on fixera le taux de recyclage à 20% pour le module moteur.
La stratégie de résolution utilisée pour minimiser l'impact environnemental conduit à effectuer 163 itérations avant d'obtenir la solution optimale.

La solution qui obtient le moins d'impact environnemental annuel étendue selon l'équation (15) est détaillée **Erreur ! Source du renvoi introuvable.Erreur ! Source du renvoi introuvable.**, avec un impact de 0.39 Pt/ an avec une durée de vie de 6.8ans. L'aspirateur tombe en panne lorsque le moteur de type « balais » d'une puissance de 108 Watt tombe en panne. Le carter choisi est en ABS et pèse 2kg. La batterie est composée de 45 cellules NI-MH, a une durée de vie de 8.2 ans correspondant à 1280 cycles et à une capacité de stockage de 43.2wh. En comparaison la solution optimale avec un scénario de fin de vie de base à un impact de 0.53 Pts/ an, soit une amélioration environnementale d'environ 26%. Toutefois l'ajout des contraintes du remanufacturing en fin de vie n'est pas venu modifier l'architecture produit présentée au scénario 2.

Tableau 5 : Comparaison des Architectures Produit

Variable	Scénario 1	Scénario 2	Scénario 3
Carter_1 matériaux	PP	ABS	ABS
Masse Carter (kg)	3	2	2
Durée de vie Carter (ans)	8.5	10	10
Surface PCB (cm2)	15	10	10
Durée de vie PCB (ans)	7.9	7.9	7.9
Type Batterie	NiMH	NiMH	NiMH
Energie stockée (Wh)	48	43.2	43.2
Durée de vie Batterie	9.2	8,2	8,2
Type Moteur	Brushless	Brushed	Brushed
Puissance Moteur (W)	116	108	108
Durée de vie Moteur (ans)	9.14	6.8	6.8
EI par an (mPts)	736.0	528.4	387,0

Lorsque l'on compare l'évaluation environnementale des 3 scénarios (Tableau 5), les architectures produit associées à un scénario de fin de vie (scénario 2 et 3) sont moins impactant que le scénario sans fin de vie car ils permettent d'allonger virtuellement la durée de vie (recyclage et remanufacturing).

3.1.4. Résultats obtenus avec prise en compte de l'attractivité

Dans cette partie nous avons évalué simultanément l'impact environnemental annuel « étendu » et l'attractivité des exigences utilisateurs (légèreté, puissance et autonomie). Nous sommes alors confrontés à un problème d'optimisation multi objectifs (impact ; attractivité). Pour chacune des simulations suivantes il est impossible, en toutes généralités, de trouver une solution unique. Il existe plutôt un ensemble de valeurs optimales, formant une frontière de Pareto. A l'aide d'un script c++, nous avons comparé chacune des solutions une à une afin de tracer le front de Pareto. Pour déterminer si une solution est meilleure qu'une autre, il convient de définir une notion de dominance :

Soit X et Y deux solutions. On dira que X domine Y si, pour tout objectif j = {impact ; attractivité}, on a

$$z^j(X) >= z^j(Y), \text{ avec au moins une inégalité stricte.}$$

La fonction « attractivité » à optimiser est la fonction d'agrégation (18)

(18) Att = 1/3 Att masse + 1/3 Att puissance + 1/3 Att autonomie

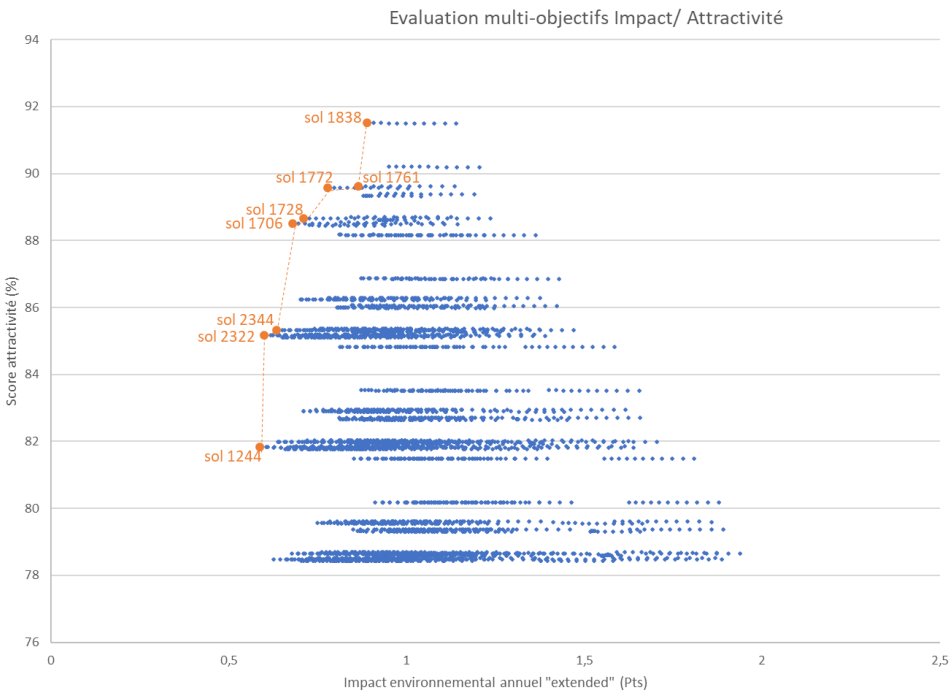

Figure 18 : Ensembles des solutions (Front de Pareto)

Parmi les 3200 solutions évaluées (Figure 18), les solutions 1244, 1706, 1728, 1761, 1772, 1838, 2322 et 2344 sont non dominées, elles forment donc l'ensemble des solutions du problème bi objectif.
Le Tableau 6 détaillent les solutions optimales. La solution 1706 est la plus légère, les solutions 1244 et 1706 ont la plus grande autonomie et la solution 1838 a la puissance d'aspiration la plus grande. La solution 1244 est la moins impactante environnementale et la solution 1838 est la plus attractive.

Tableau 6 : Ensemble des solutions du problème bi objectif.

Variable	Sol1244	Sol1706	Sol1728	Sol1761	Sol1772	Sol1838	Sol2322	Sol2344
Carter_1 matériaux	PP	ABS	ABS	ABS	ABS	ABS	ABS	ABS
Masse Carter (kg)	3	1	1	1	1	1	2	2
Durée de vie Carter (ans)	8.5	4	4	4	4	4	7	7
Surface PCB (cm2)	15	10	15	15	15	10	15	15
Durée de vie PCB (ans)	7.9	7.9	7.9	7.9	7.9	7.9	7.9	7.9
Type Batterie	NiMH	NiMH	NiMH	NiMH	NiMH	NiMH	NiMH	NiMH
Energie stockée (Wh)	43.2	43.2	47.0	53.8	53.8	66.2	43.2	47.0
Durée de vie Batterie	8.2	8,2	8	8	8	8	8.2	8
Type Moteur	Brushless	brushed	Brushless	Brushless	Brushless	brushed	Brushless	Brushless
Puissance Moteur (W)	108	108	120	140	140	180	108	120
Durée de vie Moteur (ans)	6.8	6.8	6.2	8.8	5.3	8.1	6.8	6.2
EI par an (mPts)	587.2	678.6	710.2	863.5	778.2	887.9	600.4	633.9
Attractivité (%)	81.8	88.5	88.7	89.6	89.6	91.5	85.2	85.3

La négociation fonctionnelle conduit à retenir les 8 solutions qui seront attractives pour les utilisateurs souhaitant un produit léger, un produit puissant pour nettoyer plus rapidement et un produit avec une grande autonomie. Dans notre application les 3 fonctionnalités sont données avec le même coefficient de pondération ce qui n'est pas forcément la réalité. Il serait nécessaire d'intégrer dans une approche méthodologique plus globale un outil de hiérarchisation des fonctions.

3.2. Modélisation et recherche des Architectures de Cycle de Vie

La méthode proposée a été appliquée à l'écoconception d'un aspirateur. Pour simplifier la description dans ce document, nous ne présentons que l'un des modules de l'aspirateur, mais l'approche est la même pour les autres modules et lorsque l'on remonte au niveau global de l'aspirateur. Dans le module décrit ci-dessous, nous avons 3 composants : 1 coque ou carter (composé de 2 parties qui s'assemblent), 1 carte électronique (pcb) et 1 batterie. Par simplicité, les impacts environnementaux modélisés sont basés sur l'Eco-Indicator RECIPE (EI) exprimé en milli-point (mPt).

3.2.1. Description des variables de conception

Nom de la variable	Domaine	Description
carter1_epaisseur	[0.001,0.01]	Epaisseur en m du carter 1
carter1_matériau	{ABS,PP,Aluminium}	Matériau du carter 1
carter2_epaisseur	[0.001,0.01]	Epaisseur en m du carter 2
carter2_matériau	{ABS,PP,Aluminium}	Matériau du carter 2
pcb_surface	[0.001,0.004]	Surface de la carte électronique en m2
batterie_type	{Pb,Li-ions,NiMH}	Type de la batterie
batterie_nb_cell	[0,100]	Nombre de cellules au sein de la batterie

3.2.2. Description des variables de décision pour le scénario de fin de vie

Nom de la variable	Domaine	Description
carter1_inc	[0,1]	Pourcentage du carter 1 incinéré
carter1_rec	[0,1]	Pourcentage du carter 1 recyclé
carter1_rem	[0,1]	Pourcentage du carter1 remanufacturé
carter2_inc	[0,1]	Pourcentage du carter 2 incinéré
carter2_rec	[0,1]	Pourcentage du carter 2 recyclé
carter2_rem	[0,1]	Pourcentage du carter 2 remanufacturé
pcb_inc	[0,1]	Pourcentage de la carte électronique incinéré
pcb_rec	[0,1]	Pourcentage de la carte électronique recyclé
pcb_rem	[0,1]	Pourcentage de la carte électronique remanufacturé
batterie_inc	[0,1]	Pourcentage de batterie incinéré
batterie_rec	[0,1]	Pourcentage de batterie recyclé
batterie_rem	[0,1]	Pourcentage de batterie remanufacturé

3.2.3. Description des variables de performance

Nom de la variable	Domaine	Description
carter1_IE	[0,1]	Impact environnemental du carter 1 en g/kgCO2
carter1_DV	[4,10]	Durée de vie du carter 1 en années
carter1_prix	[0.1,2]	Prix de revient du carter 1 en euros
carter2_IE	[0,1]	Impact environnemental du carter 2 en g/kgCO2
carter2_DV	[4,10]	Durée de vie du carter 2 en années
carter2_prix	[0.1,2]	Prix de revient du carter 2 en euros
pcb_IE	[0,1]	Impact environnemental de la carte électronique en g/kgCO2
pcb_DV	[4,10]	Durée de vie de la carte électronique en années
pcb_prix	[0.5,5]	Prix de revient de la carter électronique en euros
batterie_IE	[0,1]	Impact environnemental de la batterie en g/kgCO2
batterie_DV	[4,10]	Durée de vie de la batterie en années
batterie_nb_cycle	[1,1000]	Nombre de cycle de charge/décharge
batterie_energy	[0,1000]	Energie de la batterie à 100% de charge en Wh
batterie_prix	[7,20]	Prix de revient de la batterie en euros
module_IE	[0,1]	Impact environnemental global du module en g/kgCO2
module_DV	[4,10]	Durée de vie du module en année
module_prix	[10,25]	Prix de revient du module en euros

