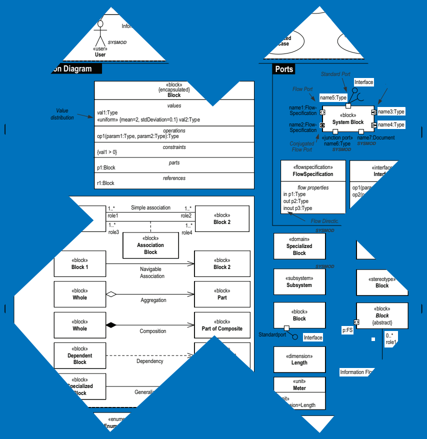
Department of Engineering Design and Production

Semantics and Knowledge Engineering for Requirements and Synthesis in Conceptual Design

Towards the Automation of Requirements Clarification and the Synthesis of Conceptual Design Solutions

---

François Christophe



A!

DOCTORAL
DISSERTATIONS

François Christophe Semantics and Knowledge Engineering for Requirements and Synthesis in Conceptual Design Aalto University

Aalto University publication series
DOCTORAL DISSERTATIONS 90/2012

Semantics and Knowledge Engineering for Requirements and Synthesis in Conceptual Design:
Towards the Automation of Requirements Clarification and the Synthesis of Conceptual Design Solutions

François Christophe

A doctoral dissertation completed for the degree of Doctor of Science (Technology) to be defended, with the permission of the Aalto University School of Engineering, at a public examination held at the lecture hall 216 (Otakaari 4) of the school on 27 July 2012 at 12.

Aalto University
School of Engineering
Department of Engineering Design and Production
Research group on Early development process and Systems Engineering

Supervising professor
Pr. Éric Coatanéa (Aalto) and Pr. Alain Bernard (ECN)

Preliminary examiners
Pr. Bernard Yannou, Ecole Centrale Paris, France
Pr. Georges Fadel, Clemson University, United States of America

Opponents
Pr. Chris McMahon, University of Bath, United Kingdom
Pr. Georges Fadel, Clemson University, United States of America

Aalto University publication series
DOCTORAL DISSERTATIONS 90/2012

ISBN 978-952-60-4696-9 (printed)
ISBN 978-952-60-4697-6 (pdf)
ISSN-L 1799-4934
ISSN 1799-4934 (printed)
ISSN 1799-4942 (pdf)
http://urn.fi/URN:ISBN:978-952-60-4697-6

Unigrafia Oy
Helsinki 2012

Finland

NORDIC ECOLABEL
441 697
Printed matter

Aalto University

Abstract

Aalto University, P.O. Box 11000, FI-00076 Aalto www.aalto.fi

Author
François Christophe

Name of the doctoral dissertation
Semantics and Knowledge Engineering for Requirements and Synthesis in Conceptual Design: Towards the Automation of Requirements Clarification and the Synthesis of Conceptual Design Solutions

Publisher School of Engineering

Unit Department of Engineering Design and Production

Series Aalto University publication series DOCTORAL DISSERTATIONS 90/2012

Field of research Engineering Design

Manuscript submitted 9 March 2012 **Date of the defence** 27 July 2012

Permission to publish granted (date) 6 June 2012 **Language** English

☐ **Monograph** ☒ **Article dissertation (summary + original articles)**

Abstract

This thesis suggests the use of tools from the disciplines of Computational Linguistics and Knowledge Representation with the idea that such tools would enable the partial automation of two processes of Conceptual Design: the analysis of Requirements and the synthesis of concepts of solution. The viewpoint on Conceptual Design developed in this research is based on the systematic methodologies developed in the literature. The evolution of these methodologies provided precise description of the tasks to be achieved by the designing team in order to achieve successful design. Therefore, the argument of this thesis is that it is possible to create computer models of some of these tasks in order to partially automate the refinement of the design problem and the exploration of the design space.

In Requirements Engineering, the definition of requirements consists in identifying the needs of various stakeholders and formalizing it into design specifications. During this task, designers face the problem of having to deal with individuals from different expertise, expressing their needs with different levels of clarity. This research tackles this issue with requirements expressed in natural language (in this case in English). The analysis of needs is realised from different linguistic levels: lexical, syntactic and semantic. The lexical level deals with the meaning of words of a language. Syntactic analysis provides the construction of the sentence in language, i.e. the grammar of a language. The semantic level aims at finding about the specific meaning of words in the context of a sentence. This research makes extensive use of a semantic atlas based on the concept of clique from graph theory. Such concept enables the computation of distances between a word and its synonyms. Additionally, a methodology and a metric of similarity was defined for clarifying requirements at syntactic, lexical and semantic levels. This methodology integrates tools from research collaborators.

In the synthesis process, a Knowledge Representation of the necessary concepts for enabling computers to create concepts of solution was developed. Such, concepts are: function, input/output flow, generic organs, behaviour, components. The semantic atlas is also used at that stage to enable a mapping between functions and their solutions. It works as the interface between the concepts of this Knowledge Representation.

Keywords Conceptual design, Requirements analysis, Synthesis process, Semantic, Knowledge Representation

ISBN (printed) 978-952-60-4696-9 **ISBN (pdf)** 978-952-60-4697-6

ISSN-L 1799-4934 **ISSN (printed)** 1799-4934 **ISSN (pdf)** 1799-4942

Location of publisher Espoo **Location of printing** Helsinki **Year** 2012

Pages 205 **urn** http://urn.fi/URN:ISBN:978-952-60-4697-6

Aalto-yliopisto

Aalto-yliopisto, PL 11000, 00076 Aalto www.aalto.fi

Tiivistelmä

Tekijä
François Christophe

Väitöskirjan nimi
Semantics and Knowledge Engineering for Requirements and Synthesis in Conceptual Design: Towards the Automation of Requirements Clarification and the Synthesis of Conceptual Design Solutions

Julkaisija Insinööritieteiden korkeakoulu

Yksikkö Koneenrakennustekniikan laitos

Sarja Aalto University publication series DOCTORAL DISSERTATIONS 90/2012

Tutkimusala Tuotekehityksen menetelmät

Käsikirjoituksen pvm 09.03.2012 **Väitöspäivä** 27.07.2012

Julkaisuluvan myöntämispäivä 06.06.2012 **Kieli** Englanti

☐ **Monografia** ☒ **Yhdistelmäväitöskirja (yhteenveto-osa + erillisartikkelit)**

Tiivistelmä

Tämä väitöskirja esittää laskennallisen kielentutkimuksen sekä tiedon esittämisen tieteenaloilla käytettyjen työkalujen hyödyntämistä kahdessa eri tuotekehityksen konseptisuunnittelun prosessissa; tuotteiden vaatimusten analyysissä sekä konseptuaalisten suunnitteluratkaisujen synteesissä. Konseptisuunnittelun näkökulma, jota käytetään tässä väitöskirjassa, perustuu kirjallisuudesta tuttuihin systemaattisiin tuotekehitysmenetelmiin. Näiden menetelmien kehitys tarjoaa tarkan kuvauksen prosessin tehtävistä, jotka suunnittelutyöryhmän tulee suorittaa onnistuakseen tuotesuunnittelussa. Tästä syystä tämä väitöskirja esittää, että on mahdollista osittain automatisoida tuotekehitysprosessia luomalla laskennallisia malleja tietyistä tuotekehitysprosessin tehtävistä, kuten suunnitteluongelman tarkentamisesta sekä suunnitteluavaruuden tutkimuksesta.

Tuotekehityksessä tuotteiden vaatimukset määritellään identifioimalla ensin eri sidostahojen tarpeet tuotteen osalta, ja sen jälkeen formalisoimalla nämä vaatimusmäärittelyksi. Tämän vaiheen aikana suunnittelijat kohtaavat usein ongelmia erilaisten käyttäjien eritasoisten tarvemääritelmien kanssa. Tutkimus tässä väitöskirjassa keskittyy luonnollisella kielellä (ja erityisesti englannin kielellä) ilmaistuihin tuotevaatimuksiin. Tarpeiden analyysi on toteutettu eri kielentutkimuksen tasoilla; sanastollisella, lauseopillisella sekä semanttisella tasolla. Sanastollisella tasolla käsitellään sanojen merkitystä kielessä, lauseopillisella tasolla käsitellään lauseen rakennetta, eli kielen kielioppia, ja semanttisella tasolla päämääränä on löytää sanojen nimenomainen merkitys kunkin lauseen kontekstissa. Tällaiset käsitteet mahdollistavat etäisyyden laskemisen sanan itsensä sekä sen eri synonyymien välillä. Lisäksi sanojen samankaltaisuuden määrittämiseen kehitettiin metodologia sekä mitta, jotka edelleen selventävät tuotevaatimuksia sekä sanastollisella, lauseopillisella että semanttisella tasolla. Tämä metodologia hyödyntää tutkimusyhteistyökumppaneiden kehittämiä työkaluja. Tuotekehitysprosessin synteesivaiheessa kehitettiin tiedollinen esitys tarvittavista käsitteistä, jotka mahdollistavat suunnittelukonseptien tuottamisen tietokoneilla. Näihin käsitteisiin lukeutuvat mm. tuotteen toiminnallisuudet, input/output-virta, geneeriset toimielimet, tuotteen käyttäytyminen sekä tuotteen rakenneosat. Myös semanttista atlastahyödynnetään tässä vaiheessa

Avainsanat

ISBN (painettu) 978-952-60-4696-9 **ISBN (pdf)** 978-952-60-4697-6

ISSN-L 1799-4934 **ISSN (painettu)** 1799-4934 **ISSN (pdf)** 1799-4942

Julkaisupaikka Espoo **Painopaikka** Helsinki **Vuosi** 2012

Sivumäärä 205 **urn** http://urn.fi/URN:ISBN:978-952-60-4697-6

Résumé

Fondements

Le processus de développement de produit doit relever de multiples exigences de qualité, de coûts et de temps d'accès au marché. À cet effet, et aussi parce qu'il implique plusieurs acteurs fournissant leurs visions sur les services que le futur produit devrait accomplir, la conception technique est une activité complexe dont le processus doit être formalisé (Deneux, 2002).

La phase conceptuelle de conception commence, dans la plupart des cas, avec l'expression d'un problème rencontré. Au tout début de la conception, ce problème est souvent exprimé en langage naturel ou sous la forme de croquis qui expliquent l'origine de ce problème de conception. Ce problème est généralement mal définis car il est fourni par plusieurs acteurs externes à la conception, par example, les clients et les futurs utilisateurs du produit à concevoir. Le premier objectif dans la phase conceptuelle du développement de produits est d'affiner ce problème mal définis et de le formuler de manière plus formelle afin d'éviter au maximum les ambiguïtés dans la compréhension de ce problème entre les différents acteurs de la conception. Cette phase correspond à l'analyse des besoins et est aussi appelée l'ingénierie des exigences. Il y a principalement deux possibilités dans la création d'un problème de conception. Soit le problème est porté à l'équipe de conception comme une demande de clients, soit l'équipe de conception cherche à créer un besoin sur le marché en vue d'acquérir un avantage concurrentiel par rapport à ses concurrents.

Dans le premier cas, où le problème provient de clients, l'ingénierie des exigences est utilisée pour s'assurer que l'équipe de conception a clairement compris les besoins attendus par les clients. Cette compréhension est essentielle afin de fournir des solutions de conception qui sont conformes à ces besoins. De plus, pour des raisons commerciales, il est important pour l'équipe de conception que ces solutions ne fournissent pas plus que ce qui est attendu par les clients. Par conséquent, le problème initialement faible doit être affiné en évitant, autant que possible, toute ambiguïté dans la signification des termes utilisés pour définir les exigences.

Dans le second cas, où l'équipe de conception aborde un problème de conception lié à la création d'un besoin sur le marché, il est important de *réaliser un examen des plus exhaustifs de la situation actuelle des solutions existantes répondant partiellement à ce problème de conception. Ce cas présente la complexité de la charge cognitive associée à l'activité de conception ainsi que la co-évolution entre un problème de conception et ses solutions (Dorst and Cross, 2001) (Zeng, 2004). De nombreux chercheurs en conception de produit affirment, après Schön, que l'ambiguité permet la créativité et que la créativité est cruciale lors de l'activité de conception (Schön, 1983). Cet argument est valable pour les parties de la conception impliquant la synthèse de solutions et donc, la créativité de l'équipe de conception. Néanmoins, l'ingénierie des exigences est une partie de la conception qui implique l'analyse du problème de conception. En outre, la résultante de ce raffinement devrait produire des exigences dénuées d'ambiguïtés de manière à établir un objectif clairement défini pour une activité de conception spécifique et hautement dépendante de son contexte. L'ingénierie des exigences est dierctement suivie par, principalement, deux processus formant la phase conceptuelle de dévelopement de produits: la synthèse de concepts et, l'évaluation, comparaison de ces concepts. Ces deux processus sont considérés comme séquentiels ou simultanés selon les différentes méthodologies proposées par la communauté scientifique. Néanmoins, ils sont des processus cruciaux de conception et sont toujours définis avec précision par ces diverses méthodologies (Tomiyama et al., 2009).*

Les méthodologies et guides décrivant la phase conceptuelle du processus de conception sont nombreuses au sein de la communauté scientifique (Tomiyama et al., 2009). En fait, dans le but de comprendre le processus de conception, les scientifiques du domaine ont décrit ce processus à travers des modèles et représentations de la connaissance mise en oeuvre lors de cette phase de conception de produits techniques (Gero, 1990) (Tomiyama et al., 2003) (Kitamura et al., 2007). Au cours de leur évolution, ces modèles ont obtenu une maturité en précision permettant de proposer leur application systématique pour une conception réussie. Les méthodolologies décrivant la phase conceptuelle de conception attachent aussi une importance particulière à la description du procédé de synthèse (Chakrabarti, 2002) (Antonsson and Cagan, 2001). De nos jours, l'évolution de l'ingénierie basée sur documents vers l'ingénierie dirigée par les modèles permet l'implémentation concrète de ces méthodologies sous la forme de méta-modèles numériques. La description formelle de ces méthodes et modèles offre la possibilité de les mettre en œuvre concrètement sous la forme d'applications *informatiques. Ceci peut permettre l'automatisation de certaines parties du processus de préconception.*

L'aspect systématique proposé par les manuels et méthodologies de conception ainsi que leur mise en œuvre par des méta-modèles numériques suggèrent la possibilité d'intégrer l'ordinateur à un autre niveau au sein du processus de conception. Cela pourrait permettre d'aider les ingénieurs de conception à considérer l'ordinateur non plus comme un assistant mais comme un membre à part entière d'une équipe de conception.