3.2.4. Paramètres utilisés[5]

Nom du paramètre	Valeur	Description
ABS_inc_EI	26,4	Impact environnemental de l'incinération d'ABS en mPt/kg
NiMH_inc_EI	314	Impact environnemental de l'incinération de NiMH en mPt/kg
…		
ABS_rec_EI	34	Impact environnemental du recyclage de l'ABS en mPt/kg
NiMH_rec_EI	248	Impact environnemental du recyclage du NiMH en mPt/kg
…		
ABS_fab_EI	223	Impact environnemental de l'extraction et de la fabrication d'un élément en ABS en mPt/kg
NiMH_fab_EI	2600	Impact environnemental de l'extraction et de la fabrication d'un élément en NiMH en mPt/kg
…		
ABS_densite	1,060	Densité de l'ABS en g/m3
PP_densite	0,92	Densité du PP en g/m3
Alu_densite	2,7	Densité de l'aluminium en g/m3
…		
carter1_surface	0,014	Surface du carter 1 en m2
carter2_surface	0,011	Surface du carter 2 en m2
…		
Transport_EI	0,00735	Impact environnemental du transport en kg.mPt/km
D_inc	10	Distance au centre d'incinération en km
D_rec	50	Distance au centre de recyclage en km
D_rem	500	Distance au centre de remanufacturing
Reduction_EI_rem	90	Pourcentage de réduction des impacts environnementaux des phases extraction et fabrication lors du remanufacturing
Duree_utilisation	0.25	Temps d'utilisation de l'aspirateur en minutes
P_ moteur	24	Puissance du moteur en W
Nb_cellule	10	Nombre de cellules de la batterie

[5] Certaines des valeurs utilisées sont fictives et traduises des tendances afin de simplifier la modélisation

3.2.5. Résultats obtenus

La résolution s'appuie sur le solveur IBEX[6] utilisant l'arithmétique intervalle et des mécanismes de programmation par contraintes pour implémenter un optimiseur global garanti. La résolution du modèle décrit dans ce document permet d'obtenir une architecture produit dimensionnée et les scénarios de fin de vie minimisant les impacts environnementaux tout en tenant compte de la durée de vie de l'architecture générée.

Afin de tester le modèle obtenu, nous avons défini plusieurs scénarios permettant d'illustrer sa cohérence et son intérêt pour l'écoconcepteur :

- **Scénario 1 :** les centres d'incinération, de recyclage et de remanufacturing sont tous proches (10 km),
- **Scénario 2 :** le centre d'incinération et de recyclage sont proches (10 et 50 km), le centre de remanufacturing est loin (500 km),
- **Scénario 3 :** le centre d'incinération est proche et les centres de recyclage et remanufacturing sont loin (200 et 1000 km),
- **Scénario 4 :** scénario 3 en minimisant le prix du module.

Les résultats ci-dessous ont été obtenu avec une précision absolue de 0.01 sur l'objectif. Ainsi, les résultats présentés sont garantis à cette précision près. Il est donc possible que d'autres solutions plus performantes soient trouvées, mais elles n'amélioreront pas l'objectif de plus de 1e-6.

Variable	Scénario 1	Scénario 2	Scénario 3	Scénario 4
Carter_1 matériaux	ABS	ABS	ABS	ABS
Carter_2 matériaux	ABS	ABS	ABS	ABS
Carter_1 épaisseur	0.0099	0.01	0,0099	0,0099
Carter_2 épaisseur	0.0099	0.0099	0,01	0,0099
Carter_1 fin de vie	Incinération	Incinération	Incinération	Incinération
Carter_2 fin de vie	Incinération	Incinération	Incinération	Incinération
Carter_1 durée de vie	4	4	4	4
Carter_2 durée de vie	4	4	4	4
PCB surface	0.001	0,001	0,001	0,001
PCB fin de vie	remanufacturing	remanufacturing	remanufacturing	Incinération
PCB durée de vie	9	9	9	9
Batterie type	NiMH	NiMH	NiMH	NiMH
Batterie fin de vie	remanufacturing	remanufacturing	remanufacturing	Incinération
Batterie durée de vie	7,2	8,28	8,75	8,75
Module durée de vie	4	4	4	4
Module EI	205,83	209,37	213,05	1621
EI99 par an	51,4	52,3	53,2	405,3
Prix du module	8,43	8,44	8,44	8,39

Pour l'ensemble des scénarios, l'ABS est choisi et incinéré, ce qui semble logique car il est plus impactant que le PP ou l'Aluminium et son recyclage impacte plus que son incinération. Le remanufacturing n'est pas choisi à cause de la durée de vie trop faible des carters obtenus. L'aluminium a un impact trop important par rapport à l'intérêt de doubler la durée de vie. Concernant le PCB, son recyclage est moins impactant que son incinération, mais c'est bien le remanufacturing qui est choisi comme cela évite la fabrication d'une autre carte. Malgré l'augmentation de la distance le remanufacturing l'emporte sur les 2 autres. Cela peut s'expliquer facilement du fait de la faible masse du PCB, ce qui génère des faibles impacts de transport. Concernant la batterie, l'impact est tel que seul l'optimisation du scénario 4 ne conduit pas au remanufacturing. Ce dernier scénario de fin de vie est préféré aux deux autres, car même si la batterie a une masse non négligeable (~420g), les impacts de fabrication d'une batterie sont tels que le transport n'est pas

[6] http://ibex-lib.org/

suffisamment pénalisant pour choisir un autre scénario de fin de vie. Le scénario 4 met clairement en évidence l'apport de notre approche pour réduire drastiquement les impacts environnementaux. Le prix n'est pas optimisé pour les trois premiers scénarios, mais nous pouvons voir qu'il n'est que faiblement impacté (0,5% d'accroissement) sur notre exemple en proportion du gain sur les impacts environnementaux (86% de réduction).

3.3. Synthèse des résultats

Les résultats obtenus dans les applications sont difficilement comparables car leurs objectifs étaient, au départ, complémentaires, l'une devait générer des architectures de produit et l'autre des architectures de cycles de vie. Finalement les deux applications ont un peu convergé (dans la deuxième application on génère aussi des architectures de produit) mais avec des hypothèses de départ différentes. Dans la première application le produit était constitué de 4 composants contre 3 dans l'autre application. Malgré cela et le fait que 2 solveurs différents étaient utilisés (ILOG et IBEX) des points de convergence existent ; le matériau des carters de protection est en ABS, la batterie en NiMH, le scénario de fin de vie le plus favorable est le remanufacturing. Cependant le score environnemental (EI) est sensiblement différent à cause du moteur qui est présent dans une seule des études.

4. Recommandations

Recommandations d'actions générales, recommandations pour l'ADEME, suites à donner…

La méthode que nous proposons est particulièrement interressante pour une utilisation dans les premières phases de la conception. Elle permettra d'évaluer une grande quantité de solutions potentielles pour guider le concepteur vers des architectures « Produit » et « Cycle de vie » prometteuses environnementallement, économiquement et attractive pour l'utilisateur. Cependant, l'approche telle qu'elle à été mise en œuvre n'est directement utilisable par les concepteurs sans un travail fastidieux de modélisation qiui impliquerait des ressources importantes, notemment informatique. C'est pourquoi nous préconisons, premièrement, une formalisation des contraintes significatives qui influent sur les impacts Environnementaux lors de la phase d'extraction-fabrication-logistique, lors de la phase d'éco-usage/maintenance et lors de la phase de fin de vie ; deuxièmement, l'élaboration d'une stratégie globale de modélisation par contraintes d'un problème d'écoconception de systèmes complexes et troisièmement, la construction d'un démonstrateur concret de la méthode développé et des algorithmes dédiées à la résolution des CSP pour l'écoconception sous la forme d'un prototype informatique.
Pour cela nous souhaitons soumettre un projet ANR afin d'investiguer ces différentes recommendations et d'aller plus loin.

5. Conclusions / Perspectives

Au cours de ce projet nous avons présenté une approche méthodologique qui repose sur une modélisation du problème par contraintes (CSP) et une résolution par une propagation des contraintes complétée par un algorithme d'optimisation.
Associé à l'outil d'analyse de cycle de vie, le raisonnement CSP permet, premièrement, une modélisation du produit/ système et des contraintes associées aux impacts environnementaux et deuxièmement une exploration systématique des « Architectures de Produits » et des « Architectures de Cycle de Vie » permettant de diminuer proactivement les impacts environnementaux, et de traiter des systèmes complexes. Les solutions obtenues dans le cadre des deux applications ne peuvent pas être appréhendé dans un processus de développement de produit traditionnel, cela nous conforte dans l'idée qu'il faut généraliser l'utilisation d'un outil numérique afin de générer et d'évaluer un maximum d'architectures couplé à des outils d'écoconception ceci pour ne pas passer à côté de solutions innovantes et environnementalement performantes.
Dans notre étude nous avons dissocié la génération des Architectures « Produit » et génération des Architectures « Cycle de vie ». Un algorithme spécifique de recherche d'optima sur cette double exploration architecturale doit être développé (Figure 19) afin de combiner les deux architectures.

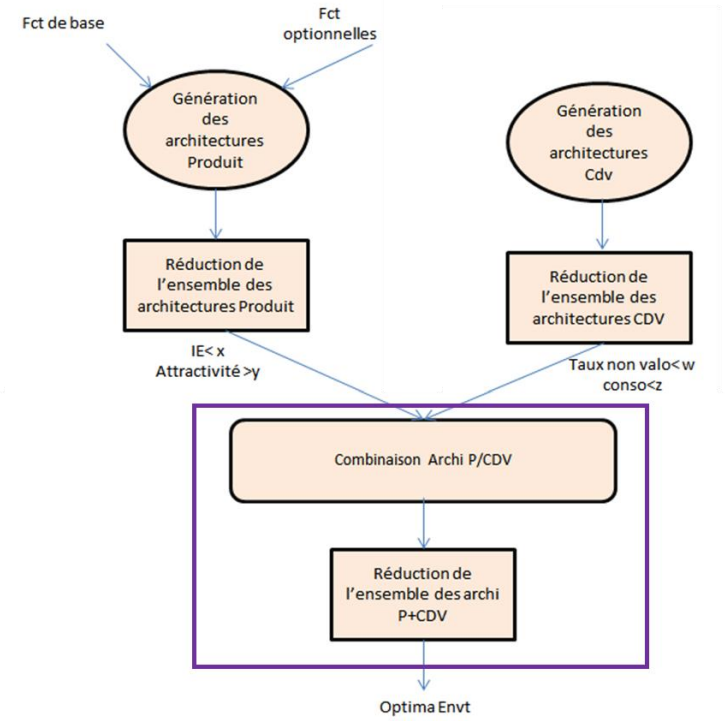

Figure 19 : Algorithme d'association des Architectures de produit et de Cycle de vie

De plus notre approche repose sur une prise en compte de l'attractivité des fonctions pour les utilisateurs par l'optimisation du couple ajout de valeur et impacts environnementaux. Alors que les analyses de cycle de vie raisonnent à unité fonctionnelle figée, notre approche amène une « négociation fonctionnelle » dans la spécification du cahier des charges.