Problème scientifique

Les fondements de cette recherche ont soulevé plusieurs questions concernant la possible automatisation partielle du processus de préconception.

Premièrement, quels sont les concepts généraux associés aux différentes méthodologies proposées en préconception? De quelle manière ces concepts ainsi que leurs relations peuvent-ils être représentés afin de permettre une meilleure compréhension de l'activité de préconception autant par des applications informatiques que par l'équipe de conception? De part les nombreuses méthodologies proposées par la communauté scientifique, ces questions révèlent un besoin d'unité entre les principaux concepts utilisés et de cohérence entre les différentes définitions de ces concepts. La représentation sous la forme de modèles numériques de ces concepts ainsi que de leurs relations doit être étudiée.

Secondement, comment assister les ingénieurs de conception dans la formulation du problème de conception, problème initialement vague ou mal défini, de manière plus formelle et dénuée d'ambiguités? L'analyse des exigences initiales par ordinateur est rendue difficile de part leur nature. En effet, elles sont principalement exprimées en langage naturel. L'automatisation de ce procédé de formulation des exigences doit être étudié du point de vue de la sémantique du langage, c'est à dire de la possibilité de l'expression d'exigences par des mots pouvant posséder des significations diverses.

Troisièmement, comment permettre à des programmes informatiques d'acquérir une connaissance suffisante à la compréhension des concepts utilisés lors de la phase de synthèse de solutions conceptuelles? Quelles sont *les connaissances nécessaires aux applications informatiques pour synthétiser des concepts de solution pertinents? Pour que ces applications puissent proposer des concepts de solution pertinents et utiles aux concepteurs, une voie de réponse serait qu'elles aient accès à la connaissance des concepteurs ainsi qu'aux concepts, modèles et produits précédemment implémentés. Cette voie de recherche fait donc logiquement appel aux domaines de recherche de l'ingénierie des connaissances, de la représentation des connaissances ainsi qu'à la capitalisation de connaissances industrielles. Le lien fait par les concepteurs entre le problème de conception auxquels ils sont confrontés et, les produits précédemment concus ainsi que l'invention de nouveaux principes de fonctionnement du produit étant concu est le reflet direct de la complexité de l'activité de conception. De plus, ceci est le reflet de la complexité de l'esprit et de sa capacité à innover en associant des briques de connaissances qui n'avaient auparavant pas de lien évident entre elles. L'idée d'essayer de donner cette capacité de liaison aux applications informatiques est donc une question difficile qui est le coeur de ces travaux de recherche.*

Objectifs de recherche

L'objectif de cette thèse est de considérer l'utilisation des ordinateurs comme support de l'équipe de conception dès le début de cette activité. Par conséquent, cette recherche vise à appliquer des méthodes et des outils informatiques dès les phases d'élicitation et de représentation en ingénierie des exigences et ce jusqu'à la synthèse de concepts de solution en phase conceptuelle de conception de produits. Ces outils devraient aider les concepteurs à obtenir une représentation plus formelle des exigences.

En outre, cette recherche étudie la possibilité d'exprimer les concepts clés de connaissance utilisée par les concepteurs au cours du processus de synthèse du design conceptuel. L'expression de cette connaissance dans un format analysable par ordinateur devrait conduire à une génération d'applications informatiques proposant des concepts de solution non triviaux assistant l'exploration de l'espace de conception.

De manière plus générale, cette recherche vise à appliquer des concepts récents provenant de l'Intelligence Artificielle, de la Représentation des Connaissances, de la Sémantique et de la Linguistique Informatique pour le *développement d'outils informatiques supportant les phases d'élicitation et de représentation des exigences et la phase de synthèse du processus de préconception.*

Méthodes de recherche

Les méthodes de recherche appliquées dans cette thèse combinent les travaux de recherche qui portent sur les méthodologies liées à la phase conceptuelle du dévelopement de produits avec les disciplines de la Linguistique Informatique et de l'Intelligence Artificielle.

Dans un premier temps, l'approche consiste à analyser les différences et similitudes existant entre les multiples définitions du processus systématique de préconception de la littérature (Motte, 2008) (Tomiyama et al., 2009). Cette analyse permet d'identifier les concepts et termes utilisés en phase conceptuelle de conception et d'en proposer une unification. De plus, les relations entre ces concepts sont observées à travers un schéma de représentation des connaissances en préconception (Gero, 1990) (Umeda and Tomiyama, 1995) (Gero and Kannengiesser, 2004). Cette analyse permet de définir une vision synthétique fournissant des définitions consensuelles des concepts utilisés dans les phases de conception considérées dans cette thèse.

Le dévelopement de produits commence par la définition d'un problème de conception. Ce problème de conception exprime l'émergence de besoins sur le marché pour un nouvel artefact devant être conçu. Comme ce problème est à l'interface entre le client, l'utilisateur potentiel et l'équipe de conception impliquée dans la création d'un artéfact répondant à ce problème, il est par nature un problème mal ou faiblement défini (Simon, 1996). Cette propriété du problème de conception initial est due au manque d'informations sur les besoins réels du client ou utilisateur. Le problème initial doit être affiné afin d'en formuler une représentation plus formelle des exigences liées au futur produit. dans cette thèse, la représentation plus formelle des exigences doit être percue de la manière suivante:

- *Les exigences doivent contenir des informations sur l'ensemble des services que l'artefact conçu doit fournir. L'ensemble des services sont également appelés fonctions de service dans la littérature (Dardy et al., 2003)*

(Miles, 1961).

- *En plus d'exprimer les besoins en termes de fonctions de service, les exigences doivent contenir des informations sur l'environnement et le contexte dans lequel le produit conçu se trouvera selon les différentes phases de son cycle de vie.*

- *Les exigences doivent également décrire les relations entre les différentes parties prenantes qui sont en interaction avec l'artefact.*

- *Enfin, les exigences doivent décrire la performance attendue de l'artefact pour fournir les fonctions de service, d'une manière qualitative et, si possible, d'une manière quantitative.*

Le processus de synthèse vient à la suite de l'ingénierie des exigences dans la phase conceptuelle du dévelopement de produit. Comme l'expriment les méthodologies de conception (Ulrich and Eppinger, 2011) (Pahl and Beitz, 2007) (Otto and Wood, 2001), le processus de synthèse commence par une décomposition fonctionnelle des fonctions de service en fonctions techniques nécessaires à la mise en œuvre des fonctions de service. Cette approche est appelée approche descendante ("top-down"). Il en résulte la création d'un arbre fonctionnel. L'idée de l'approche descendante est de décomposer un problème complexe exprimé par les fonctions de service en plusieurs sous-problèmes plus simples. Les méthodologies suggèrent également que les fonctions élémentaires puissent être immédiatement implémentées grâce à l'utilisation de composants de base, solutions structurelles remplissant cette fonction technique (Coatanéa, 2005). L'étape suivante du processus de synthèse consiste à intégrer ensemble ces composants de base afin d'obtenir un ensemble structurel formant un concept de solution.

Après cet aperçu des méthodes utilisées, cette thèse étudie le potentiel des outils existants dans les disciplines de l'informatique qui pourraient être utilisés pour automatiser en partie le processus de préconception et assister les concepteurs au cours de cette activité. Comme l'élicitation des exigences implique la reformulation du problème exprimé en langage naturel, cette thèse étudie les pratiques utilisées en linguistique informatique et en traitement informatique du langage naturel. Dans ce domaine, les chercheurs ont créé des métriques définissant un espace sémantique qui *permet l'évaluation de distances entre les synonymes d'un mot (Ploux and Victorri, 1998).*

Les fondements de cette recherche et les méthodologies présentées ont révélé l'importance de la charge cognitive associée à l'activité de conception. C'est pourquoi cette thèse dresse tout naturellement un examen des pratiques utilisées pour formaliser les connaissances dans les disciplines de l'ingénierie des connaissances et de la représentation des connaissances du domaine de l'Intelligence Artificielle.

Positionnement de la recherche

Cette thèse se positionne au niveau de la formalisation de certaines parties du processus de préconception. Son but est de fournir des connaissances sur ces sous-processus afin qu'elles soient lisibles par des applications informatiques et que celles-ci puissent, par la suite, inférer à partir de ces connaissances. Cette formalisation commence par l'élicitation et la formulation des exigences initialement exprimées en langage naturel. Dans une deuxième phase, cette recherche porte sur la possibilité pour les machines de synthétiser des concepts de solution.

Cette recherche se limite à la synthèse de base des éléments génériques qui seront utilisés et combinés dans la conception du produit. Elle ne traite pas de la multiplicité de ces éléments ou des interactions entre eux. Néanmoins, des sources de solutions possibles à ces problèmes sont apparues au cours de ce travail de recherche et sont proposées comme de possibles orientations futures de la recherche dans la conclusion de cette thèse.

Contribution scientifique

L'étude des méthodologies et modèles de conception a conduit cette recherche vers des contributions positionnées à différents niveaux de description du processus de préconception.

Concernant l'élicitation des exigences, cette recherche a permis la réalisation d'un assistant informatique pour le processus de désambiguation des exigences. De fait, à ce stade de la conception, les sources de malentendus sont nombreuses. Elles proviennent de diverses origines lors de la *création du cahier des charges: elles peuvent intervenir entre les clients et l'équipe de conception, entre les consommateurs ou utilisateurs et l'équipe de conception ou encore à l'intérieur de l'équipe de conception elle-même. Ces sources de malentendus sont toutes des obstacles à l'expression des besoins de manière claire et précise. Cet outil fournit à l'équipe de conception le sens de chaque mot utilisé dans l'expression initiale du problème de conception selon le contexte d'utilisation de ce mot dans la phrase. Dans le cas où un mot reste avec de multiples significations possibles, l'assistant incite les concepteurs à préciser le sens de ce qu'ils souhaitent exprimer en posant des questions aux parties prenantes au sujet de ce terme spécifique. En ce sens, cette contribution est intéressante car elle concentre l'attention des concepteurs sur des concepts manquant de clareté qui pourraient être au coeur du problème de conception.*

Au niveau de la synthèse de concepts, cette thèse a mené au développement d'une ontologie unifiant des taxonomies de fonctions standards, d'organes génériques, de variable de flux et d'énergie et de composants physiques. La Figure 1 présente schématiquement cette ontologie et les taxonomies qu'elle intègre. Cette ontologie est conforme avec l'ontologie de l'ingénierie de systèmes developpée par Tudorache (Tudorache, 2006). L'utilisation d'un atlas sémantique pour créer des liens entre fonctions techniques et organes génériques permet à cette liaison de devenir une liaison dynamique. En effet, dans les cas précédents de synthèse automatisée observés dans la littérature (Kitamura et al., 2007) (Chakrabarti et al., 2011), ces liens sont habituellement définis en tant que relations fixée "en dur" dans la base de connaissances de l'application informatique. Dans notre cas, ce lien est créé par un atlas sémantique disponible en ligne. Ce mode dynamique de liaison apporte de la flexibilité à la correspondance entre fonctions et composants génériques. Cette flexibilité dépend du corpus de textes utilisé par l'atlas sémantique et, par conséquent, il est possible d'obtenir des résultats différents lorsqu'il est choisi d'utiliser une même fonction dans des domaines techniques différents.