Un des aspects non traités actuellement et la hiérarchisation des fonctions. En effet, pour pouvoir exprimer l'attractivité fonctionnelle comme une contrainte il est nécessaire de hiérarchiser les fonctions en donnant un poids aux différents paramètres fonctionnels du problème (Popoff, 17). A. Popoff décrit une méthode « ECOCSP-QFD » (Figure 20) qui associe raisonnement CSP et QFD pour générer un modèle optimisé d'un produit en début de conception. Dans la première phase de la méthode QFD les besoins client sont utilisés pour hiérarchiser les caractéristiques techniques du produit qui permettront de les satisfaire. Les scores de la matrice QFD obtenus sont ensuite utilisés pour pondérer les sous-objectifs de l'optimisation afin que la résolution prenne en compte les demandes des clients. En conséquence, une telle méthode permet une meilleure intégration des besoins du client au cours du processus de conception. Pour optimiser les performances multiples, les variables du modèle sont utilisées pour créer des fonctions d'impact correspondant aux impacts environnementaux et aux coûts de chaque phase du cycle de vie des composants du produit. En conséquence, le modèle optimisé est celui qui présente les meilleurs scores de performance tout en répondant au mieux aux besoins variés des clients.

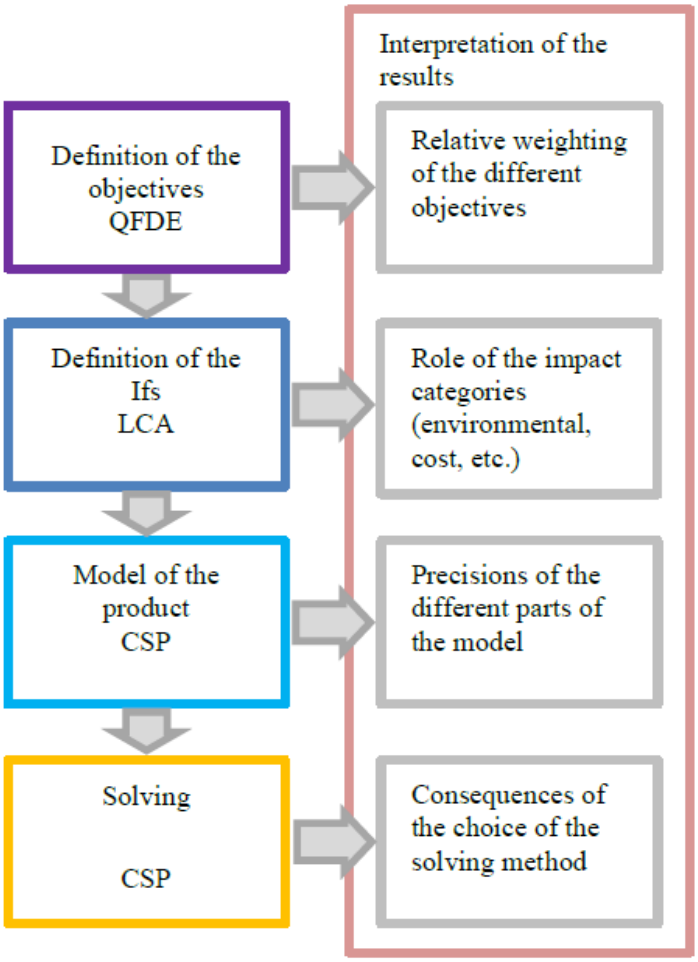

Figure 20 : Méthode ECOCSP- QFD

Références bibliographiques

Classer les références par ordre alphabétique des noms d'auteur ou par ordre d'apparition dans le texte. Il est vivement conseillé de les numéroter, avec renvoi depuis le texte vers la bibliographie.

[1] Azapagic A., Clift R., 1999. Life cycle assessment and multiobjective optimisation. Journal of Cleaner Production. 7 (2), 135-143.
[2] Hao F., Merlet J.P., Multi-criteria optimal design of parallel manipulators based on interval analysis, in Mechanism and Machine Theory, 2004
[3] LCA-ship, 2002. Design tool for energy efficient ships - A Life Cycle Analysis Program for Ships, Final report.
[4] Montanari U (1974) Networks of Constraints: Fundamental Properties and Applications to Picture Processing. Inform Science 7(2):95–132.
[5] Pahl, G. and Beitz, W. (1988), Engineering Design: A Systematic Approach, Springer-Verlag, Berlin.
[6] Popoff A., Vers une meilleure prise en compte de l'utilisation et des utilisateurs en écoconception : propositions d'éléments méthodologiques pour l'enrichissement d'une démarche d'optimisation amont. These de Doctorat Université de Toulon, 2017.
[7] Scaravetti D., Formalisation préalable d'un problème de conception, pour l'aide à la décision en conception préliminaire, These de Doctorat ENSAM, 2004.
[8] Schmidt WP, Dahlqvist E, Finkbeiner M, Krinke S, Lazzari S, Oschmann D, Pichon S, et al. (2004) Life Cycle Assessment of Lightweight and End-of-Life Scenarios for Generic Compact Class. The International Journal of Life Cycle Assessment 9(6):405 – 416.
[9] Tchertchian N., Yvars P-A., Millet D., Benefits and limits of a Constraint Satisfaction Problem/ Life Cycle Assessment approach for the ecodesign of complex systems. A case applied to a hybrid passenger ferry, Journal of Cleaner Production (JCP), Volume 42, 2013, Pages 1-18
[10] Unal, R., Stanley, D.O. & Joyner, C.R., 1993. Propulsion system design optimization using the Taguchi method. IEEE Transactions on Engineering Management, 40(3), pp.315-322.
[11] Van der Ploeg, A., 2011. A Comparison of Several Strategies to Optimize a Ship's Aft Body. Journal of Ship Production and Design. 27 (4), 202-211.
[12] Xie, H., Henderson, P. & Kernahan, M., 2005. Modelling and solving engineering product configuration problems by constraint satisfaction. International Journal of Production Research, 43(20), pp.4455-4469.
[13] Yvars, P.A., Lafon, P. & Zimmer, L., 2009. Optimization of mechanical system: Contribution of constraint satisfaction method. 2009 International Conference on Computers & Industrial Engineering, pp.1379-1384.
[14] Yvars, P.-A., 2010. A Constraint-Based Approach to the Composition Relation Management of a Product Class in Design. Journal of Computing and Information Science in Engineering, 10(3), 031002 (8 pages)

Index des tableaux et figures

Tableaux [i]

Tableau 1 : Tableau de Matériau 15
Tableau 2 : Tableau de Moteur 16
Tableau 3 : Tableau de PCB 17
Tableau 4 : Tableau de batterie 18
Tableau 5 : Comparaison des Architectures Produit 24
Tableau 6 : Ensemble des solutions du problème bi objectif. 25

Figures[i]

Figure 1 : complexité en écoconception 6
Figure 2 : Méthodes d'écoconception applicables aux systèmes complexes 6
Figure 3 : Optimisation fonctionnelle 7
Figure 4 : Différences entre l'approche traditionnelle et la nouvelle approche NPECS 8
Figure 5 : Exemple de modélisation pour la phase de fin de vie 9
Figure 6 : Diagramme de la méthodologie mise en œuvre 10
Figure 7 : Evaluation de l'attractivité 13
Figure 8: Schéma de la structure du modèle de calcul des impacts environnementaux par composant ou matière 14
Figure 9 : Modèle simplifié de l'aspirateur 15
Figure 10 : caractérisation de la durée de vie d'un module passif 15
Figure 11 : Caractérisation de la durée de vie du module moteur 16
Figure 12 : Caractérisation de la durée de vie du module carte de commande 17
Figure 13 : Caractérisation de la durée de vie du module batterie 18
Figure 14 : Ensemble des solutions scénario 1 : Sans fin de vie 20
Figure 15 : Architecture Produit optimisée 21
Figure 16 : Ensemble des solutions scénario 2 : fin de vie de base 22
Figure 17 : Ensemble des solutions 3 : scénario fin de vie améliorée 23
Figure 18 : Ensembles des solutions (Front de Pareto) 25
Figure 19 : Algorithme d'association architectures de produit et de cycle de vie 30
Figure 20 : Méthode ECOCSP- QFD 31

Sigles et acronymes

ADEME	Agence de l'Environnement et de la Maîtrise de l'Energie
AFPIA	Association pour la Formation Professionnelle dans les Industries de l'Ameublement

Annexes

Modélisation C++ librairies Ilog Solver

```
#ifndef CARTER_H
#define CARTER_H

#include <ilsolver/ilosolverint.h>
#include <ilsolver/ilosolverfloat.h>

class carter {
public :
carter(IloModel);

 IloNumVar n_carter;//nombre plastique
 IloIntVar M_carter;//masse plastique
 IloNumVar PLT_carter;// fiabilité du plastique
 IloNumVar impact_carter;// impact du module plastique
 IloNumVar impact_fab;// impact fabrication module plastique
 IloNumVar impact_ext;// impact extraction du module plastique
 IloNumVar impact_recy;// impact recyclage du module plastique
 IloNumVar impact_dech;// impact décharge du module plastique
 IloNumVar impact_inci;// impact incinération du module plastique
 IloNumVar r_carter;
 IloNumVar cout_carter;// prix du module plastique
 IloIntVar Surf_carter;
 IloNumVar Vol_carter;
 IloIntVar ep_carter;
 IloNumVar ep_carter_plus;
 //paramètres table plastique
 IloIntVar type_carter;// ABS; PS; PC
 IloIntVar IE_ext;// impact/ extraction kg mPt/kg
 IloIntVar IE_fab;//
 IloIntVar IE_inci;//
 IloIntVar IE_recy;
 IloIntVar IE_dech;
 IloIntVar prix_carter;// prix/ kg
 IloIntVar d_carter;// densité matériaux kg/m3
 //coeff courbe de tendance
 IloIntVar a_int_carter;//
 IloNumVar a_carter;
};
#endif
```

```
#include "carter.h"

carter::carter(IloModel mod)//constructeur
{
IloEnv env=mod.getEnv();
//////////////////////********** Carter plastique (module passif) *********/////////////////
 n_carter = IloNumVar(env, 1, 1000);//nombre plastique
 M_carter = IloIntVar(env, 1, 4);//masse plastique
 PLT_carter = IloNumVar(env,4,10);// fiabilité du plastique
 impact_carter = IloNumVar(env);// impact du module plastique
 impact_fab = IloNumVar(env);// impact fabrication module plastique
 impact_ext = IloNumVar(env);// impact extraction du module plastique
 impact_recy = IloNumVar(env);// impact recyclage du module plastique
 impact_dech = IloNumVar(env);// impact décharge du module plastique
 impact_inci = IloNumVar(env);// impact incinération du module plastique
//taux fin de vie
 r_carter = IloNumVar(env);
 cout_carter = IloNumVar(env);// impact du module plastique
 Surf_carter = IloIntVar(env);
 Vol_carter = IloNumVar(env);
 ep_carter = IloIntVar(env);
 ep_carter_plus = IloNumVar(env);
 //paramètres table materiaux
 type_carter = IloIntVar(env);// ABS; PS; PC
 IE_ext = IloIntVar(env);// impact extraction/ kg mPt/kg
 IE_fab = IloIntVar(env) ;//
 IE_inci = IloIntVar(env);//
 IE_recy = IloIntVar(env);
 IE_dech = IloIntVar(env);
 prix_carter = IloIntVar(env);// prix/ kg
 d_carter = IloIntVar(env);// densité matériaux kg/m3
 //coeff courbe de tendance
 a_int_carter = IloIntVar(env);//
 a_carter = IloNumVar(env);
 //Définition des contraintes du module plastique
 mod.add(a_int_carter == a_carter * 100);
 mod.add(Surf_carter == 5000);
 mod.add(PLT_carter == a_carter * ep_carter + 1);
 mod.add(impact_ext *1000 ==  M_carter* IE_ext);// Pt (multiplication par 1000 car IE en mPt)
mod.add(impact_fab *1000 ==  M_carter* IE_fab);// Pt
mod.add(impact_recy *1000 ==  M_carter* IE_recy);// Pt
mod.add(impact_dech *1000 ==  M_carter* IE_dech);// Pt
mod.add(impact_inci *1000 ==  M_carter* IE_inci);// Pt
mod.add(r_carter == 0.5);// H 50% du module est recyclé
mod.add(impact_carter  == n_carter* (impact_ext+ impact_fab-r_carter*(impact_ext - impact_recy)+(1-
r_carter)*(impact_dech+impact_inci) ));// Pt avec fin de vie on multiplie les impacts par le nombre de module a
fabriquer pour prolonger la vie du produit
mod.add(cout_carter ==  M_carter* prix_carter/10.);
mod.add(Vol_carter == Surf_carter*(ep_carter /10. + ep_carter_plus));
mod.add(M_carter == Vol_carter * d_carter /1000000.);
 //Tableau plastiques (carter matériaux)
IloIntVarArray tab_plas(env, 6, type_carter, d_carter, IE_ext,IE_fab, prix_carter, a_int_carter);
IloIntTupleSet tuple_plas(env, 6);
tuple_plas.add(IloIntArray(env, 6, 1, 1060, 460, 132, 21, 300));//ABS moulage par injection
tuple_plas.add(IloIntArray(env, 6, 2, 920, 277, 132,  12,  125));//PP moulage par injection
tuple_plas.add(IloIntArray(env, 6, 3, 2700, 629, 21,  12 , 88));//Alu emboutissage
mod.add(IloIfThen(env, type_carter ==1 , IE_recy ==7 && IE_dech ==0 && IE_inci ==148  ));// mPts /kg
mod.add(IloIfThen(env, type_carter ==2 , IE_recy ==7 && IE_dech ==0 && IE_inci ==119 ));// mPts /kg
mod.add(IloIfThen(env, type_carter ==3 , IE_recy ==11 && IE_dech ==31 && IE_inci ==0 ));// mPts /kg

mod.add(IloTableConstraint(env, tab_plas, tuple_plas, IloTrue));
}
```

```
#ifndef BATTERIE_H
#define BATTERIE_H
#include <ilsolver/ilosolverint.h>
#include <ilsolver/ilosolverfloat.h>

class batterie {
public :
batterie(IloModel);

//////////////////////********** Module Batterie *********/////////////////
 // Batterie (module actif)
 IloNumVar n_bat;//nombre plastique
 IloNumVar M_bat;//masse batterie
 IloNumVar PLT_bat;// fiabilité batterie
 IloNumVar impact_bat;// impact du module Batterie (extraction fabrication)
 IloNumVar impact_recy;// impact recyclage du module batterie
 IloNumVar impact_dech;// impact décharge du module batterie
 IloNumVar impact_inci;// impact incinération du module batterie
 IloNumVar cout_bat; //prix de la batterie
 IloNumVar E_stock;// Energie stockée par la batterie
 IloNumVar E_aspi;//energie nécessaire pour 1 utilisation du système
 IloIntVar nC; //nombre de cycle
 IloIntVar nCell; //nombre de cellule de la batterie
 IloNumVar duree; // duree de passage pour piece de 40m2 fonction de la puissance d'aspiration 100Watt min; duree = 0.3
- 0.1(p-100) d diminue de 10% tous les 100Watt
 IloIntVar d_int;
 IloNumVar d_auto;// autonomie de l'aspirateur MAX
 IloNumVar d_auto_min;// autonomie de l'aspirateur MIN
 IloInt nU;// nb d'utilisation par semaine
 //paramètres table batterie
 IloIntVar type_bat;//
 IloIntVar IE_bat;// impact/ kg*10
 IloIntVar IE_inci;//
 IloIntVar IE_recy;
 IloIntVar IE_dech;
 IloIntVar prix_bat;// prix/ kWh
 IloIntVar Unom_bat;// x10.Volt -> tension nominale d'une cellule
 IloIntVar C_bat;// capacité d'une cellule x10.Ah
 IloIntVar d_bat; //Wh/Kg
 IloIntVar a_bat; // pente décharge
 IloNumVar alpha;// capacité résiduelle (pourcentage de décharge en fonction du nombre de cycle)
};
#endif
```

```
#include "batterie.h"

batterie::batterie(IloModel mod)//constructeur
{
IloEnv env=mod.getEnv();
//////////////////////********** Module Batterie *********////////////////
n_bat = IloNumVar(env, 1, 1000);//nombre plastique
M_bat = IloNumVar(env, 0, 1);//masse batterie
PLT_bat = IloNumVar(env,4,10);// fiabilité batterie
impact_bat = IloNumVar(env);// impact du module Batterie
impact_recy = IloNumVar(env) ;// impact recyclage du module batterie
impact_dech = IloNumVar(env) ;// impact décharge du module batterie
impact_inci = IloNumVar(env) ;// impact incinération du module batterie
cout_bat = IloNumVar(env); //prix de la batterie
E_stock = IloNumVar(env);// Energie stockée par la batterie
E_aspi = IloNumVar(env);//energie nécessaire pour 1 utilisation du système
nC = IloIntVar(env, 1100, 1300); //nombre de cycle
nCell = IloIntVar(env, 40, 80); //nombre de cellule de la batterie
duree = IloNumVar(env, 0.2, 0.4); // duree de passage pour piece de 40m2 fonction de la puissance d'aspiration 100Watt
min; duree = 0.3 - 0.1(p-100) d diminue de 10% tous les 100Watt
d_int = IloIntVar(env);
d_auto = IloNumVar(env);// autonomie de l'aspirateur MAX
 d_auto_min = IloNumVar(env);// autonomie de l'aspirateur MIN
 nU = 3;// nb d'utilisation par semaine
 //paramètres table batterie
 type_bat = IloIntVar(env);//
 IE_bat = IloIntVar(env);// impact/ kg*10
 IE_inci = IloIntVar(env);//
 IE_recy = IloIntVar(env);
 IE_dech = IloIntVar(env);
 prix_bat = IloIntVar(env);// prix/ kWh
 Unom_bat = IloIntVar(env);// x10.Volt -> tension nominale d'une cellule
 C_bat = IloIntVar(env);// capacité d'une cellule x10.Ah
 d_bat = IloIntVar(env); //Wh/Kg
 a_bat = IloIntVar(env); // pente décharge
 alpha = IloNumVar(env);// capacité résiduelle (pourcentage de décharge en fonction du nombre de cycle)
 mod.add(alpha == 100 - a_bat/1000*nC);
 mod.add(M_bat == E_stock / d_bat);
 mod.add(E_stock == nCell * 0.1 *Unom_bat * 0.1* C_bat);  // unité Wh = Volts x Ampreres x heures
 mod.add(E_stock * alpha/100. == E_aspi);//on pourrait définir une Estock_eol =  E_stock * alpha/100
                                 //il faut donc Estock_eol >= E_aspi pour avoir suffisament d'énergie pour passer l'aspi
mod.add(PLT_bat*(nU*52) == nC);
 ///contraintes produit
//calcul de l'impact
mod.add(impact_dech == IE_dech* M_bat/1000.);// EI_dech en mpts dans le tableau
mod.add(impact_bat == n_bat*(IE_bat * M_bat/10 + impact_dech)); //EI -> x10 Pts/kg dans le tableau
mod.add(cout_bat == prix_bat * E_stock/1000.); //prix -> x1000 €/kg dans le tableau
 //définition BDD
 //Tableau batterie
IloIntVarArray tab_bat(env, 8, type_bat, Unom_bat, C_bat, d_bat, a_bat, IE_bat, IE_dech, prix_bat);
IloIntTupleSet tuple_bat(env, 8);
tuple_bat.add(IloIntArray(env, 8, 1, 36, 20, 170, 67, 22, 31, 180));//Li-ion 7
tuple_bat.add(IloIntArray(env, 8, 2, 12, 8, 80, 20, 41, 31, 600));//NIMH
tuple_bat.add(IloIntArray(env, 8, 3, 60, 13, 31, 175, 42, 31, 130));//Pb pour IE cf simapro base de données deesse

        // ajouter ici les autres matériaux du catalogue
        // ...
mod.add(IloTableConstraint(env, tab_bat, tuple_bat, IloTrue));
}
```

```
#ifndef MOTEUR_H
#define MOTEUR_H
#include <ilsolver/ilosolverint.h>
#include <ilsolver/ilosolverfloat.h>

class moteur {
public :
moteur(IloModel);

///parametres
 IloNumVar n_mot;//nombre plastique
 IloIntVar P;
 IloNumVar M_mot;//en kg
 IloNumVar PLT_mot;
 IloNumVar impact_mot;
 IloNumVar impact_fab;// impact fabrication module moteur
 IloNumVar impact_ext;// impact extraction du module moteur
 IloNumVar impact_recy;// impact recyclage du module moteur
 IloNumVar impact_dech;// impact décharge du module moteur
 IloNumVar impact_inci;// impact incinération du module moteur
 IloNumVar r_mot;
 IloNumVar cout_mot;
 IloNumVar surf_mot;// surface de carte electronique pour la commande du moteur brushless