À un niveau plus général, les contributions précédentes et les constatations au sujet des connexions sémantiques entre fonctions et composants ont amené naturellement à la création d'un nouveau modèle de représentation des connaissances nécessaires en préconception et des modifications au niveau des relations entre ces concepts de connaissance. Ce modèle

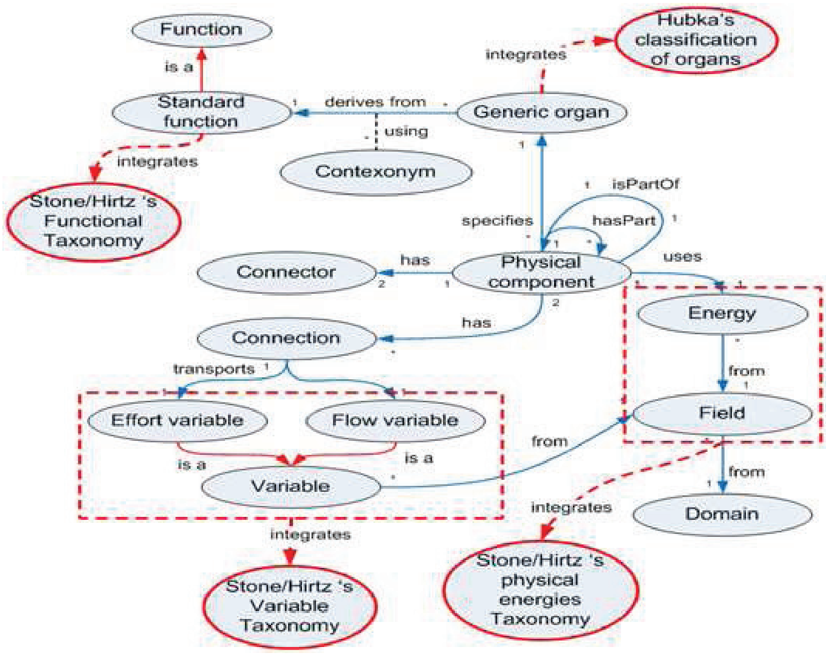


Figure 1. Description ontologique de la synthèse de solutions

de connaissances est intitulé "RFBS model" (RFBS pour "Requirement-Function-Behavior-Structure") car il apporte des modifications au modèle FBS de Gero (Gero, 1990). Ce modèle inclut le concept d'Exigence (R) qui est un concept majeur de la préconception car il en est à l'origine. Le modèle RFBS ajoute également le concept de généricité structurelle, en y incluant le concept de Structure Générique (GS). La structure générique est utilisée pour lier la fonction et la structure physique. C'est elle qui établit le lien sémantique entre l'expression fonctionnelle (verbe d'action et flux d'entrée/sortie) et le composant physique remplissant cette action (équation de liaison connue et variables requises en entrée/sortie ainsi que paramètres internes au composant). La Figure 2 présente ce modèle de connaissances ainsi que les relations entre ces concepts au cours de l'avancement du processus de préconception. La Table 1 détaille les relations entre concepts de connaissance en alignement avec les phases de préconception.

De fait, la structure générique est similaire à la notion de classe abstraite dans le paradigme de conception orientée objet. En outre, la présente recherche associe clairement chaque concept du modèle RFBS avec les types de diagrammes du langage de modélisation de systèmes SysML. Cette

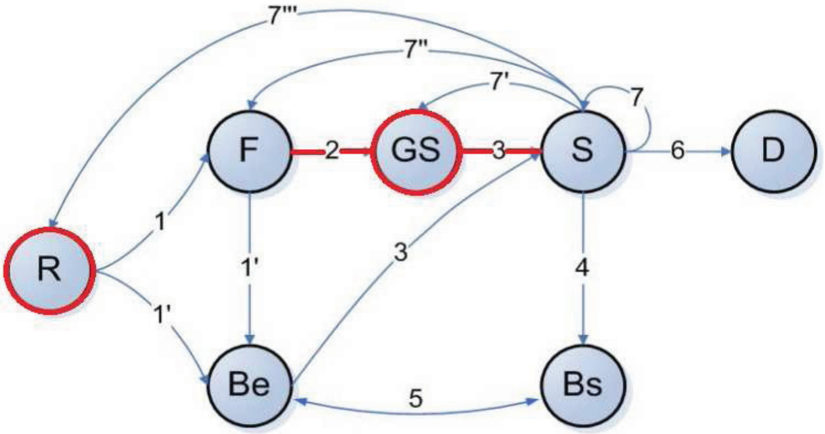
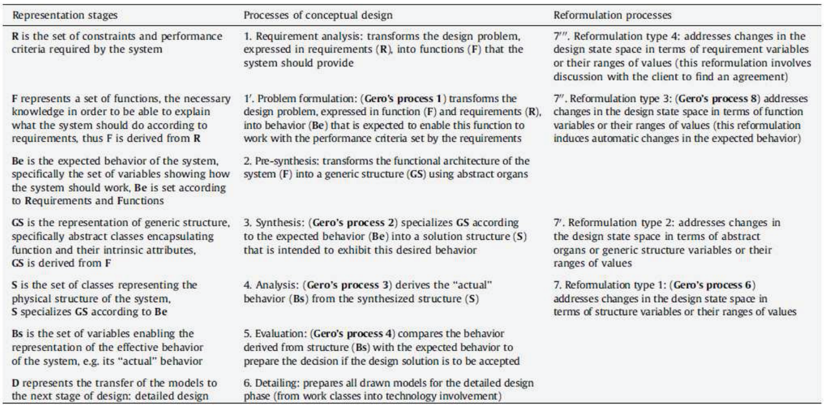


Figure 2. Le modèle RFBS



Representation stages	Processes of conceptual design	Reformulation processes
R is the set of constraints and performance criteria required by the system	1. Requirement analysis: transforms the design problem, expressed in requirements (**R**), into functions (**F**) that the system should provide	7‴. Reformulation type 4: addresses changes in the design state space in terms of requirement variables or their ranges of values (this reformulation involves discussion with the client to find an agreement)
F represents a set of functions, the necessary knowledge in order to be able to explain what the system should do according to requirements, thus **F** is derived from **R**	1′. Problem formulation: (**Gero's process 1**) transforms the design problem, expressed in function (**F**) and requirements (**R**), into behavior (**Be**) that is expected to enable this function to work with the performance criteria set by the requirements	7″. Reformulation type 3: (**Gero's process 8**) addresses changes in the design state space in terms of function variables or their ranges of values (this reformulation induces automatic changes in the expected behavior)
Be is the expected behavior of the system, specifically the set of variables showing how the system should work, **Be** is set according to **Requirements** and **Functions**	2. Pre-synthesis: transforms the functional architecture of the system (**F**) into a generic structure (**GS**) using abstract organs	
GS is the representation of generic structure, specifically abstract classes encapsulating function and their intrinsic attributes, **GS** is derived from **F**	3. Synthesis: (**Gero's process 2**) specializes **GS** according to the expected behavior (**Be**) into a solution structure (**S**) that is intended to exhibit this desired behavior	7′. Reformulation type 2: addresses changes in the design state space in terms of abstract organs or generic structure variables or their ranges of values
S is the set of classes representing the physical structure of the system, **S** specializes **GS** according to **Be**	4. Analysis: (**Gero's process 3**) derives the "actual" behavior (**Bs**) from the synthesized structure (**S**)	7. Reformulation type 1: (**Gero's process 6**) addresses changes in the design state space in terms of structure variables or their ranges of values
Bs is the set of variables enabling the representation of the effective behavior of the system, e.g. its "actual" behavior	5. Evaluation: (**Gero's process 4**) compares the behavior derived from structure (**Bs**) with the expected behavior to prepare the decision if the design solution is to be accepted	
D represents the transfer of the models to the next stage of design: detailed design	6. Detailing: prepares all drawn models for the detailed design phase (from work classes into technology involvement)	

Table 1. Les processus de modification des connaissances durant les phases de préconception

association a pour objectif de montrer que le langage SysML couvre entièrement les concepts requis lors du processus de préconception. Table 2 présente la correspondance entre les concepts de connaissance nécessaires en conception préliminaire et les éléments graphiques du langage SysML. L'utilisation du langage SysML en préconception ouvre une voie vers une pensée objet de ce processus qui, jusqu'alors n'est que majoritairement dominé par une vision fonctionnelle du système. Le paradigme objet apporte de nombreuses avancées en préconception car il permet d'abstraire le problème de conception au niveau de sa description par ses modèles. En effet, en ingénierie logicielle, il a permis d'abstraire l'expression du problème et de ses solutions au niveau des modèles graphiques et, de donner plus d'importance au modèle, en terme de structuration de code, qu'au code en soi. De ce point de vue, utiliser un langage de formalisation objet en con-

ception de systèmes, de manière générale, pourrait ouvrir ce même niveau d'abstraction et permettre la réalisation de manière concrète d'une conception dirigée par les modèles.

RFBS states	*Corresponding SysML diagrams*	*Description of the diagram type*
R	*Requirement diagram (req)*	*Requirements made with or coming from stakeholders are first defined by req. Stereotypes of requirements can be defined in order to classify requirements into, for example, functional and non-functional.*
	Block Definition Diagram (bdd)	*The description of the boundaries of the system "as-is", its environment and the relations between the system and actors of its environment can be represented with bdd*
F	*Use Case diagram (uc)*	*Services provided by the system to actors are described in a uc. uc is at the interface between R and F as a use case should $<< refine >>$ a functional requirement and $<< trace >>$ non-functional requirements for traceability reasons. It is also possible to describe the functional decomposition of services with technical functions in uc with relation $<< include >>$*
GS	*Block Definition Diagram (bdd)*	*Each technical functions at lowest level of the functional tree decomposition is encapsulated into an abstract organ.*
B_e	*Activity diagram (act)*	*act represents the behaviour of the System and the order in which technical functions should appear to realise the service.*
	State Machine diagram (stm)	*stm represents the different states in which the System can be during its operation*

S	*Block Definition diagram (bdd)* *Internal Block diagram (ibd)*	*System physical structure is defined by bdd. Each block of this bdd is derived from GS and specialises GS with the states variables discovered while modeling B_e. Such block contains internal variables (attributes of the block) and defines the necessary input variables for the function to be achieved as well as the provided output variables (prototype of operation of the block).* *Whereas bdd defines the hierarchical structure of the system, ibd represents the interactions between its components and the way they exchange matter, energy or information flows.*
B_S	*Activity diagram (act)* *State Machine diagram (stm)* *Sequence diagram (seq)* *Parametric diagram (par)*	*Similar than for B_e but for the analysis of the behavior of the concepts of solution.* *Analysis of the sequence of flow exchange between components of the system* *When all variables are known par represents their use in the physical equations governing the system in a graph. The analysis of the variables involved in most of these equations as well as the causal analysis of interaction between variables helps finding the variables of most importance for the efficiency of the system (performance variables).*
D	*every diagram*	*The entire project modeled with SysML should be used for the detailed design phases of each discipline involved in the design (e.g. mechanical, electrical and software engineering).*

Table 2. Correspondance entre les concepts RFBS et les diagrammes SysML

Conclusions et perspectives

Les travaux de recherche présentés dans ce manuscrit montrent:

- *l'utilité d'une analyse sémantique au sens de la linguistique computationnelle durant les phases:*
  - *d'élicitation et de représentation des exigences, de manière à assurer une compréhension claire du problème de conception*
  - *de synthèse de concepts de solution, de manière à lier, par leurs sens partagés, une fonction technique à des composants génériques réalisant cette fonction*
- *la possiblité de réaliser cette analyse à l'aide d'assistants logiciels grâce aux outils provenant de la linguistique computationnelle: outils d'analyse syntaxique, lexicale et sémantique,*
- *l'importance d'un formalisme de représentation des connaissances et surtout,*
- *l'importance de la représentation numérique des concepts de connaissance utilisés lors de la synthèse de manière à permettre aux assistants informatiques d'utiliser ces concepts clés de connaissance.*

Cette thèse couvre un spectre important des phases de préconception de produits et systèmes puisque, seule la partie d'évaluation des concepts de solution n'y est pas examinée en détail. En effet, l'objet de cette thèse est avant tout d'observer les phases de la conception préliminaire des points de vue de la sémantique et de la formalisation des connaissances nécessaires au dévelopement de concepts de solution. La phase d'évaluation, elle, relève plutôt d'un problème mathématique complexe d'optimisation multi-critères et d'aide à la décision. En effet, afin de pouvoir réaliser cette phase, le sens des données problème / solutions doit être analysé et formulé de manière à faciliter cette évaluation. C'est pour ces raisons que cette thèse couvre principalement les phases d'élicitation et représentation des exigences ainsi que la phase de synthèse de concepts de solution. De plus, ces trois phases contiennent de nombreux éléments textuels qui ne peuvent *être réduits à un formalisme mathématique du fait de leurs sens multiples. En ce sens, ces travaux de recherche préparent et tendent à faciliter la phase d'évaluation de concepts de deux manières:*

- *premièrement en mettant l'accent sur les points critiques contenus au niveau des exigences; points sur lesquels des ambiguités liées aux multiples sens possibles persistent malgré l'ajout de précisions. Ceci permet alors de donner une importance particulière aux variables dérivées de cette partie du probléme lors de la confrontation entre concepts de solution et leur validité par rapport au problème de conception.*

- *secondement, en assistant l'équipe de conception lors de l'exploration de l'espace de solution par une synthèse systématique et logicielle de concepts de solution multiples.*

Les perspectives envisagées à la suite de ce travail ouvrent deux voies de recherche:

- *La première concernant l'approfondissement des pistes proposées dans cette thèse pour la synthèse automatique de concepts de solution.*

- *La seconde concernant l'analyse des exigences grâce aux outils de linguistique computationnelle.*

Dans le cas de la première voie de recherche, la synthèse d'embryons de solution par composants génériques pourrait être approfondie par l'étude des relations possibles entre ces composants. Les composants génériques obtenus lors de la phase de synthèse proposée dans cette thèse pourraient être spécialisés en applicant ces composants à différents domaines spécifiques. Cette spécialisation pourrait être rendue possible par la création de librairies de composants sur étagère spécifiques à un certain domaine d'applications (électronique, hydraulique, mécanique, ...) Ceci permettrait alors la synthèse combinatoire de nombreuses briques de solutions. Néanmoins, l'analyse des relations entre composants permettrait alors de limiter le nombre de concepts généré grâce à l'étude d'incompatibilité entre variables de sortie d'un composant et variables d'entrée d'un autre composant devant être connecté au précédent (Kurtoglu et al., 2005).