 //paramètres table moteur
 IloIntVar type_mot;// aimant permanent / Brushless
 IloIntVar IE_ext;// impact/ kg (mPt)
 IloIntVar IE_fab;//
 IloIntVar IE_inci;//
 IloIntVar IE_recy;
 IloIntVar IE_dech;
 IloIntVar prix_mot;// prix/ Watt
 //coeff courbe de tendance
 IloIntVar a_int_mot;//
 IloIntVar b_int_mot;//
 IloIntVar c_int_mot;//
 IloIntVar d_int_mot;//
 IloNumVar a_mot;
 IloNumVar b_mot;
 IloNumVar c_mot;
 IloNumVar d_mot;
};
#endif
```

```
#include "moteur.h"

moteur::moteur(IloModel mod)//constructeur
{
IloEnv env=mod.getEnv();
//////////////////////********** Module Moteur (Actif) *********/////////////////
 ///parametres
n_mot = IloNumVar(env, 1, 1000);//nombre plastique
P = IloIntVar(env,100, 200);
M_mot = IloNumVar(env);//en kg
PLT_mot = IloNumVar(env,4,10);
impact_mot = IloNumVar(env);
impact_fab = IloNumVar(env);// impact fabrication module moteur
impact_ext = IloNumVar(env);// impact extraction du module moteur
impact_recy = IloNumVar(env);// impact recyclage du module moteur
impact_dech = IloNumVar(env);// impact décharge du module moteur
impact_inci = IloNumVar(env);// impact incinération du module moteur
cout_mot = IloNumVar(env);
surf_mot = IloNumVar(env);// surface de carte electronique pour la commande du moteur brushless
//paramètres table moteur
type_mot = IloIntVar(env);// aimant permanent / Brushless
IE_ext = IloIntVar(env);// impact/ kg (mPt)
 IE_fab = IloIntVar(env) ;//
 IE_inci = IloIntVar(env);//
 IE_recy = IloIntVar(env);
 IE_dech = IloIntVar(env);
 r_mot = IloNumVar(env);
 prix_mot = IloIntVar(env);// prix/ Watt
 //coeff courbe de tendance
 a_int_mot = IloIntVar(env);//
 b_int_mot = IloIntVar(env);//
 c_int_mot = IloIntVar(env);//
 d_int_mot = IloIntVar(env);//
 a_mot = IloNumVar(env);
 b_mot = IloNumVar(env);
 c_mot = IloNumVar(env);
 d_mot = IloNumVar (env);
 ///contraintes moteurs  ////
 mod.add(PLT_mot == - a_mot * P + b_mot);
 mod.add(M_mot*1000 == c_mot * P + d_mot);
 //mod.add(impact_mot ==  P * IE_mot);
 mod.add(impact_ext *1000 ==  M_mot * IE_ext);
 mod.add(impact_fab *1000 ==  M_mot * IE_fab);
 mod.add(impact_recy *1000 ==  M_mot* IE_recy);// Pt
 mod.add(impact_dech *1000 ==  M_mot* IE_dech);// Pt
 mod.add(impact_inci *1000 ==  M_mot* IE_inci);// Pt
//// calcul de l'impact
mod.add(r_mot == 0.5);// H 50% du module est recyclé
mod.add(impact_mot  == n_mot* (impact_ext+ impact_fab-r_mot*(impact_ext - impact_recy)+(1-
r_mot)*(impact_dech+impact_inci) ));// Pt avec fin de vie on multiplie les impacts par le nombre de module a fabriquer
pour prolonger la vie du produit
mod.add(cout_mot *100 == prix_mot * P);
mod.add(a_int_mot == a_mot * 1000);
mod.add(b_int_mot == b_mot );
mod.add(c_int_mot == c_mot*1000 );
mod.add(d_int_mot == d_mot );
mod.add(IloIfThen(env, type_mot == 1, surf_mot==0));
mod.add(IloIfThen(env, type_mot == 2, surf_mot==5));
//Tableau moteurs
IloIntVarArray tab_mot(env, 8, type_mot, IE_ext, IE_fab, prix_mot, a_int_mot, b_int_mot, c_int_mot, d_int_mot);
IloIntTupleSet tuple_mot(env, 8);
tuple_mot.add(IloIntArray(env, 8, 1, 1540, 63,  10, 48, 12, 625, 237));//aimant permanant
tuple_mot.add(IloIntArray(env, 8, 2, 1386, 57, 12,  16,  11, 500, 237));//brushless (hypothese IE_ext et fab -10%)
mod.add(IloTableConstraint(env, tab_mot, tuple_mot, IloTrue));
mod.add(IloIfThen(env, type_mot ==1 , IE_recy ==11 && IE_dech ==20 && IE_inci ==0  ));// mPts /kg
mod.add(IloIfThen(env, type_mot ==2 , IE_recy ==11 && IE_dech ==19 && IE_inci ==0 ));// mPts /kg  (hypothèse pas
d'aimant permanant)
}
```

```
#ifndef PCB_H
#define PCB_H

#include <ilsolver/ilosolverint.h>
#include <ilsolver/ilosolverfloat.h>

class pcb {
public :
pcb(IloModel);
IloNumVar n_pcb;//nombre plastique
IloNumVar M_pcb;//masse PCB
IloNumVar PLT_pcb;// fiabilité PCB
IloNumVar impact_pcb;// impact du module PCB (extraction et fabrication)
IloNumVar impact_recy;// impact recyclage du module PCB
IloNumVar impact_dech;// impact décharge du module PCB
IloNumVar impact_inci;// impact incinération du module PCB
IloNumVar cout_pcb; //prix de la batterie
IloIntVar surf_pcb;// surface de la carte
IloNumVar ep_pcb;// epaisseur de la carte
//paramètres table pcb
IloIntVar type_pcb;//
IloIntVar IE_pcb;// impact/ kg
IloIntVar IE_inci;//
IloIntVar IE_recy;
IloIntVar IE_dech;
IloIntVar prix_pcb;// prix/ kg
IloIntVar d_pcb;// // densité de la carte (resine epoxy FR4 1.85g/cm3) sans composants électroniques
                        //2€/kg (hypothèse)
IloIntVar a_pcb; // pente décharge
IloIntVar b_pcb; // pente décharge
};

#endif
```

```
#ifndef ASPIRATEUR_H
#define ASPIRATEUR_H
#include <ilsolver/ilosolverint.h>
#include <ilsolver/ilosolverfloat.h>

#include "batterie.h"
#include "moteur.h"
#include "pcb.h"
#include "carter.h"

class aspirateur {
public :
aspirateur(IloModel, carter*, moteur*, batterie*, pcb*);// constructeur

carter* ca;
moteur* mo;
batterie* ba;
pcb* pc;

IloNumVar IMP;// Impact du système
IloNumVar DV0;// Durée de vie du système
IloNumVar DV_max;// Durée de vie du système
IloNumVar IE_an;// Impact moyen annuel du système
IloNumVar IE_an_extended;// Impact moyen annuel du système
IloNumVar Masse;// Masse totale du système
IloNumVar COST; // cout du système
IloNumVar ATT_masse;
IloNumVar ATT_auto;
IloNumVar ATT_puissance;
IloNumVar ATT;
IloNum c_masse;
IloNum c_auto;
IloNum c_puissance;
};
#endif
```

```
#include "aspirateur.h"


aspirateur::aspirateur(IloModel mod, carter* ca, moteur* mo, batterie* ba, pcb* pc)//constructeur
{
IloEnv env=mod.getEnv();

IMP = IloNumVar(env);// Impact du système
DV0 = IloNumVar(env);// Durée de vie du système
DV_max = IloNumVar(env);// Durée de vie du système
IE_an = IloNumVar(env);// Impact moyen annuel du système
IE_an_extended = IloNumVar(env);// Impact moyen annuel du système
Masse = IloNumVar(env);// Masse totale du système
COST = IloNumVar(env); // cout du système
ATT_masse = IloNumVar (env);//Attractivité liée à la masse
ATT_auto = IloNumVar(env);// Attractivité liée à l'autonomie
ATT_puissance = IloNumVar(env);// Attractivité liée à la rapidité de passage d'aspiration
ATT = IloNumVar(env); // Attractivité du système //////////
c_masse = 1/3.;
c_auto = 1/3.;
c_puissance =1/3.;
mod.add(ba->duree == 0.3 - 0.3*0.1*(mo->P-100)/100.);
mod.add(IMP == ba->n_bat*ba->impact_bat + mo->n_mot*mo->impact_mot + ca->n_carter *ca->impact_carter + pc->n_pcb*pc-
>impact_pcb);// impact total
mod.add(Masse == ba->M_bat + mo->M_mot + ca->M_carter + pc->M_pcb);//
mod.add(COST == ba->cout_bat +  mo->cout_mot + ca->cout_carter + pc->cout_pcb);// cout total
// mod.add(DV0 == 10);
mod.add(DV0 == IloMin(IloMin(ca->PLT_carter , ba->PLT_bat), IloMin(mo->PLT_mot, pc->PLT_pcb)));// durée de vie du
produit
mod.add(DV_max == IloMax(IloMax(ca->PLT_carter , ba->PLT_bat), IloMax(mo->PLT_mot, pc->PLT_pcb)));// durée maximale à
atteindre (module le plus fiable)
//mod.add(DV_max == 1000);

mod.add(ca->n_carter == DV_max / ca->PLT_carter );
mod.add(mo->n_mot == DV_max / mo->PLT_mot);
mod.add(ba->n_bat == DV_max / ba->PLT_bat);
mod.add(pc->n_pcb == DV_max / pc->PLT_pcb);
mod.add(IE_an == IMP/DV0);
mod.add(IE_an_extended == IMP / DV_max);
//contraintes mixtes
mod.add(pc->M_pcb *1000== pc->d_pcb/100 * (pc->surf_pcb + mo->surf_mot) * pc->ep_pcb);//*1000 pour avoir des kg
mod.add(ba->E_aspi == mo->P* ba->duree); // Energie consommée par l'aspi
mod.add(ba->E_stock == mo->P* ba->d_auto);// autonomie maxi losque les batterie sont neuves
mod.add(ba->E_stock * ba->alpha / 100. == mo->P * ba->d_auto_min); // autonomie mini losque les batterie sont neuves
//////////////////////   Caracterisation de l'attractivité ////////////////////
////// Attractivité liée à la masse //////////
mod.add(ATT_masse == - 30.051 * Masse + 155.19);//100pts
////// Attractivité liée à l'autonomie //////////
mod.add(ATT_auto ==1111*d_auto - 355);//100Pts
////// Attractivité liée à la rapidité de passage d'aspiration //////////
mod.add(ATT_puissance ==   -3333*duree + 1000 );//100pts
////// Attractivité du système //////////
mod.add(3*ATT == c_masse * ATT_masse + c_puissance*ATT_puissance + c_auto * ATT_auto );
}
```

```
#include <iostream>
#include <fstream>
#include <ilsolver/ilosolverint.h>
#include <math.h>

#include "aspirateur.h"
#include "carter.h"
#include "moteur.h"
#include "batterie.h"
#include "pcb.h"

ILOSTLBEGIN

        int main(){
IloEnv env;                        //Construction de l'environnement
try {
IloModel mod(env);                //Création d'un modèle

carter* Ca= new carter(mod);
moteur* Mo= new moteur(mod);
batterie* Ba= new batterie(mod);
pcb* Pc= new pcb(mod);
aspirateur* Aspi= new aspirateur(mod, Ca, Mo, Ba, Pc);

//résolution
IloSolver solver(mod);
//fonction objectif
IloObjective I = IloMinimize(env, Aspi->IE_an);
IloObjective I_extended = IloMinimize(env, Aspi->IE_an_extended);
IloObjective Ma = IloMaximize(env, Aspi->Masse);
IloObjective Autonom = IloMinimize(env, Ba->d_auto);//E_stock);
IloObjective temps = IloMaximize(env, Ba->duree);
//IloObjective usage = IloMinimize(env, IE_usage);
//mod.add(I);
//mod.add(I_extended);
//mod.add(temps);
//mod.add(Ma);
//mod.add(Autonom);
//mod.add(usage);
ofstream fout("resultat.txt", ios::out | ios::app);//création d'un fichier d'enregistrement pour tracer front de pareto
ofstream fout2("resultat2.txt", ios::out | ios::app);//création d'un fichier d'enregistrement pour générer les
caractéristiques des solutions
//Propagation des contraintes
if(fout)//ecriture dans le fichier
 {
          if(fout2)
         {
//cout << "Propagation... "<< solver.propagate() << endl;

solver.startNewSearch();//
//
IloInt solutionCounter =0;

//Résolution
//if (solver.solve())
while (solver.next())
{
        solver.out() << "Solution " << ++ solutionCounter << ":\t" << "\n";

cout<<"******** configuration du Systeme **********"<<endl<<endl;
solver.out() << " durée d'utilisation " <<solver.getValue(Ba->duree)<<" heure(s) "<<endl;
solver.out() << " autonomie batterie neuves " <<solver.getValue(Ba->d_auto)<<" heure(s) "<<endl;