Dans le cas de la seconde voie, il semble que l'analyse sémantique des exigences basée sur le contexte exprimé de manière textuelle pourraient apporter de nombreux autres avantages que la clarification et la désambiguation de ces exigences. En effet, le processus de clarification et la métrique de similarité entre textes proposés dans cette thèse pourraient être utilisés à l'échelle entière du document de définition des exigences. Ce document est habituellement constitué de différentes catégories dans lesquelles sont regroupées des exigences de même nature (ergonomie, sécurité, ...). L'utilisation de l'analyse contextuelle ainsi que de la métrique de similarité pourraient alors permettre de détecter des exigences appartenant à plusieurs catégories à la fois mais n'étant mentionnées que dans leur catégorie principale dans le document de définition. Ceci permettrait alors d'éviter les oublis lors de l'analyse de ce document dans le cadre du dévelopement de la partie du système liée à une catégorie spécifique d'exigences.

Il semble par ailleurs que de nombreux industriels consacrent actuellement une partie de leurs ressources et accordent un intérêt grandissant à ces deux pistes de recherche.

Tiivistelmä

Tämä väitöskirja esittää laskennallisen kielentutkimuksen sekä tiedon esittämisen tieteenaloilla käytettyjen työkalujen hyödyntämistä kahdessa eri tuotekehityksen konseptisuunnittelun prosessissa; tuotteiden vaatimusten analyysissä sekä konseptuaalisten suunnitteluratkaisujen synteesissä. Konseptisuunnittelun näkökulma, jota käytetään tässä väitöskirjassa, perustuu kirjallisuudesta tuttuihin systemaattisiin tuotekehitysmenetelmiin. Näiden menetelmien kehitys tarjoaa tarkan kuvauksen prosessin tehtävistä, jotka suunnittelutyöryhmän tulee suorittaa onnistuakseen tuotesuunnittelussa. Tästä syystä tämä väitöskirja esittää, että on mahdollista osittain automatisoida tuotekehitysprosessia luomalla laskennallisia malleja tietyistä tuotekehitysprosessin tehtävistä, kuten suunnitteluongelman tarkentamisesta sekä suunnitteluavaruuden tutkimuksesta.

Tuotekehityksessä tuotteiden vaatimukset määritellään identifioimalla ensin eri sidostahojen tarpeet tuotteen osalta, ja sen jälkeen formalisoimalla nämä vaatimusmäärittelyksi. Tämän vaiheen aikana suunnittelijat kohtaavat usein ongelmia erilaisten käyttäjien eritasoisten tarvemääritelmien kanssa. Tutkimus tässä väitöskirjassa keskittyy luonnollisella kielellä (ja erityisesti englannin kielellä) ilmaistuihin tuotevaatimuksiin. Tarpeiden analyysi on toteutettu eri kielentutkimuksen tasoilla; sanastollisella, lauseopillisella sekä semanttisella tasolla. Sanastollisella tasolla käsitellään sanojen merkitystä kielessä, lauseopillisella tasolla käsitellään lauseen rakennetta, eli kielen kielioppia, ja semanttisella tasolla päämääränä on löytää sanojen nimenomainen merkitys kunkin lauseen kontekstissa. Tällaiset käsitteet mahdollistavat etäisyyden laskemisen sanan itsensä sekä sen eri synonyymien välillä. Lisäksi sanojen samankaltaisuuden määrittämiseen kehitettiin metodologia sekä mitta, jotka edelleen selventävät tuotevaatimuksia sekä sanastollisella, lauseopillisella että semanttisella tasolla. Tämä metodologia hyödyntää tutkimusyhteistyökumppaneiden kehittämiä työkaluja.

Tuotekehitysprosessin synteesivaiheessa kehitettiin tiedollinen esitys tarvittavista käsitteistä, jotka mahdollistavat suunnittelukonseptien tuottamisen tietokoneilla. Näihin käsitteisiin lukeutuvat mm. tuotteen toiminnallisuudet, input/output-virta, geneeriset toimielimet, tuotteen käyttäytyminen sekä tuotteen rakenneosat. Myös semanttista atlasta hyödynnetään tässä

vaiheessa mahdollistamaan toiminnallisuuksien yhdistäminen niihin teknisiin ratkaisuihin, joilla kyseiset toiminnallisuudet toteutetaan. Semanttinen atlas toimii rajapintana tiedollisen esityksen eri käsitteiden välillä.

Abstract

This thesis suggests the use of tools from the disciplines of Computational Linguistics and Knowledge Representation with the idea that such tools would enable the partial automation of two processes of Conceptual Design: the analysis of Requirements and the synthesis of concepts of solution. The viewpoint on Conceptual Design developed in this research is based on the systematic methodologies developed in the literature. The evolution of these methodologies provided precise description of the tasks to be achieved by the designing team in order to achieve successful design. Therefore, the argument of this thesis is that it is possible to create computer models of some of these tasks in order to partially automate the refinement of the design problem and the exploration of the design space.

In Requirements Engineering, the definition of requirements consists in identifying the needs of various stakeholders and formalizing it into design specifications. During this task, designers face the problem of having to deal with individuals from different expertise, expressing their needs with different levels of clarity. This research tackles this issue with requirements expressed in natural language (in this case in English). The analysis of needs is realised from different linguistic levels: lexical, syntactic and semantic. The lexical level deals with the meaning of words of a language. Syntactic analysis provides the construction of the sentence in language, i.e. the grammar of a language. The semantic level aims at finding about the specific meaning of words in the context of a sentence. This research makes extensive use of a semantic atlas based on the concept of clique from graph theory. Such concept enables the computation of distances between a word and its synonyms. Additionally, a methodology and a metric of similarity was defined for clarifying requirements at syntactic, lexical and semantic levels. This methodology integrates tools from research collaborators.

In the synthesis process, a Knowledge Representation of the necessary concepts for enabling computers to create concepts of solution was developed. Such, concepts are: function, input/output flow, generic organs, behaviour, components. The semantic atlas is also used at that stage to enable a mapping between functions and their solutions. It works as the interface between the concepts of this Knowledge Representation.

Preface

At the term of this thesis, it feels to me as it has been like walking a long and winding road. With this regard, I would like to first thank my supervisors, Eric Coatanéa and Alain Bernard, for their support all along this path for their trust at every difficult steps, and there was quite a few.

Secondly, this work is dedicated to my daughters, Aura and Ella, who are most certainly my biggest achievements during the past years. It is also dedicated to my wife, Terhi, who help me to stay humble and to keep close to the simple reality of life.

The work context of this thesis is in cooperation between Nantes Centrale Engineering School and Aalto University School of Engineering. I would like to thank all my colleagues from Nantes for their warm welcoming behaviour and, particularly Philippe, Benjamin, Benoit, Joanna and Florent. I am also thankful to my colleagues of Aalto for their support and comments: Tuomas, Mohamed, Galina , Sarayut and Andrea.

Luckily, this winding road is never really ending and even if it feels now like an achievement, it is only a small contribution compared to what is left to do. On this matter, I would also like to thank the reviewers of this dissertation Professors Bernard Yannou and Georges Fadel for their comments on how to improve this work and on its possible continuations.

Espoo, July 4, 2012,

François Christophe

Contents

Preface 21

Contents 23

List of Publications 25

Author's Contribution 27

1 Introduction 31

1.1 Background 31
1.2 Research problem 33
1.3 Aim of the research 34
1.4 Research methods 34
1.5 Scope of the research 38
1.6 Author's contribution 38
1.7 Outline of the Thesis 40

2 State of the art 43

2.1 Design methodologies 43
2.1.1 The systematic approach towards Engineering Design of Pahl & Beitz 44
2.1.2 Conceptual Design for Engineers from French 47
2.1.3 Total Design from Pugh 47
2.1.4 Practices in Systems Engineering 47
2.1.5 Summary of the conceptual design activity in design methodologies 54
2.2 Context and models of Conceptual Design 55
2.3 Requirements Engineering: the initial stage of Conceptual Design 59
2.3.1 Requirements definition 60

2.3.2 Environment representation with System Dynamics 62
2.3.3 Elicitation and Representation of the initial design conditions 68
2.4 The synthesis process 70
2.4.1 TRIZ 70
2.4.2 C-K theory 71
2.4.3 Formal synthesis in Engineering Design 73
2.4.4 Conclusion on synthesis process 73
2.5 Knowledge Representation (KR): Definition and KRs for Conceptual Design 73
2.5.1 Knowledge Representation 74
2.5.2 Logic 76
2.5.3 Conceptual Graphs 78
2.5.4 Ontologies 79
2.5.5 Knowledge Representations for Conceptual Design . 81
2.6 Computational linguistics and Natural Language Processing 84
2.7 Summary of concepts reviewed in state of the art 89

3 Contributions of this thesis 93
3.1 Model of Conceptual Design and Object - Oriented Design paradigm 93
3.2 Requirements Engineering 99
3.2.1 Protocol for semantic disambiguation 99
3.3 Steps towards the automated synthesis of design 102
3.3.1 Knowledge Representation of the concepts used for the synthesis 102
3.3.2 Library of components 103

4 Conclusion and Perspectives 105

Bibliography 109

Publications 115

List of Publications

This thesis consists of an overview and of the following publications which are referred to in the text by their Roman numerals.

I François Christophe, Raivo Sell and Eric Coatanéa. Conceptual design framework supported by dimensional analysis and System Modelling Language. *Estonian Journal of Engineering*, Volume 14, issue 4, pp. 303-316, DOI: 10.3176/eng.2008.4.02, 2008.

II François Christophe, Raivo Sell, Alain Bernard and Eric Coatanéa. OPAS: Ontology processing for assisted synthesis of Conceptual Design Solutions. In *Proceedings of the ASME 2009 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, San Diego, CA, USA, pp. 249-260, DOI: 10.1115/DETC 2009-87776, September 2009.

III François Christophe, Alain Bernard and Eric Coatanéa. RFBS: A model for knowledge representation of conceptual design. *CIRP Annals - Manufacturing Technology*, Volume 59, issue 1, pp. 155-158, ISBN: 9781424432905, DOI: 10.1016/j.cirp.2010.03.105, 2010.

IV François Christophe, Min Wang, Eric Coatanéa, Yong Zeng and Alain Bernard. Grammatical and semantic disambiguation of requirements at elicitation and representation stages. In *Proceedings of the ASME 2011 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, Washington D.C., USA, pp., and other detailed information, August 2011.

V François Christophe, Faisal Mokammel, Thanh An Nguyen, Eric Coatanéa, Mohamed Ba Khouya and Alain Bernard. A Methodology for Syntactic, Lexical and Semantic Clarification of Requirements in Systems Engineering. *Advanced Engineering Informatics*, Special Issue on Modeling, Extraction, and Transformation of Semantics in Computer Aided Engineering Systems, June 29^{th} 2012.

Author's Contribution

Publication I: "Conceptual design framework supported by dimensional analysis and System Modelling Language"

In this journal paper, author's contribution consists in showing how elements of the System Modeling Language (SysML) match with the necessary concepts of knowledge in Conceptual Design. Such necessary concepts are described in Gero's Function-Behavior-Structure model (FBS model) of Knowledge for Conceptual Design. Additionally, the sequence of development of a system are represented with a framework using SysML and presenting the steps to follow during the Conceptual Design process.

Publication II: "OPAS: Ontology processing for assisted synthesis of Conceptual Design Solutions"

The author developed an ontology enabling the synthesis of concepts of solution in Conceptual Design. This ontology provides a dynamic mapping between the functional representation of a technical system and generic components used for the development of this system. Such mapping is dynamical as it is realised through the use of a semantic atlas. The connection between a technical function and generic components is achieved when function and components share common links of meaning in the semantic atlas. The dynamical aspect of this mapping is due to these links of meaning: when the expression of a function in Natural Language is modified, e.g. precised or redefined, the semantic links between function and components is suggest to changes.

Publication III: “RFBS: A model for knowledge representation of conceptual design”

Author’s contribution in this article is a modification of Gero’s FBS representation of Knowledge for Conceptual Design into Requirement-Function-Behavior-Structure model (RFBS). This RFBS model presents two new ideas compared to Gero’s FBS model: the concept of Requirement and a direct relation between Function and Structure.

First, Requirements embeds more than the concept of Function. Functions represent the expected actions from the system, i.e. the services it would provide to its users, regardless of the performance of achieving this service, the quality of this service or constraint to take into account while realising this service. Therefore, Requirements are needed as a concept giving the possibility to express the non-functional expected features from the system. Second, this article shows a link between the expression of Function and Structure through a semantic analysis. In other words, the analysis of meaning of the functional representation of what is expected from a system allows the synthesis of structural components enabling these functions without consideration, at first, of the expected behavior of the system.

Publication IV: “Grammatical and semantic disambiguation of requirements at elicitation and representation stages”

In this article the author contributed to the part dealing with semantic disambiguation of requirements expressed in Natural Language whereas the grammatical disambiguation is realised by ROM (Recursive Object Model), a model for clarified the syntactic ambiguities in requirements, developed by Prof. Yong Zeng.

The semantic disambiguation of terms used in requirements expressed in Natural Language consists in extracting the meaning of these terms within the context of the requirements. The difficulty lies in the fact that words of a natural language are polysemous, i.e. they possess several meanings. The main principle for semantic disambiguation is to reduce the possible meanings of a word due to its use in the sentence and the words it is surrounded by in the context of this sentence.

Publication V: “A Methodology for Syntactic, Lexical and Semantic Clarification of Requirements in Systems Engineering”

The author contributed to the development of a metric for measuring the lexical and semantic relevance of an answer to a question. This work is integrate with ROM and ROM Question asking process. The combination of these two approaches consists in a methodology clarifying the expression of requirements, mainly in terms of expressiveness but also partly in terms of the consistency of requirement and their completeness.

1. Introduction

1.1 Background

The product development process has to tackle multiple requirements of quality, costs and time-to-market. To that effect and also because it involves several actors providing their vision about the services that the future product should accomplish, engineering design is a complex activity for which processes need to be formalized (Deneux, 2002).

Conceptual design is this part of engineering which starts,in most cases, with the expression of a problem encountered. In the very beginning of design, this problem is often expressed in the form of natural language and sketches with the aim of explaining the origin of this design problem. The first aim in the conceptual design phase is to refine this weakly defined problem in a more formal manner in order to avoid ambiguities in the understanding of this problem. This phase corresponds to the analysis of the needs and is also called requirements engineering. There are mainly two possibilities within the creation of a design problem. Either this problem is brought to the designing team as a request from customers or the designing team seeks to create a market need in order to acquire a competitive advantage with regard to its competitors. In the first case, where the deign problem comes from customers, requirements engineering is used to make sure that the team of designers clearly understood the needs expected from customers so as for them to provide design solutions that are compliant with these required needs but also solutions that do not provide more than what is expected from customers for business reasons. Therefore, the initially weak defined problem needs to be refined by avoiding, as much as possible, any ambiguity in the meanings of the terms used to specify the requirements. In the second case, where the designing team tackles a design problem as the creation of a market need, it is important to dress a rather exhaustive review of the actual status of existing solutions to previous problems involved in the same design context. This case presents the heavy complexity of the cognitive load associated with the design activity as well as the co-evolution between a design problem and its solutions (Dorst and Cross, 2001) (Zeng, 2004). Many design researchers argue, following Schön, that ambiguity is at the heart of creativity and creativity at the heart of design (Schön, 1983). In view with this thesis, this statement is valid for the parts of design involved with synthesis and, thus, involved with creativity. Nevertheless, requirements engineering is a part of design involved with the analysis of the design problem. Additionally, as just stated, the output of this refinement should produce requirements denuded of ambiguity in order to set a clearly defined goal for the specific, context dependent, design activity. After requirements engineering, conceptual design is formed of mainly two processes: the synthesis of concepts and, the evaluation and comparison of these concepts. These two processes are seen as sequential or concurrent according to different guidelines and methodologies. Nevertheless, they are crucial processes of conceptual design and are always precisely defined in methodologies (Tomiyama et al., 2009).

Guidelines and methodologies for conceptual design were provided by the design community (Tomiyama et al., 2009). In fact, in order to understand the conceptual design process, researchers described this process through models and representations of the knowledge involved during that stage of engineering design (Gero, 1990) (Tomiyama et al., 2003) (Kitamura et al., 2007). These models became very precise and suggest a systematic application of them for a successful conception. Methodologies of conceptual design also attach a particularly important part to the description of the synthesis process (Chakrabarti, 2002) (Antonsson and Cagan, 2001). Nowadays, with the evolution from document-based engineering into model-driven engineering, these models and representations could be implemented as digital meta-models. The formal description of these methods and models enables the possibility to implement these methods and models in forms of computer applications. This makes possible the automation of parts of the design process.

The systematic aspect of conceptual design guidelines and methodologies and their digital implementation as digital meta-models suggest the possibility to bring up computers from assisting design engineers into being considered as full members of a design team having abilities to understand design problems and to propose relevant concepts of solution.

1.2 Research problem

The research background raised several questions concerning the possible automation of parts of the conceptual design process.

First, what are the general concepts associated with conceptual design in methodologies? How to represent these concepts and their relations for a better understanding of the conceptual design activity from designers and computers? From the numerous methodologies proposed for conceptual design, these questions formulate the need for unity and definition of the main concepts being handled during the conceptual design stage. Models of these concepts and their interactions shall also be studied.

Second, how to help designers in refining the initially weak defined design problem into more formal and clear of ambiguity design requirements? What are the possible tools for computers to provide this help in a rather systematic manner? As initial requirements are expressed in the form of natural language, it is difficult for computers to process them. The automation of this refinement process should be studied in light with the possibility of multiple meanings of the words used in the expression of requirements.

Third, how to provide sufficient knowledge for computers to understand the concepts used in the synthesis process of conceptual design? What is the knowledge needed for computers to synthesize relevant concepts of solution? In order for computers to be capable of proposing relevant and useful concepts of solution to designers, computers should have access the knowledge of designers as well as having access to several previous designs. The link made by designers between the design problem they are facing and existing products and working principles reflects the complexity of the mind and of their knowledge. Giving this linking ability to computers is thus a challenging issue.

1.3 Aim of the research

The aim of this thesis is to consider the use of computers in assisting designers in their activity from the very beginning. Therefore, this research aims at applying automated methods and tools from computer science to the elicitation and representation phases of requirements engineering and, the synthesis of concepts of solution. These tools should assist designers in obtaining requirements in a more formal representation than the initially weakly defined design problem.

Additionally, this research investigates the possibility to express the key concepts of the knowledge used by designers during the synthesis process of conceptual design. The expression of this knowledge into a computer readable format could probably lead to computers proposing non trivial concepts of solution and assists them during the exploration of the design space which is the synthesis process of conceptual design.

In a more general manner, this research aims at applying the latest concepts from Artificial Intelligence, Knowledge Representation, Semantics and Computational Linguistics into the development of computer aided tools for requirements elicitation and synthesis phases of the conceptual design process.

1.4 Research methods

The research methods applied in this thesis make combination of methodological research related to conceptual design and disciplines from Computational Linguistics and Artificial Intelligence.

At first, the approach consists of analysing the differences and similarities within the multiple definitions of systematic conceptual design process from the literature (Motte, 2008) (Tomiyama et al., 2009). This unifying analysis enables the identification of the concepts and terms used in conceptual design. Additionally, the relations between these concepts is observed through a knowledge representation schema of conceptual design (Gero, 1990) (Umeda and Tomiyama, 1995) (Gero and Kannengiesser, 2004). After that, this analysis enables use to provide consensual definitions for the concepts used at the stages of conceptual design considered in this thesis, namely requirements elicitation and synthesis of concepts of solutions.

In the vision of a systematic application of design methodologies, conceptual design begins with the definition of a design problem. This design problems expresses the emergence of market needs for a new artefact to be designed. Because this problem is at the interface between customer/user needs and the team involved in the creation of an artefact as a solution, it is by nature an ill or weak defined problem (Simon, 1996). This property of the initial design problem is due to the lack of information about the actual needs. Therefore, this initial problem needs to be refined into a more formal representation of requirements. The understanding of more formal representation of requirements suggested in this thesis is the following. Requirements shall contain information about the overall services the designed artefact shall provide. In this thesis, the term service takes the meaning proposed by Vargo and Lusch theory called Service-Dominant-Logic (SDL) (Lusch et al., 2007). More precisely, they address that an artifact has an interest only when in use, i.e. when it is providing services to users. Overall services are also called service functions in the literature (Coatanéa, 2005) (Dardy et al., 2003) (Miles, 1961).
The concept of function is a key concept in this thesis and has different definitions and representations in the literature (Micouin, 2006)(Coatanéa, 2005)(Hirtz et al., 2002). In this thesis, we differentiate two types of functions: service function and technical function. Service function is considered at the highest level of analysis of an artifact. Basically, it is similar to the black-box viewpoint or the highest level of building block from EAI 632 standard (ANSI/EIA-632-1998, 2003). This black-box viewpoint on a product focuses on what this product does. As soon as the product is being analyzed further into its composition, functions should be named technical functions as they express how the product is functioning and not only

what it does. In turn, components of a product can also be seen as a technical system on their own providing service functions.

In addition to expressing the needs in terms of service functions, requirements should contain information about the environment of the designed artefact according to the different phases of its life-cycle. In fact, researchers in engineering design (Zeng, 2004) and in Systems Engineering (Weilkiens, 2008)(Estefan, 2008) consider that, during engineering processes, the team of designers should collect information about the so-called “system as-is” in its environment in order to be able to engineer the “system to-be”. This represents the recursive aspect of the engineering design activity. The viewpoint of this thesis is that such information about the environment and previous artifacts answering similar design problem should be collected during the elicitation of requirements. In this vision, the performances of previous concepts tested in the current environment could be used for setting quantitatively the required performances of the “system to-be”. For example, in past decades, concerns about human emissions of carbon-oxides (CO and CO_2) in the atmosphere generated new design problems in various industries. In the case of transportation and, more particularly, automotive industry, engineers are currently considering several options for reducing drastically the consumption of hydrocarburant which causes such release of gases in the atmosphere. As a result of testing existing products, new standards and restrictions have appeared for car manufacturers in terms of CO and CO_2 release.

Requirements should also describe the relations between the different stakeholders that are interacting with the artefact. In fact, stakeholders do not necessarily have relations between themselves. Nevertheless, during elicitation of requirements, designers try to extract wishes and needs from various stakeholders. Requirements express these wishes and needs from various stakeholders. These wishes and needs have the product in common. It is thus important to analyze the semantic relations between requirements as it could show inconsistencies, conflicts or contradictions within the set of requirements and the expected performances of the future design. Therefore, researchers in Systems Engineering suggest bringing stakeholders together during common meetings while defining requirements (Weilkiens, 2008).

Finally, requirements shall describe the expected performance of the artefact to provide the service functions in a qualitative manner and, if possible in a quantitative manner. This paragraph described the input and output of the requirements engineering process.

The synthesis process follows requirements engineering in conceptual design. According to design methodologies (Ulrich and Eppinger, 2011) (Pahl and Beitz, 2007) (Otto and Wood, 2001), the synthesis process starts with a functional decomposition of the service functions into technical functions needed in order to implement the service functions. This approach is called top-down approach. It results in the creation of function trees. The idea from the top-down approach is to decompose a complex problem expressed with service functions into simpler sub-problems. Methodologies also suggest that functions at the bottom of the function trees can be immediately linked with basic components, structural solutions, that fulfil this technical function (Coatanéa, 2005). The following step consists of integrating together these basic components in order to obtain complete structural concepts of solution. This approach is, in reverse, called bottom-up approach.

After this overview of the methods used in the research focus, this thesis investigates the potential of existing tools from Computer Science disciplines that could be used to partly automate and assists designers during these conceptual design processes. As requirement elicitation deals with refining problem expressed in natural language, the research investigated practices from computational linguistics and natural language processing. In this field, researchers have created metrics on a semantic space which enables the evaluation of distances between the meanings of a word according to its synonyms (Ploux and Victorri, 1998).

The background of this research and the methodologies presented highlighted the importance of the cognitive load associated with the design activity. Naturally, this research dresses a review of the practices used to formalize knowledge from the discipline of Knowledge Engineering and Knowledge Representation in the field of Artificial Intelligence.

These elements of research are further described in Chapter 2 presenting the state of the art and are used to implement the method presented in this thesis.

1.5 Scope of the research

This research takes its focus in formalizing parts of the conceptual design process with the aim to provide machine readable knowledge about this process. This formalization starts at the elicitation and formulation of requirements initially expressed in natural language. In a second phase, this research addresses the possibility for machines to synthesize concepts of solution.

This research is limited to the synthesis of basic generic elements to be used and combined in the design of the product. It does not address the multiplicity of these elements or the interactions between them. Nevertheless, possible solutions to tackle these issues emerged during this research work and are proposed as potential future research directions in conclusion of this thesis.

1.6 Author's contribution

From the study of the methodologies and models of conceptual design, this research lead to contributions at different levels of this process.

At requirements elicitation level, the research enabled the creation of an computer assistant for the disambiguation of requirements. In fact, at that stage of design, the sources of misunderstandings can be numerous. There can be misunderstandings between customers and the design team, between consumers or users and the design team, and within the design team itself. These sources of misunderstandings are all barriers for the expression of requirements in a clear manner. This tool provides the designing team with the meaning of each word used in the initial expression of the problem according to the context of use of this word in the sentence. In the case where a word remains with multiple possible meanings, the assistant gives incentive to designers for asking more questions to stakeholders about this specific term. In this sense, this contribution is valuable as it focuses designers attention on unclear concepts which might be at the core of the design problem.

At the synthesis level, an ontology unifying taxonomies of standard functions, generic organs, energies, variables and physical components was created. This ontology is compliant with ontologies from Systems Engineering. The use of a semantic atlas to create links between technical function and associated generic organs makes these links more dynamic. Actually, in most previous cases of automated synthesis from the literature, these links were usually defined as relationships in the knowledge base of the computer application. This time, this link is created through an online semantic atlas. This new feature gives flexibility to the mapping between functions and generic components. This flexibility depends on the corpus of the semantic atlas and, can therefore be different when applied to different technical domains.