solver.out() << " autonomie batteerie en fin de vie " << solver.getValue(Ba->d_auto_min) << " heure(s) " << endl;
solver.out() << " nombre d'utilisation par semaine " <<Ba->nU<<endl;
solver.out() << " Energie aspi " <<solver.getValue(Ba->E_aspi)<<" Wh "<<endl;
solver.out() << " Duree de vie " <<solver.getValue(Aspi->DV0)<<" an(s) "<<endl;
solver.out() << " Duree de vie etendue " << solver.getValue(Aspi->DV_max) << " an(s) " << endl;
solver.out() << " Masse " <<solver.getValue(Aspi->Masse)<<" kg "<<endl;
solver.out() << " Impact total " <<solver.getValue(Aspi->IMP)<<" Pt(s) "<<endl;
```

```
solver.out() << " Cout du systeme  " <<solver.getValue(Aspi->COST)<<" EURO(s)"<<endl;
solver.out() << " Impact moyen annuel " <<solver.getValue(Aspi->IE_an)<<" Pt/ an "<<endl<<endl;
solver.out() << " Impact moyen annuel extended " << solver.getValue(Aspi->IE_an_extended) << " Pt/ an " << endl << endl;
cout << "*********quantité de composant pdt cycle de vie************" << endl << endl;
solver.out() << " nombre carter  " << solver.getValue(Ca->n_carter ) << endl;
solver.out() << " nombre moteur " << solver.getValue(Mo->n_mot) << endl;
solver.out() << " nombre batterie  " << solver.getValue(Ba->n_bat) << endl;
solver.out() << " nombre pcb  " << solver.getValue(Pc->n_pcb) << endl<<endl;
cout<<"********  Carter de protection **********"<<endl<<endl;
solver.out() << " type de Materiau  " <<solver.getValue(Ca->type_carter )<<endl;
solver.out() << " Masse Carter " <<solver.getValue(Ca->M_carter )<<" kg "<<endl;
solver.out() << " Volume Carter " <<solver.getValue(Ca->Vol_carter )<<" cm3 "<<endl;
solver.out() << " impact Carter " <<solver.getValue(Ca->impact_carter )<<" Pt(s) "<<endl;
solver.out() << " impact extraction " <<solver.getValue(Ca->impact_ext )<<" Pt(s) "<<endl;
solver.out() << " impact fabrication " <<solver.getValue(Ca->impact_fab )<<" Pt(s) "<<endl;
solver.out() << " impact décharge " <<solver.getValue(Ca->impact_dech )<<" Pt(s) "<<endl;
solver.out() << " impact incineration " <<solver.getValue(Ca->impact_inci )<<" Pt(s) "<<endl;
solver.out() << " impact recyclage " <<solver.getValue(Ca->impact_recy )<<" Pt(s) "<<endl;
solver.out() << " cout Carter " <<solver.getValue(Ca->cout_carter )<<" EURO(s) "<<endl;
solver.out() << " PLT Carter (Materiau) " <<solver.getValue(Ca->PLT_carter )<<" an(s) "<<endl<<endl;
solver.out() << " IE décharge " <<solver.getValue(Ca->IE_dech )<<" Pt(s) "<<endl;
solver.out() << " IE incineration " <<solver.getValue(Ca->IE_inci )<<" Pt(s) "<<endl;
solver.out() << " IE recyclage " <<solver.getValue(Ca->IE_recy )<<" Pt(s) "<<endl<<endl;
cout<<"********  Batterie **********"<<endl<<endl;
solver.out() << " Batterie choisie " <<solver.getValue(Ba->type_bat)<<endl;
solver.out() << " tension " <<solver.getValue(Ba->Unom_bat/10)<<" Volts "<<endl;
solver.out() << " Capacite " <<solver.getValue(Ba->C_bat/10)<<" Ah "<<endl;
solver.out() << " nombre de cellules " <<solver.getValue(Ba->nCell)<<endl;
solver.out() << " Masse batterie " <<solver.getValue(Ba->M_bat)<< " kg " <<endl;
solver.out() << " PLT batterie " <<solver.getValue(Ba->PLT_bat)<<" an(s) "<<endl;
solver.out() << " Energie stockée " <<solver.getValue(Ba->E_stock)<<" Wh "<<endl;
solver.out() << " nombre de cycle " <<solver.getValue(Ba->nC)<<endl;
solver.out() << " pente alpha " <<solver.getValue(Ba->alpha)<<" % "<<endl<<endl;
solver.out() << " impact batterie " <<solver.getValue(Ba->impact_bat)<<" Pt(s) "<<endl;
solver.out() << " impact decharge batterie " <<solver.getValue(Ba->impact_dech)<<" Pt(s) "<<endl;
solver.out() << " cout batterie " <<solver.getValue(Ba->cout_bat)<<" EURO(s) "<<endl;
cout<<"********  Moteur **********"<<endl<<endl;
solver.out() << " type de moteur " <<solver.getValue(Mo->type_mot)<<endl;
solver.out() << " Puissance du moteur " <<solver.getValue(Mo->P)<<endl;
solver.out() << " Masse du moteur " <<solver.getValue(Mo->M_mot)<<endl;
solver.out() << " PLT moteur " <<solver.getValue(Mo->PLT_mot)<<" an(s) "<<endl<<endl;
solver.out() << " impact moteur " <<solver.getValue(Mo->impact_mot)<<" Pt(s) "<<endl;
solver.out() << " impact extraction " <<solver.getValue(Mo->impact_ext)<<" Pt(s) "<<endl;
solver.out() << " impact fabrication " <<solver.getValue(Mo->impact_fab)<<" Pt(s) "<<endl;
solver.out() << " impact décharge " <<solver.getValue(Mo->impact_dech )<<" Pt(s) "<<endl;
solver.out() << " impact incineration " <<solver.getValue(Mo->impact_inci )<<" Pt(s) "<<endl;
solver.out() << " impact recyclage " <<solver.getValue(Mo->impact_recy )<<" Pt(s) "<<endl;
solver.out() << " prix du moteur " <<solver.getValue(Mo->cout_mot)<<"EURO(s)"<<endl<<endl;
solver.out() << " IE décharge " <<solver.getValue(Mo->IE_dech )<<" Pt(s) "<<endl;
solver.out() << " IE incineration " <<solver.getValue(Mo->IE_inci )<<" Pt(s) "<<endl;
solver.out() << " IE recyclage " <<solver.getValue(Mo->IE_recy )<<" Pt(s) "<<endl<<endl;
cout<<"********  Carte électronique **********"<<endl<<endl;
solver.out() << "PCB choisie " <<solver.getValue(Pc->type_pcb)<<endl;
solver.out() << "Masse PCB " <<solver.getValue(Pc->M_pcb)<<" kg "<<endl;
solver.out() << "Surface de la carte " <<solver.getValue(Pc->surf_pcb)<<" cm3 "<<endl;
solver.out() << "Surface de la carte controle moteur " <<solver.getValue(Mo->surf_mot)<<" cm3 "<<endl;
solver.out() << "impact du PCB " <<solver.getValue(Pc->impact_pcb)<<" Pts "<<endl;
solver.out() << "cout du PCB " <<solver.getValue(Pc->cout_pcb)<<" EURO(s) "<<endl;
solver.out() << "PLT PCB " <<solver.getValue(Pc->PLT_pcb)<<" an(s) "<<endl<<endl;
cout<<"******** Attractivité  **********"<<endl<<endl;
solver.out() << " Attractivité sur la masse du système " <<solver.getValue(Aspi->ATT_masse)<<" points "<<endl<<endl;
solver.out() << " Attractivité sur la puissance du système " <<solver.getValue(Aspi->ATT_puissance)<<" points
"<<endl<<endl;
solver.out() << " Attractivité sur l'auto du système " <<solver.getValue(Aspi->ATT_auto)<<" points "<<endl<<endl;
solver.out() << " Attractivité du système " <<solver.getValue(Aspi->ATT)<<" points "<<endl<<endl;

// ecriture fichier resultats pour Pareto (impact/ Attract)
fout<< solver.getInt(solutionCounter)<<""<<solver.getValue(Aspi->IE_an_extended)<<"     "<<solver.getValue(Aspi->ATT)<<"
        "<<endl;
```

```
//// ecriture fichiers resultats2 pour caractérisation des diférents paramètres
fout2<< "*********Solution " << solver.getInt(solutionCounter) << ":\t" << "************\n";
fout2<<"******** configuration du Systeme **********"<<endl<<endl;
fout2<< " durée d'utilisation " <<solver.getValue(Ba->duree)<<" heure(s) "<<endl;
fout2 << " autonomie batterie neuves " <<solver.getValue(Ba->d_auto)<<" heure(s) "<<endl;
fout2 << " autonomie batteerie en fin de vie " << solver.getValue(Ba->d_auto_min) << " heure(s) " << endl;
fout2 << " nombre d'utilisation par semaine " <<Ba->nU<<endl;
fout2<< " Energie aspi " <<solver.getValue(Ba->E_aspi)<<" Wh "<<endl;
fout2 << " Duree de vie " <<solver.getValue(Aspi->DV0)<<" an(s) "<<endl;
fout2 << " Duree de vie etendue " << solver.getValue(Aspi->DV_max) << " an(s) " << endl;
fout2 << " Masse " <<solver.getValue(Aspi->Masse)<<" kg "<<endl;
fout2 << " Impact total " <<solver.getValue(Aspi->IMP)<<" Pt(s) "<<endl;
fout2 << " Impact moyen annuel " <<solver.getValue(Aspi->IE_an)<<" Pt/ an "<<endl<<endl;
fout2 << " Impact moyen annuel extended " << solver.getValue(Aspi->IE_an_extended) << " Pt/ an " << endl << endl;
fout2 << "*********quantité de composant pdt cycle de vie************" << endl << endl;
fout2 << " nombre carter  " << solver.getValue(Ca->n_carter ) << endl;
fout2 << " nombre moteur " << solver.getValue(Mo->n_mot) << endl;
fout2 << " nombre batterie  " << solver.getValue(Ba->n_bat) << endl;
fout2 << " nombre pcb  " << solver.getValue(Pc->n_pcb) << endl<<endl;
fout2<<"********  Carter de protection **********"<<endl<<endl;
fout2 << " type de Materiau  " <<solver.getValue(Ca->type_carter )<<endl;
fout2 << " Masse Carter " <<solver.getValue(Ca->M_carter )<<" kg "<<endl;
fout2 << " impact Carter " <<solver.getValue(Ca->impact_carter )<<" Pt(s) "<<endl;
fout2 << " PLT Carter (Materiau) " <<solver.getValue(Ca->PLT_carter )<<" an(s) "<<endl<<endl;
fout2<<"********  Batterie **********"<<endl<<endl;
fout2 << " Batterie choisie " <<solver.getValue(Ba->type_bat)<<endl;
fout2 << " Energie stockée " <<solver.getValue(Ba->E_stock)<<" Wh "<<endl;
fout2 << " Masse batterie " <<solver.getValue(Ba->M_bat)<< " kg " <<endl;
fout2 << " PLT batterie " <<solver.getValue(Ba->PLT_bat)<<" an(s) "<<endl;
fout2  << " nombre de cycle " <<solver.getValue(Ba->nC)<<endl;
fout2 << " pente alpha " <<solver.getValue(Ba->alpha)<<" % "<<endl<<endl;
fout2 << " impact batterie " <<solver.getValue(Ba->impact_bat)<<" Pt(s) "<<endl;
fout2<<"********  Moteur **********"<<endl<<endl;
fout2 << " type de moteur " <<solver.getValue(Mo->type_mot)<<endl;
fout2 << " Puissance du moteur " <<solver.getValue(Mo->P)<<endl;
fout2 << " Masse du moteur " <<solver.getValue(Mo->M_mot)<<endl;
fout2 << " PLT moteur " <<solver.getValue(Mo->PLT_mot)<<" an(s) "<<endl<<endl;
fout2 << " impact moteur " <<solver.getValue(Mo->impact_mot)<<" Pt(s) "<<endl;
fout2<<"********  Carte électronique **********"<<endl<<endl;
fout2 << "PCB choisie " <<solver.getValue(Pc->type_pcb)<<endl;
fout2 << "Masse PCB " <<solver.getValue(Pc->M_pcb)<<" kg "<<endl;
fout2 << "Surface de la carte " <<solver.getValue(Pc->surf_pcb)<<" cm3 "<<endl;
fout2 << "Surface de la carte controle moteur " <<solver.getValue(Mo->surf_mot)<<" cm3 "<<endl;
fout2 << "impact du PCB " <<solver.getValue(Pc->impact_pcb)<<" Pts "<<endl;
fout2 << "PLT PCB " <<solver.getValue(Pc->PLT_pcb)<<" an(s) "<<endl<<endl;
}
solver.endSearch();

solver.printInformation(); //Affichage des informations sur la solution
fout.close();//fermeture du fichier d'enregistrement
}//
fout2.close();//fermeture du fichier d'enregistrement résultats2
}//
}
catch (IloException& ex) {
cout << "Error: " << ex << endl;
}
env.end();//Destruction de l'environnement
getchar();//
return 0;
}
```