On a more general level, the previous contributions and findings about semantic function-component connection forced the creation of a new knowledge representation of conceptual design and the interactions bricks of knowledge. This model includes Requirements as an important concept of the model. It also contains the concept of genericity, with a concept called Generic Structure. The Generic Structure is used to bridge Function and Physical Structure. In fact, the Generic Structure is similar to the concept of abstract class in object-oriented paradigm. Additionally, this research presents the association between this model and the System Modeling Language (SysML), this makes space for thinking engineering design with the object-oriented paradigm. This paradigm is very interesting for designing as it enabled automatic code generation in the field of software development.

1.7 Outline of the Thesis

This thesis is organised as follows.

Chapter 2 positions the scope of the thesis, i.e. Conceptual Design, within the bigger picture of the design activity and presents a state of the art of methodologies and models proposed for designing concepts in a systematic manner. Figure 1.1 presents the scope of the research and the research problem within the broader picture of the design activity. The subjects in red are the ones to which we attach particular attention to solve the research problem in this thesis.
Narrowing down to the subject of this thesis, two sections present, first, the current state of developments in the field of Requirements Engineering and, second, the state of methodologies and automation techniques used for design synthesis. This chapter also presents a broad vision about existing computer tools in the field of Knowledge Representation, Semantics and Natural Language Processing.

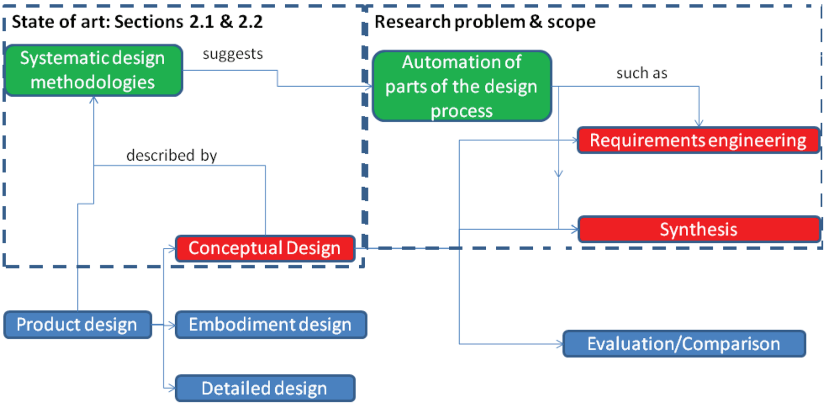


Figure 1.1. Scope of the thesis and research problem presented within the design activity

Chapter 3 presents the contributions of this thesis with regard to:

- a model of knowledge for conceptual design

- semantic analysis of textual requirements and their clarification

- ontology processing for the assisted synthesis of conceptual design solutions

More specifically, Section 3.1 presents the new model of conceptual design as well as the shift in paradigm suggested by this thesis.

Section 3.2 shows the protocol developed for disambiguation of terms at requirement elicitation stage. This chapter also presents the combination of the semantic disambiguation with a grammatical manner of transforming sentences into formal graphs. Figure 1.2 presents this contribution.

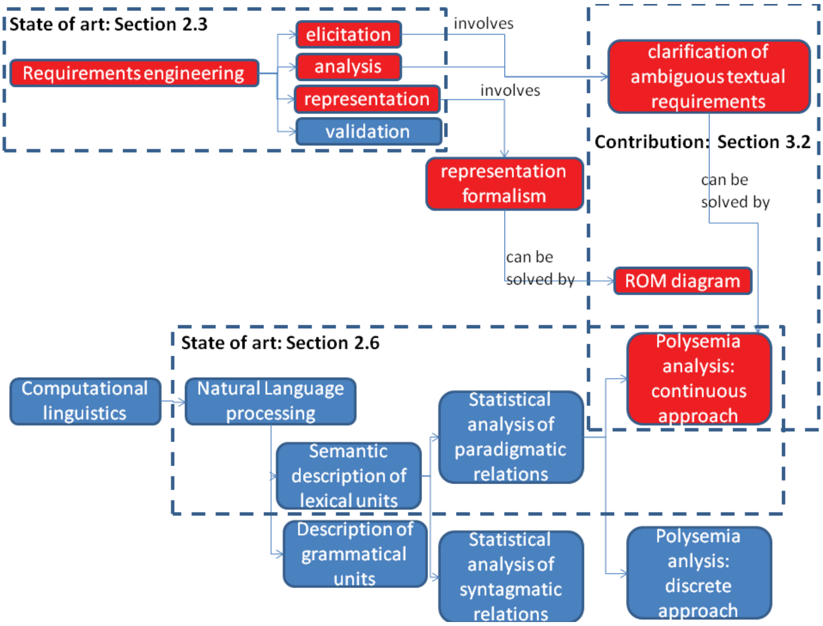


Figure 1.2. Contribution in disambiguation of textual requirements

Finally, section 3.3 presents the algorithmic protocol for associating design problems with early concepts of solution. This section presents how taxonomies of functions and generic components can be linked due to semantic relations. This link enables computers to provide preliminary concepts of solution corresponding to the technical functions required by the product.

Finally conclusions are drawn regarding to the overall pertinence of this research. The evaluation of the contributions has been achieved with comparison to other tools provided by the community. Nevertheless, we aim at testing them on concrete cases in order to obtain empirical data enabling the assessment of this contribution compared to others.

2. State of the art

This chapter introduces a review of the current practices related to requirements engineering and synthesis phases of conceptual design. First of all, we provide a view on the entire design process through the review of different design methodologies proposed in the literature (Section 2.1). In addition, the design process is analysed through a systems engineering viewpoint and a review of systems engineering practices is proposed. Section 2.1 concludes with a vision of the design process adopted in this thesis. This section positions the focus of this thesis in view with the entire design process. Secondly, we approach this research work with the analysis of the context and different models of conceptual design (Section 2.2). Thirdly and fourthly, we present the core subjects of this thesis, requirements engineering and design synthesis (Sections 2.3 and 2.4, respectively). Section 2.5 investigates practices in the discipline of knowledge representation in order to capture designers' knowledge into computer readable format. Section 2.6 addresses practices of natural language processing and computational linguistics so as to understand designers' language. These two sections provide the building blocks used to tackle the semantic and linguistic aspects of the research problem. Finally, Section 2.7 dresses a summary of the main concepts used as basis for this thesis and their connections.

2.1 Design methodologies

This chapter aims at gathering a review of design methodologies developed through the years. First books about design methods or methodology appeared in the beginning of the 1960s (Asimow, 1962) (Simon, 1991). Nevertheless, as stated in (Bayazit, 2004) (Cross, 2007), systematic approaches towards engineering design only emerged in the 1980s with, for instance, systematic design methodologies from Hubka in 1982 (Hubka and Eder, 1996), Pahl & Beitz in 1984 (Pahl and Beitz, 2007), French in 1985 (French, 1999) and Pugh in 1991 (Pugh, 1991), only to cite a few. This thesis is inscribed in the continuation of this school of thoughts as systematic means reproducible and thus, possible to automate. In this section, we have extracted the descriptions of the design process from these methodologies: Pahl & Beitz, French, Pugh, Ulrich & Eppinger, Ullman and Otto & Wood. This study will allow us to point out the differences and similarities between these methodologies with a special focus regarding the conceptual phase of design.

2.1.1 The systematic approach towards Engineering Design of Pahl & Beitz

Pahl & Beitz were probably of the first to propose the adoption of a systematic approach towards the design activity (Pahl and Beitz, 1984). Several editions of this book have followed and the current one dates from 2007 but the process described remains more or less unchanged (Pahl and Beitz, 2007). To a certain extent, this proves the stability and the strength of the methodology as it is still standing after years of reviews and criticisms (Dekker, 1995) (Cross, 2007) (Bayazit, 2004). In fact, it is often used as a corner stone for numerous other researchers in the community (Motte, 2011). In their viewpoint, the design process is seen as a general problem solving process which can be described precisely as a procedure shown Figure 2.1. Pahl and Beitz describe this procedure with four major sub-processes: planning and clarifying of the design task, conceptual design, embodiment design and detail design. Each of these sub-processes provide respectively the following outcomes: list of requirements (design specification), concept (solution principle), preliminary and definitive layouts, and product documentation. Each outcome is used as input for the following sub-process.

Concerning the conceptual design phase, Pahl & Beitz describe it as in Figure 2.2. The conceptual design process starts with the list of requirements as input of this process. From this list of requirements, the engineering team should use abstraction in order to extract essential problems. In this viewpoint, abstraction means *"ignoring what is particular or incidental and emphasising what is general and essential"*. Such abstraction is realised through functional analysis of the requirements.

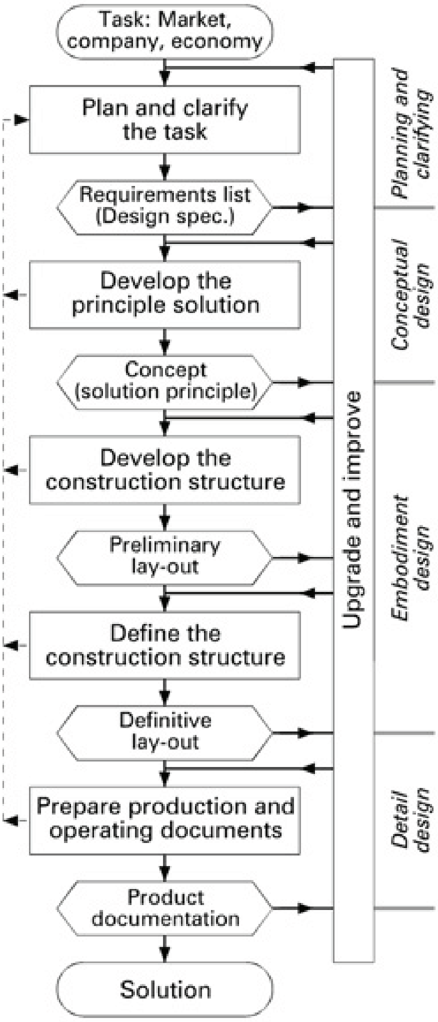


Figure 2.1. Design process from Pahl & Beitz (Pahl and Beitz, 2007)

This analysis is then used to establish functional structures of the technical system. This is realised by breaking down the overall function of the system into technical sub-functions. These initial steps of conceptual design are compliant with the functional analysis proposed in (NF-X50-151, 1991) and value analysis proposed in (Miles, 1961). The search for working principles that fulfill subfunctions follows the functional analysis of the technical system. This is a first phase into the synthesis of concepts. Once working principles have been found for each subfunction, they need to be combined in order to form working structures of the technical system. In this stage, many possible combination appear. It is important to note that Pahl & Beitz introduce a selection of so-called "suitable" solutions after the combination of working principles. Indeed, some combination of working principles may be impossible to realise with current technology. This is a preliminary selection within the synthesis stage. Nevertheless, in our viewpoint, this preliminary selection may be dangerous for designers in search for optimal design. Therefore, we advocate to keep track of all possible working principles as they might be possible to combine in the future with the appearance of a new technology. In general, in this thesis, we advocate for the complete separation between synthesis and evaluation of concepts in order to enable a more objective evaluation of all possible concepts found at synthesis stage. In their description of the conceptual design process, Pahl & Beitz evaluate variant concepts according to technical and economic criteria. This phase is called evaluation and enables taking decision about the principle solution, e.g. the concept, that will be further developed at embodiment and detailed design stages.

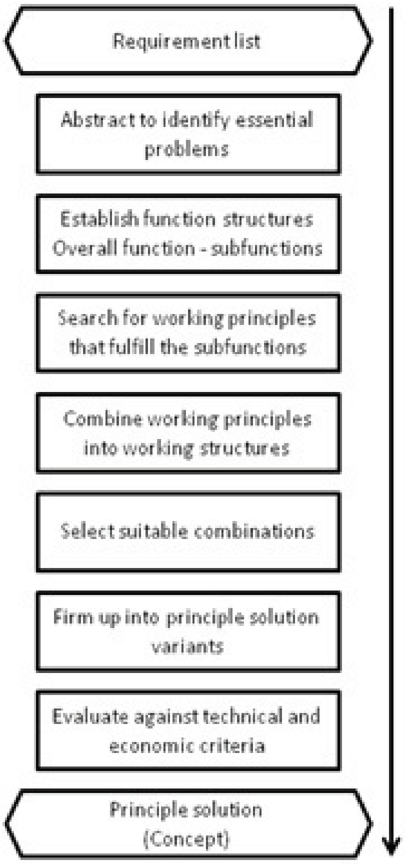


Figure 2.2. The conceptual design phase (Pahl and Beitz, 2007)

2.1.2 Conceptual Design for Engineers from French

French is often referenced as one of the first to define the boarders of conceptual design (French, 1999) in the field of mechanical engineering with the first edition published in 1971. In his book, conceptual design for engineers, French points out that conceptual design is the phase in design that generates the greatest demands for the designing team. Even though, it is mostly focused on mechanics, French highlights the facts that conceptual design brings together economical aspects along with engineering and production methods. The description of the conceptual design process is general enough to be applied in other fields of engineering.