Modélisation IBEX

constants
ABS_inc_EI99 = 26.4; // mPt/kg
PP_inc_EI99 = 24.7; // mPt/kg
Alu_inc_EI99 = 182; //mPt/kg
NiMH_inc_EI99 = 314; // mPt/kg
Pb_inc_EI99 = 1042; // mPt/kg
LiIon_inc_EI99 = 501; // mPt/kg

ABS_rec_EI99 = 34; // mPt/kg
PP_rec_EI99 = 31; // mPt/kg
Alu_rec_EI99 = 82; //mPt/kg
NiMH_rec_EI99 = 248; // mPt/kg
Pb_rec_EI99 = 220; // mPt/kg
LiIon_rec_EI99 = 101; // mPt/kg

ABS_fab_EI99 = 223; //mPt/kg
PP_fab_EI99 = 99.3; //mPt/kg
Alu_fab_EI99 = 782; //mPt/kg
NiMH_fab_EI99 = 2600; // mPt/kg
Pb_fab_EI99 = 4200; // mPt/kg
LiIon_fab_EI99 = 1100; // mPt/kg
rem_EI99 = 0.9; // % de X_fab_EI99 à retirer si remanufacturing

ABS_densite = 1060; // kg/m3
PP_densite = 920; // kg/m3
Alu_densite = 2700; // kg/m3

ABS_prix = 2.1; // €/kg
PP_prix = 1.2;
Alu_prix = 1.2;
NiMH_prix = 600; // €/kWh
LiIon_prix = 780;
Pb_prix = 130;

carter1_surface = 0.014; // m2
carter2_surface = 0.011; // m2

pcb_epaisseur = 0.002; // m
pcb_densite = 1850; // kg/m3
pcb_masse_EI99 = 229; // mPt/kg
pcb_inc_EI99 = 12.3;
pcb_rec_EI99 = 8.3;
pcb_fab_EI99 = 59.3;
pcb_prix = 2; // €/kg

batterie_Pb_capa = 1.3; // Ah
batterie_LiIon_capa = 2; // Ah
batterie_NiMH_capa = 0.8; // Ah
batterie_Pb_U = 6; // V
batterie_LiIon_U = 3.6; // V
batterie_NiMH_U = 1.2; // V
batterie_Pb_densite = 31; // Wh/kg
batterie_LiIon_densite = 170;
batterie_NiMH_densite = 180;
batterie_Pb_viellissement = 1.4; // coeff de vieillissement de la batterie
batterie_LiIon_viellissement = 1.2;
batterie_NiMH_viellissement = 1.05;
batterie_nb_cell = 10;
duree = 0.25; // h
puissance_moteur = 24; // W