According to French view on conceptual design, it is composed of four major activities : the combination of ideas, design optimisation, insight on the physics involved with a particular product, and analysis of costs.

2.1.3 Total Design from Pugh

In the total design theory, Pugh introduces a very important view of the design activity: design is highly inter-disciplinary. For instance, some of the typical disciplines involved in electronic product development can be considered as the followings: industrial design, graphic design, ergonomics, electronics, mechanical design, electromechanical design, software, information technology (Pugh, 1991). This view on the multiple disciplines involved in the design of artefacts points out the high complexity of the design activity.

This viewpoint towards the design activity emphasizes on the communication between practitioners of different disciplines. Nevertheless, the core process of the total design activity as described in Figure 2.3 does not cover the means for these practitioners to communicate with each other. We will see further in this chapter that applying Systems Engineering viewpoint on a product attends to cover this communication issue by providing models, definitions of terms and language for the common understanding of all the design practitioners.

2.1.4 Practices in Systems Engineering

In mechanical engineering, the conceptual design process is still poorly assisted by computers. In other engineering disciplines, such as computer science, techniques and computer tools in conceptual design are more de-

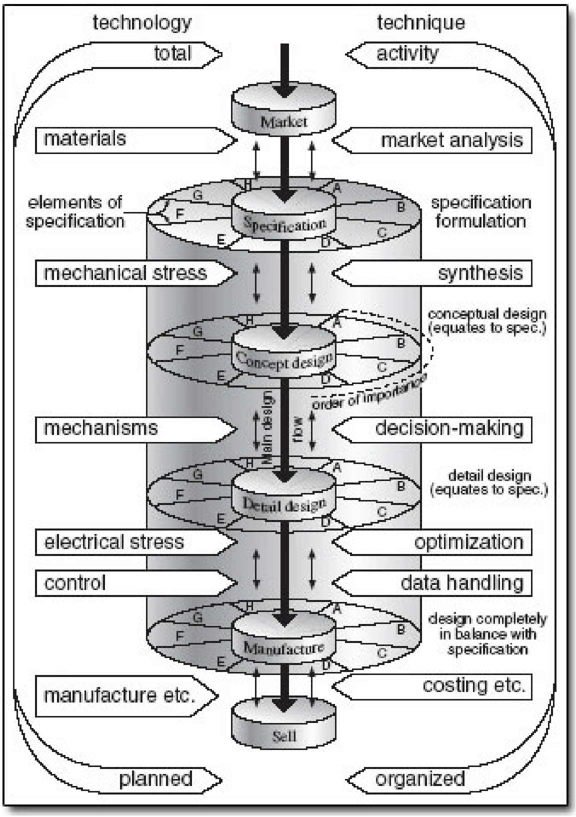


Figure 2.3. The total design activity model from Pugh

veloped and allow spending more time on the definition and specification of the design problem in order to enable faster developments at the stages following conceptual design.

Definition of system

A technical system is defined as a set of interrelated components which interact with one another in an organised manner towards a common purpose which would be unachievable by the individual components alone. The components of a system may be diverse consisting of persons, organizations, procedures, software, equipment or facilities.

Systems are composed of components, attributes and relationships. These are described as follows:

Components are the operating parts of a system consisting of inputs, processes and outputs. Each component of a system may assume a variety of values to describe a state of the system as set by some control actions and one or more restrictions.

Attributes are the properties or discernible manifestations of the components of a system. These attributes characterise the system.

Relationships are the links between components and attributes.

As a set of interrelated components, a system has the following properties:

- The properties and behaviour of each component of the set have an effect on the properties and behaviour of the set as a whole

- The properties and behaviour of each component of the set depends on the properties and behaviour of at least one other component of the set

- Each possible subset of components has the two properties listed above; the components cannot be divided into independent subsets

The paradigm expressed in these properties is holism: the whole system accomplishes more than its parts.

Systems Engineering

There is currently no commonly accepted definition of Systems Engineering (SE) in the literature. Definitions are usually based on the background and experience of the individual or the performing organization and, therefore may vary in viewpoint. Nevertheless, the definition we consider here in this thesis is the following.

Systems Engineering is a profession, a process, and a perspective as illustrated by these three representative definitions:

- SE is a discipline that concentrates on the design and application of the whole as distinct from the parts. It involves looking at a problem in its entirety, taking into account all the facets and all the variables and relating the social to the technical aspects (Ramo, 2007)

- SE is an iterative process of top-down synthesis, development, and operation of a real-world system that satisfies, in a near optimal manner, the full range of requirements for the system (Eisner, 1996)

- SE is an interdisciplinary approach and means to enable the realization of successful systems. It focuses on defining customers' needs and required functionality early in the development cycle, documenting requirements, and then proceeding with design synthesis and system validation while considering the complete problem (INCOSE-SEH-WG, 2004)

Standards of SE.

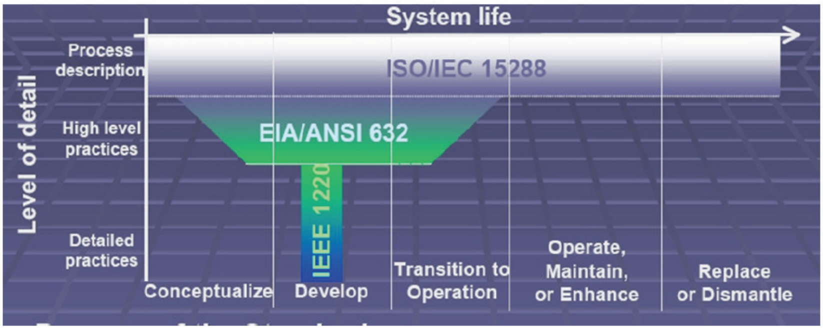


Figure 2.4. Position of leading SE standards

From these definitions we note the emergence of certain keywords such as: interdisciplinary, iterative, socio-technical, and wholeness.

Definition of complexity in Systems Engineering

We as humans, we are legitimately asking to though to help us to dissipate fogs and darkness in our understanding of the world, to put order and to reveal the laws that are governing the real. The word complexity on the opposite can only express our embarrassment and confusion, our inability to define in a simple manner, to name clearly, to put order in our ideas. Historically the scientific knowledge has been conceived as a manner to dissipate the apparent complexity of the phenomena in order to reveal the simple order by which they are ruled. Each type of knowledge is operating by selection of the significant data and rejection of non-significant data. This is done more precisely by separation (distinction or disjunction); unification (association and identification); development of a hierarchy (central, secondary) and centralization (according to a core set of key concepts). These operations that are using the logic are in fact governed by the paradigms that are governing our manner of thinking. These principles are governing our vision of the thinks and of the world. We are most of the time not conscious of it. This paradigm which is leading our occidental way of thinking since the 17th century has been formulated by Descartes. This is the paradigm of simplification which is governed by the principles of disjunction, reduction and abstraction. This paradigm has permitted great progress of knowledge both in science and philosophy but its harmful effects have just started to be apparent during the 20th century. This disjunction is limiting the communication between the different fields of science as well as between philosophy and science. In addition, the disjunction is reducing the complexity to the simple (reduction of the biologic to the physic, reduction of the complex interaction between machine and object to the design of the machine). This type of knowledge is necessary basing its rigour and effectiveness on measure and calculus but this is leading to a vision of reality governed by equations and formulas. This is simplifying the complexity by unifying abstractly and by avoiding diversity. In some other cases diversity is juxtaposed without being able to conceive the unity.

The pathology of the contemporary thinking approach is oversimplifying the real and is necessary leading to oversimplified actions. This pathology is present in engineering design too and is leading to poor machines or services unable to consider the complexity of the various interactions with the environment and to evaluate the impacts. There are major issues related to the epistemology of the classic science that are also reflecting in the engineering design. First, there is interdependence between the subject (i.e. the design project and problem, the engineering team with its culture, knowledge, organization, period of history) and the object (i.e. the machine or the service resulting from the design process), second there is fuzziness in the knowledge, third the apparition of logical contradictions in the physical description of the object of the design process (i.e. for example there is an internal contradiction in desiring of having a car which simultaneously maximize its cruising range and maximize its carrying capacity). The engineering design process has to deal constantly with the complexity of the real world which is not addressed by the simple categories of the classical science. The system theory offers three virtues that are to consider the concept of system as a fundamental element of the theory. The system is more than the sum of its parts. The concept of system is not real neither formal. This is an ambiguous concept. The spectrum of the theory is transdisciplinary. From the theory of open systems, it can be said that the laws of systems do not rely on equilibrium but instead on disequilibrium constantly compensated or dynamically stabilized. The main important consequence is that the understanding of an open system should not be considered only in the system but also in its relation with its environment. This relation is not a simple dependence but the relation is really part of the system. This is why in engineering design the study of the environment is so important.

The consequence of this is that it is difficult to study systems as entities that can be isolated from the environment. Like in the evolution theory, we should consider engineering design as the design of the interactions between the systems and their environments. The relation between the systems and the environment is simultaneously material/energetic and organizational/informational. This relation is deterministic but also random.

SysML for the Conceptual Design of systems

For instance, the use of semi-formal language from the early stages of specification of the design problem eases the generation solutions in computer science. Unified Modeling Language, UML, has been created in this sense and is currently of great use in the computer science community as well as in the industry.

Current efforts in the field of Systems Engineering are put in order to provide the same type of language with application to mechnical and electrical engineering. Last significant developments in systems engineering is the new modeling language – SysML (System Modeling Language) derived from UML 2.0 (Unified Modeling Language). SysML 1.2 specification (OMG, 2010) was published in June 2010 by the Object Management Group (OMG). The OMG is composed of many industry leading corporations and organizations. This new language brings closer software design concept and product design, enabling the natural synergy of multidisciplinary design of products (e.g. software, mechanics, electronics and others) at the very beginning of the design process and continuing to support it through the design lifetime. SysML specification is defined by using UML 2.0 specification techniques. These techniques are used to achieve the following goals.

1. Correctness

2. Precision

3. Conciseness

4. Consistency

5. Understandability

SysML concept is similar to software design techniques, but expands it in several ways. As engineering design, mechatronic product design is not a pure technical problem anymore. It has become a rather complex activity, which needs to also involve artifacts, people, environment, market, in addition to hardware and software components. In order to reach an understanding, all these aspects have to be modeled in the same methodical way. SysML tries to provide the generic language and environment to support complex systems engineering design process. In general, SysML diagrams are divided into three main groups, Requirements, Structure and Behavior, as shown Figure 2.5.

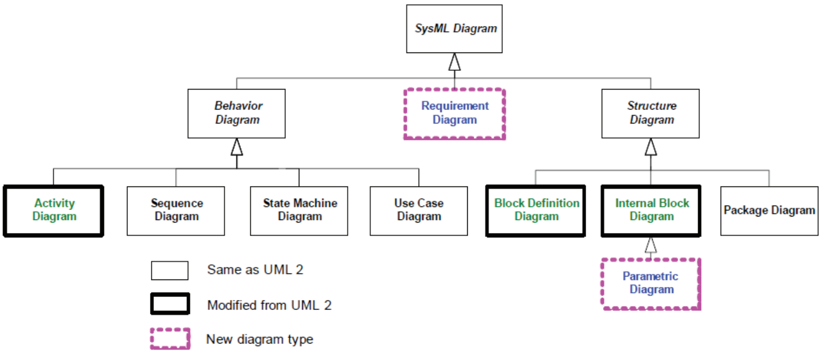


Figure 2.5. The SysML diagrams taxonomy

Depending on the design concept some diagrams can be used for different purposes. A widely used example is Use Case diagram which describes the system functions or services and can be successfully used for clarifying system requirements as well as main activities. It is also common to regroup SysML diagrams into four main pillars where parametric diagram becomes a separate group. Figure 2.6 shows the generic diagrams in four pillars.

By this brief presentation of SysML, we have shown that the diagrams provided by the language fit the description of Function (with Use Case, Block Definition and Internal Block diagrams), Behavior (with State-Machine, Activity and Sequence diagrams) and Structure (components, Packages and Internal Block diagrams). Table presents the partial coherence between the FBS model and SysML diagrams. Nevertheless, SysML provides more than Gero's FBS because it contains Requirement and Para-

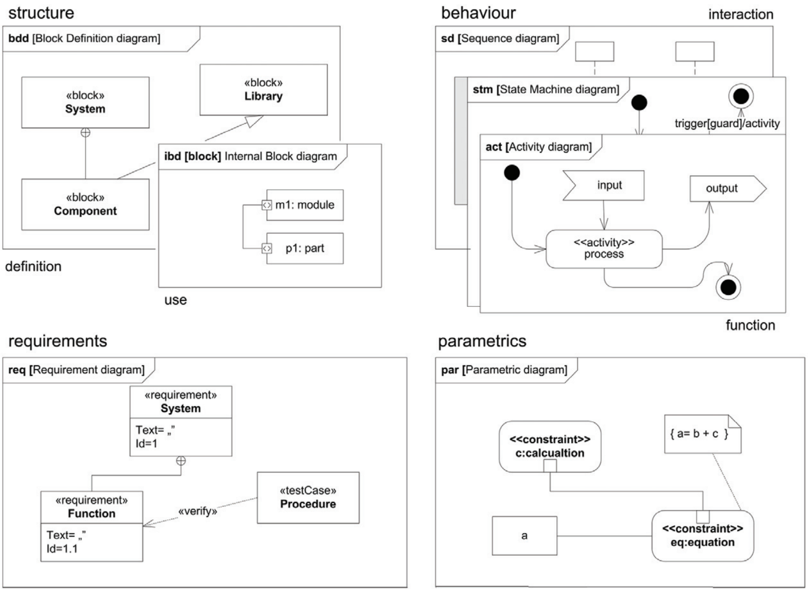


Figure 2.6. The four pillars of SysML

metric diagrams. Additionally, one of the strength of SysML is that it is a modeling language, and, as such, engineering becomes now model centered in contrast with document centered. First of all, documents can now be generated automatically but, most of all, model based engineering allows the verification and validation of the coherency of each phase of systems engineering. This coherency is insured and maintained by the coherency between each models of a project. This coherency can even be verified nowadays by computer tools called model checkers. Additionally, model-driven engineering enables the use of design patterns, stereotypes, specific profiles and libraries for reuse which was not possible with document based engineering.