transport_EI99 = 7.35e-3 ; // kg.mPt/km
d_inc = 10; // distance between collect and incineration in km
d_rec = 200;

```
d_rem = 1000;

prix_inc = 0.25; // €/kg
prix_rec = 2.5;
prix_rem = 1;

variables
carter1_epaisseur in [0.001,0.01]; // m
carter2_epaisseur in [0.001,0.01]; // m
carter1_ABS in [0,1]; // % ABS
carter1_PP in [0,1]; // % PP
carter1_Alu in [0,1]; // % PP
carter2_ABS in [0,1]; // % ABS
carter2_PP in [0,1]; // % PP
carter2_Alu in [0,1]; // % PP
//carter1_volume in [1e-6,1]; // m3
//carter2_volume in [1e-6,1]; // m3
//carter1_masse in [0.001,1]; // kg
//carter2_masse in [0.001,1]; // kg
carter1_inc in [0,1]; // % incineration
carter2_inc in [0,1]; // % incineration
carter1_rec in [0,1]; // % recyclage
carter2_rec in [0,1]; // % recyclage
carter1_rem in [0,1]; // % remanufacturing
carter2_rem in [0,1]; // % remanufacturing
carter1_dv in [4,15]; // durée de vie en années
carter2_dv in [4,15]; // durée de vie en années

pcb_surface in [0.001,0.004]; // surface pcb m2
//pcb_volume in [1e-9,1]; // volume pcb m3
//pcb_masse in [1e-3,0.01]; // masse pcb kg
pcb_dv in [4,15]; //durée de vie en années
pcb_inc in [0,1];
pcb_rec in [0,1];
pcb_rem in [0,1];

batterie_Pb in [0,1];
batterie_LiIon in [0,1];
batterie_NiMH in [0,1];
//batterie_energy in [0,1e3]; // Wh
//batterie_densite in [0,1e4]; // Wh/kg
//batterie_masse in [0.001,1]; // kg
//batterie_autonomy in [0,10]; //h
//batterie_nb_cycle;// in [0,6000];
//batterie_decharge in [0,100];
batterie_dv in [4,15];
//nb_usage in [0,1000];
batterie_inc in [0,1];
batterie_rec in [0,1];
batterie_rem in [0,1];

//ABS_EI99 in [0,1e3];
//PP_EI99 in [0,1e3];
//Alu_EI99 in [0,1e3];
//Pb_EI99 in [0,1e4];
//LiIon_EI99 in [0,1e4];
//NiMH_EI99 in [0,1e4];
EI99 in [100,1e5];
DV in [4,10];

//carter1_prix in [0,1e2];
//carter2_prix in [0,1e2];
//pcb__prix in [0,1e2];
//batterie_prix in [0,1e2];
module_prix in [0,1e2];
minimize EI99/DV;

constraints
// choix de matériau
carter1_ABS+carter1_PP+carter1_Alu<=1;
```

```
carter1_ABS+carter1_PP+carter1_Alu>=1;
carter2_ABS+carter2_PP+carter2_Alu=1;
carter2_ABS+carter2_PP+carter2_Alu>=1;
carter1_ABS*(1-carter1_ABS)<=0;
carter1_ABS*(1-carter1_ABS)>=0;
carter1_PP*(1-carter1_PP)<=0;
carter1_PP*(1-carter1_PP)>=0;
carter1_Alu*(1-carter1_Alu)<=0;
carter1_Alu*(1-carter1_Alu)>=0;
carter2_ABS*(1-carter2_ABS)<=0;
carter2_ABS*(1-carter2_ABS)>=0;
carter2_PP*(1-carter2_PP)<=0;
carter2_PP*(1-carter2_PP)>=0;
carter2_Alu*(1-carter2_Alu)<=0;
carter2_Alu*(1-carter2_Alu)>=0;
batterie_Pb+batterie_LiIon+batterie_NiMH <= 1;
batterie_Pb+batterie_LiIon+batterie_NiMH >= 1;
batterie_Pb*(1-batterie_Pb) <= 0;
batterie_Pb*(1-batterie_Pb) >= 0;
batterie_LiIon*(1-batterie_LiIon) <= 0;
batterie_LiIon*(1-batterie_LiIon) >= 0;
batterie_NiMH*(1-batterie_NiMH) <= 0;
batterie_NiMH*(1-batterie_NiMH) >= 0;

// calcul masse et volume
carter1_volume = carter1_surface*carter1_epaisseur;
carter2_volume = carter2_surface*carter2_epaisseur;
carter1_masse = carter1_ABS*carter1_volume*ABS_densite + carter1_PP*carter1_volume*PP_densite + carter1_Alu*carter1_volume*Alu_densite;
carter2_masse = carter2_ABS*carter2_volume*ABS_densite + carter2_PP*carter2_volume*PP_densite + carter2_Alu*carter2_volume*Alu_densite;
pcb_volume = pcb_surface*pcb_epaisseur;
pcb_masse = pcb_volume*pcb_densite;
batterie_energy = batterie_nb_cell*(batterie_Pb_U*batterie_Pb_capa*batterie_Pb + batterie_LiIon_U*batterie_LiIon_capa*batterie_LiIon +
batterie_NiMH_U*batterie_NiMH_capa*batterie_NiMH);
batterie_densite = batterie_Pb_densite*batterie_Pb + batterie_LiIon_densite*batterie_LiIon + batterie_NiMH_densite*batterie_NiMH;
batterie_masse = batterie_energy/batterie_densite;
batterie_autonomy = batterie_energy / puissance_moteur;

// choix fin de vie
carter1_inc+carter1_rec+carter1_rem<=1;
carter1_inc+carter1_rec+carter1_rem>=1;
carter2_inc+carter2_rec+carter2_rem<=1;
carter2_inc+carter2_rec+carter2_rem>=1;
carter1_inc*(1-carter1_inc)>=0;
carter1_inc*(1-carter1_inc)<=0;
carter1_rec*(1-carter1_rec)>=0;
carter1_rec*(1-carter1_rec)<=0;
carter1_rem*(1-carter1_rem)>=0;
carter1_rem*(1-carter1_rem)<=0;
carter2_inc*(1-carter2_inc)>=0;
carter2_inc*(1-carter2_inc)<=0;
carter2_rec*(1-carter2_rec)>=0;
carter2_rec*(1-carter2_rec)<=0;
carter2_rem*(1-carter2_rem)>=0;
carter2_rem*(1-carter2_rem)<=0;
pcb_inc+pcb_rec+pcb_rem<=1;
pcb_inc+pcb_rec+pcb_rem>=1;
pcb_inc*(1-pcb_inc)>=0;
pcb_inc*(1-pcb_inc)<=0;
pcb_rec*(1-pcb_rec)>=0;
pcb_rec*(1-pcb_rec)<=0;
pcb_rem*(1-pcb_rem)>=0;
pcb_rem*(1-pcb_rem)<=0;
batterie_inc+batterie_rec+batterie_rem <= 1;
batterie_inc+batterie_rec+batterie_rem >= 1;
batterie_inc*(1-batterie_inc) <= 0;
batterie_inc*(1-batterie_inc) >= 0;
batterie_rec*(1-batterie_rec) <= 0;
batterie_rec*(1-batterie_rec) >= 0;
batterie_rem*(1-batterie_rem) <= 0;
batterie_rem*(1-batterie_rem) >= 0;
```

```
// Calcul EI Recipe
ABS_EI99 = carter1_ABS*(carter1_masse*ABS_fab_EI99 + carter1_masse*carter1_inc*ABS_inc_EI99 + carter1_masse*carter1_rec*ABS_rec_EI99 - carter1_masse*carter1_rem*ABS_fab_EI99*rem_EI99) + carter2_ABS*(carter2_masse*ABS_fab_EI99 + carter2_masse*carter2_inc*ABS_inc_EI99 + carter2_masse*carter2_rec*ABS_rec_EI99 - carter2_masse*carter2_rem*ABS_fab_EI99*rem_EI99);
PP_EI99 = carter1_PP*(carter1_masse*PP_fab_EI99 + carter1_masse*carter1_inc*PP_inc_EI99 + carter1_masse*carter1_rec*PP_rec_EI99 - carter1_masse*carter1_rem*PP_fab_EI99*rem_EI99) + carter2_PP*(carter2_masse*PP_fab_EI99 + carter2_masse*carter2_inc*PP_inc_EI99 + carter2_masse*carter2_rec*PP_rec_EI99 - carter2_masse*carter2_rem*PP_fab_EI99*rem_EI99);
Alu_EI99 = carter1_Alu*(carter1_masse*Alu_fab_EI99 + carter1_masse*carter1_inc*Alu_inc_EI99 + carter1_masse*carter1_rec*Alu_rec_EI99 - carter1_masse*carter1_rem*Alu_fab_EI99*rem_EI99) + carter2_Alu*(carter2_masse*Alu_fab_EI99 + carter2_masse*carter2_inc*Alu_inc_EI99 + carter2_masse*carter2_rec*Alu_rec_EI99 - carter2_masse*carter2_rem*Alu_fab_EI99*rem_EI99);

Pb_EI99 = batterie_Pb*batterie_masse*(Pb_fab_EI99 + batterie_inc*Pb_inc_EI99 + batterie_rec*Pb_rec_EI99 - batterie_rem*Pb_fab_EI99*rem_EI99);
LiIon_EI99 = batterie_LiIon*batterie_masse*(LiIon_fab_EI99 + batterie_inc*LiIon_inc_EI99 + batterie_rec*LiIon_rec_EI99 - batterie_rem*LiIon_fab_EI99*rem_EI99);
NiMH_EI99 = batterie_NiMH*batterie_masse*(NiMH_fab_EI99 + batterie_inc*NiMH_inc_EI99 + batterie_rec*NiMH_rec_EI99 - batterie_rem*NiMH_fab_EI99*rem_EI99);

EI99 = ABS_EI99+PP_EI99+Alu_EI99 + transport_EI99*d_inc*carter1_masse*carter1_inc + transport_EI99*d_inc*carter2_masse*carter2_inc + transport_EI99*d_rec*carter1_masse*carter1_rec + transport_EI99*d_rec*carter2_masse*carter2_rec + transport_EI99*d_rem*carter1_masse*carter1_rem + transport_EI99*d_rem*carter2_masse*carter2_rem + pcb_masse*pcb_masse_EI99 + transport_EI99*d_inc*pcb_inc + transport_EI99*d_rec*pcb_rec + transport_EI99*d_rem*pcb_rem + batterie_nb_cell*(Pb_EI99 + LiIon_EI99 + NiMH_EI99) + transport_EI99*d_inc*batterie_inc + transport_EI99*d_rec*batterie_rec + transport_EI99*d_rem*batterie_rem;

// calcul durée de vie
carter1_dv = carter1_ABS*(300*carter1_epaisseur+1) + carter1_PP*(125*carter1_epaisseur+1) + carter1_Alu*(88.24*carter1_epaisseur+1);
carter2_dv = carter2_ABS*(300*carter2_epaisseur+1) + carter2_PP*(125*carter2_epaisseur+1) + carter2_Alu*(88.24*carter2_epaisseur+1);
pcb_dv = -0.11*pcb_masse+9;
//batterie_dv = 10*(-0.175*batterie_Pb*batterie_nb_cycle - 0.067*batterie_LiIon*batterie_nb_cycle - 0.02*batterie_NiMH*batterie_nb_cycle + 100)/100;
nb_usage = 2*52*batterie_dv;
total_energy = nb_usage*puissance_moteur*duree;
batterie_nb_cycle = (batterie_Pb*batterie_Pb_viellissement + batterie_LiIon*batterie_LiIon_viellissement + batterie_NiMH*batterie_NiMH_viellissement)*total_energy/batterie_energy;
batterie_decharge = 100 - batterie_nb_cycle*(0.175*batterie_Pb + 0.067*batterie_LiIon + 0.067*batterie_NiMH);
batterie_decharge >= 60;
DV = min(carter1_dv,carter2_dv,pcb_dv,batterie_dv);

batterie_rem*(batterie_dv) >= batterie_rem*1.8 * DV;
pcb_rem*(pcb_dv) >= pcb_rem*1.8 * DV;
carter1_rem*(carter1_dv) >= carter1_rem*1.8 * DV;
carter2_rem*(carter2_dv) >= carter2_rem*1.8 * DV;


carter1_prix = (carter1_ABS*ABS_prix + carter1_PP*PP_prix + carter1_Alu*Alu_prix)*carter1_masse;
carter2_prix = (carter2_ABS*ABS_prix + carter2_PP*PP_prix + carter2_Alu*Alu_prix)*carter2_masse;
pcb__prix = pcb_masse*pcb_prix;
batterie_prix = (batterie_Pb*Pb_prix + batterie_LiIon*LiIon_prix + batterie_NiMH*NiMH_prix)*batterie_energy*1e-3;
module_prix = carter1_prix+carter2_prix+pcb_prix+batterie_prix + (carter1_masse*carter1_inc+carter2_masse*carter2_inc+pcb_masse*pcb_inc+batterie_masse*batterie_inc)*prix_inc + (carter1_masse*carter1_rec+carter2_masse*carter2_rec+pcb_masse*pcb_rec+batterie_masse*batterie_rec)*prix_rec + (carter1_masse*carter1_rem+carter2_masse*carter2_rem+pcb_masse*pcb_rem+batterie_masse*batterie_rem)*prix_rem;
end
```

L'ADEME EN BREF

L'Agence de l'Environnement et de la Maîtrise de l'Energie (ADEME) participe à la mise en œuvre des politiques publiques dans les domaines de l'environnement, de l'énergie et du développement durable. Elle met ses capacités d'expertise et de conseil à disposition des entreprises, des collectivités locales, des pouvoirs publics et du grand public, afin de leur permettre de progresser dans leur démarche environnementale. L'Agence aide en outre au financement de projets, de la recherche à la mise en œuvre et ce, dans les domaines suivants : la gestion des déchets, la préservation des sols, l'efficacité énergétique et les énergies renouvelables, la qualité de l'air et la lutte contre le bruit.

L'ADEME est un établissement public sous la tutelle conjointe du ministère de l'Ecologie, du Développement durable et de l'Energie, et du ministère de l'Éducation nationale, de l'Enseignement supérieur et de la Recherche.

ADEME
Agence de l'Environnement et de la Maîtrise de l'Energie

Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE
MINISTÈRE DE L'ÉCOLOGIE, DU DÉVELOPPEMENT DURABLE ET DE L'ÉNERGIE
MINISTÈRE DE L'ÉDUCATION NATIONALE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE

ADEME
20, avenue du Grésillé
BP 90406 | 49004 Angers Cedex 01

www.ademe.fr

ABOUT ADEME

The French Environment and Energy Management Agency (ADEME) is active in the implementation of public policy in the areas of the environment, energy and sustainable development. The Agency provides expertise and advisory services to businesses, local authorities and communities, government bodies and the public at large, to enable them to establish and consolidate their environmental action. As part of this work ADEME helps finance projects, from research to implementation, in the areas of waste management, soil conservation, energy efficiency and renewable energy, air quality and noise abatement.

ADEME is a public agency under the joint authority of the Ministry for Ecology, Sustainable Development and Energy, and the Ministry for Education, Higher Education and Research.

ADEME
Agence de l'Environnement et de la Maîtrise de l'Energie

Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE
MINISTÈRE DE L'ÉCOLOGIE, DU DÉVELOPPEMENT DURABLE ET DE L'ÉNERGIE
MINISTÈRE DE L'ÉDUCATION NATIONALE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE

ADEME
20, avenue du Grésillé
BP 90406 | 49004 Angers Cedex 01

www.ademe.fr