2.1.5 Summary of the conceptual design activity in design methodologies

According to the literature review realised in previous paragraphs, it is important to notice that methodologies are consistent with the terms used to define the different stages of conceptual design. In fact, except from slight divergence on the boarders of the conceptual design frame, each methodology follows a slightly similar pattern. Other methodologies also describe generally the same phases in the design process (Ulrich and Eppinger, 2011) (Otto and Wood, 2001) (Ullman, 2002) (VDI, 1987). Figure

2.7 presents conceptual design within the overall design process.

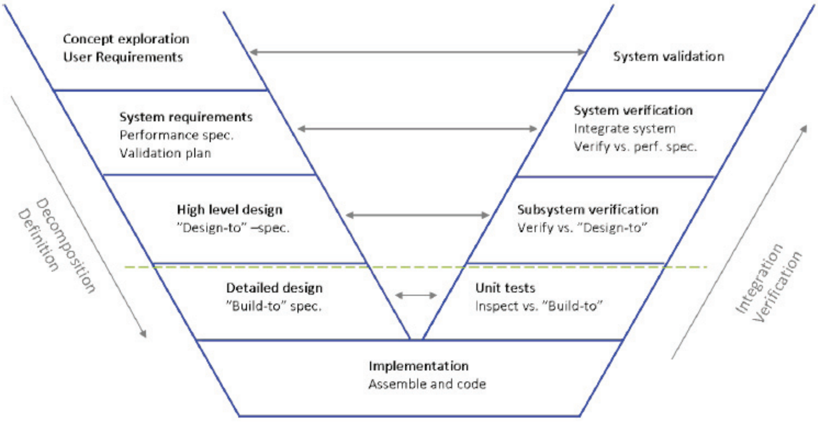


Figure 2.7. The V-model of engineering design

This was already pointed out by Yannou in his description of the conceptual design process (Yannou, 2000). According to our understanding, the conceptual design activity can be summarized into three major activities:

1. Analysis of the needs and requirements engineering

2. Synthesis of concepts

3. Evaluation, Comparison and selection of concepts

Figure 2.8 represents these major concepts of the conceptual design process.

2.2 Context and models of Conceptual Design

Conceptual design is the first stage of the engineering design process. It is considered as the stage constraining the most the performance of the future product or system (Lotter, 1986). In my viewpoint, Conceptual Design is mainly composed of three sub-processes: requirements engineering, concepts generation, and evaluation and comparison of concepts. Figure 2.8 presents these sub-processes and their interactions. It is important to note that this representation does not account for the precedence of processes as, in fact, these sub-processes can be operated concurrently and the conceptual design process itself can be regarded as a long term iterative process.

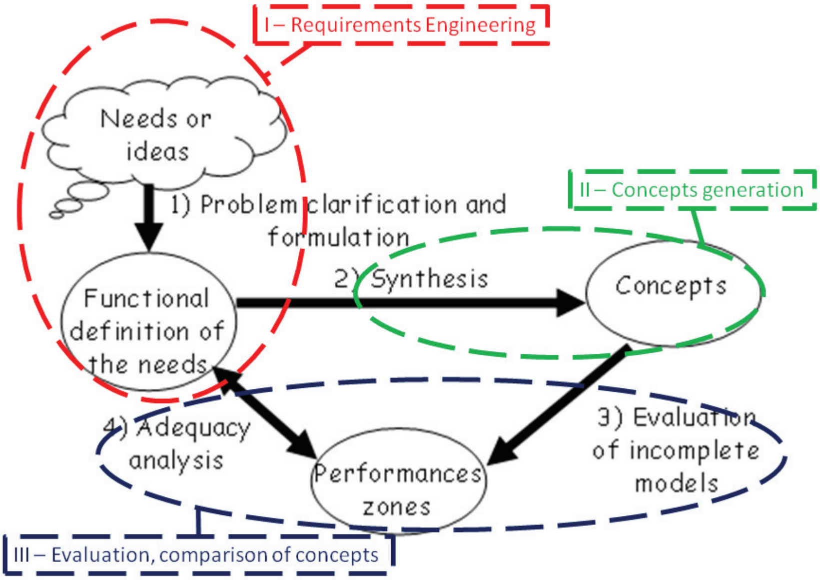


Figure 2.8. The conceptual design process adapted from (Yannou, 2000) (Coatanéa, 2005)

The whole design process focuses on defining customers' needs and requires functionality in early stages of the development cycle, documenting requirements followed by design synthesis and system validation, considering the complete problem of operations, performance, testing, manufacturing, cost, schedule, training, support and disposal (INCOSE-SEH-WG, 2004). This definition points out the importance of early design and integrating activity very clearly setting high demands for modelling concepts and tools. Complex system design embraces several domains, which have their own tools and techniques, used for several years already. In this section we first describe the main elements of conceptual design through Gero's model of necessary knowledge used in this process. During this description of the early design activity, we position the synthesis of concepts process, which is the core of our research. Secondly, we present the major concepts contained in SysML and how the formalism provided by this modeling language suits the conceptual stage of engineering design. The conceptual design process is mostly divided into three sub-processes: Requirement Engineering, Synthesis of concepts of solution and Evaluation of the concepts according to requirements. The requirement engineering phase consists on refining the design problem which is originally ill-defined (Simon, 1996) into a formally defined design problem. From the analysis of customer's needs, the outputs of Requirement Engineering are the service functions that the system shall fulfill and the performance criteria required by the future system. During the Synthesis phase of conceptual design, the designing team breaks down the service functions of the system into a functional architecture of technical functions. These technical functions are then to be fulfilled by components of the system and, once assembled, these components are supposed to provide the required service functions. The output of the Synthesis provides models of the components and their relationships inside the entire system; these models are called concepts of solution. At evaluation stage of conceptual design, the designing team assess the potential validity of each concept for fulfilment of the performance criteria initially required. This task is complex due to the usual lack of significant quantitative data, it involves multi-criteria optimization techniques. Moreover, at the end of the evaluation stage, designers shall be able to compare the concepts of solution established during Synthesis in order to go on with the detailed phase of engineering design with the solution that they consider as most relevant to be achieved. This description of conceptual design might seem rather simple, nevertheless this process is most of the time iterative if not recursive, and thus time consuming. Additionally, these three sub-processes can be treated in a concurrent manner by parts of the designing team, therefore involving proficient communication skills within the team. Nevertheless, these three sub-processes remain independent of their sequence of application and belong to the common practice in conceptual design.

This preliminary description of conceptual design highlighted some of the major concepts used during that stage. Gero has developed an interesting model of the knowledge required during conceptual design (Gero, 1990). This model is developed around the triplet of knowledge concepts, Function, Behavior and Structure, thus it is called FBS model. The interest of the FBS model is that it represents this triplet of knowledge and their interaction in alignment with the processes of conceptual design. This is probably a reason for this model to remain current regardless to its age. Figure 2.9 presents the FBS model and the interactions between Function, Behavior and Structure.

In this model:

- ***F*** represents a set of functional variables, the necessary knowledge in order to be able to explain what the system should do,

- B_e is the expected behavior of the system, e.g. the set of variables show-

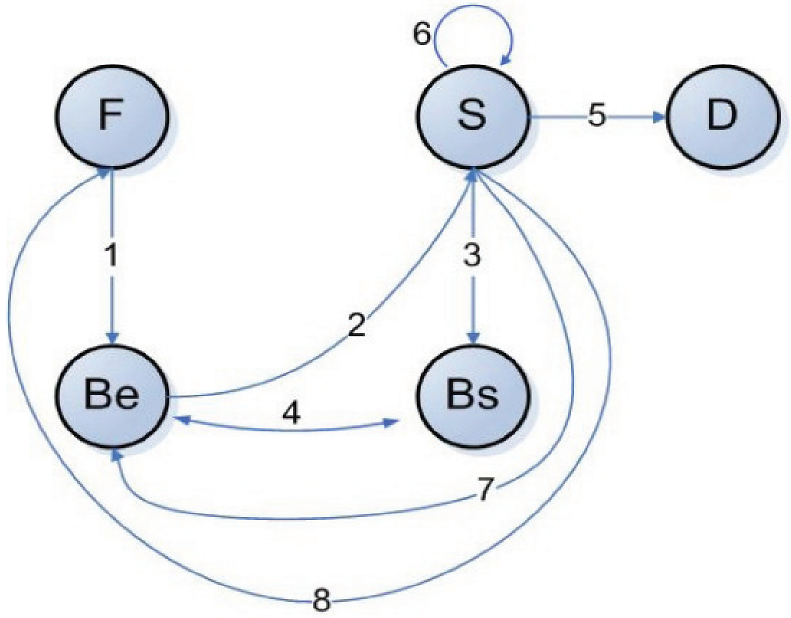


Figure 2.9. The FBS model adapted from (Gero, 1990)

ing how the system should work,

- S is the set of variable representing the physical structure of the system,

- B_s is the set of variables enabling the representation of the effective behavior of the system, e.g. its "actual" behavior

- D represents the variables contained in the documentation given for more detailed design.

The different stages of conceptual design are represented by eight fundamental processes in Gero's FBS model:

1. **Formulation:** transforms the design problem, expressed in function (*F*), into behaviour (B_e) that is expected to enable this function.

2. **Synthesis:** transforms the expected behaviour (B_e) into a solution structure (S) that is intended to exhibit this desired behaviour.

3. **Analysis:** derives the "actual" behaviour (B_s) from the synthesized structure (S).

4. **Evaluation:** compares the behaviour derived from structure (B_s) with the expected behaviour to prepare the decision if the design solution is to be accepted.

5. **Documentation:** produces the design description (D) for constructing or manufacturing the product.

6. **Reformulation type 1:** addresses changes in the design state space in terms of structure variables or ranges of values for them.

7. **Reformulation type 2:** addresses changes in the design state space in terms of behaviour variables or ranges of values for them.

8. **Reformulation type 3:** addresses changes in the design state space in terms of function variables or ranges of values for them.

This section has presented the synthesis activity in the context of conceptual design. In the following section we present the graphical language developed for the engineering of systems, SysML. We stress the similarities and differences encountered with Gero's model during the description of the language.

2.3 Requirements Engineering: the initial stage of Conceptual Design

The discipline of Requirements Engineering has emerged from Systems and Software Engineering (Hull et al., 2005). This term is used to reconcile the different terminologies from the literature. Requirements engineering contains three major disciplines as shown in Figure 2.10, namely: requirements definition, requirements management and acceptance testing. Regarding Conceptual Design, one focuses mostly on the requirements definition part. However, issues such as change management, traceability and status tracking, which belong to requirement management, remain of great importance in order to keep track of the history of the product different versions.

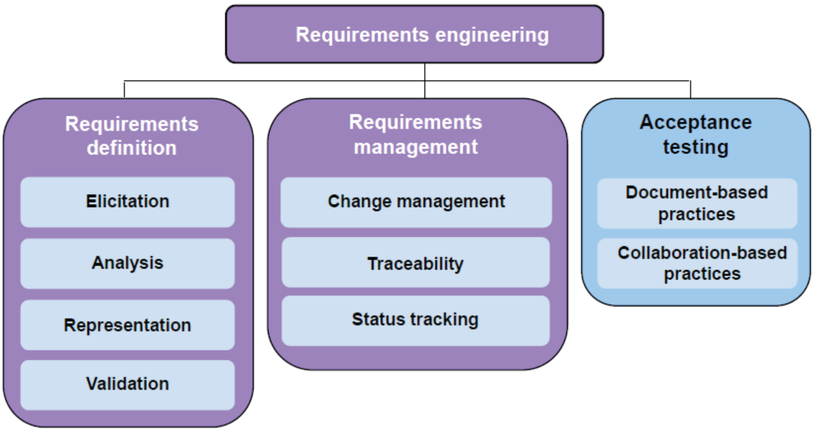


Figure 2.10. Classification of the activities involved in Requirements Engineering

2.3.1 Requirements definition

Requirements definition is usually decomposed into: elicitation, analysis, representation (i.e. modeling) and validation. Requirement elicitation consists in gathering and expressing the different needs and potential problems from as many stakeholders as possible, e.g. customers, users, authorities, developers. The term elicitation is used because the intent during that phase is to really extract as much information as possible on the current status of the design problem and needs. This elicitation is often expressed in natural language and sketches as it deals with stakeholders that are not necessarily experts in engineering design. Figure 2.11 presents the potential results from requirement elicitation process with requirements being classified according to business, user or technical levels.

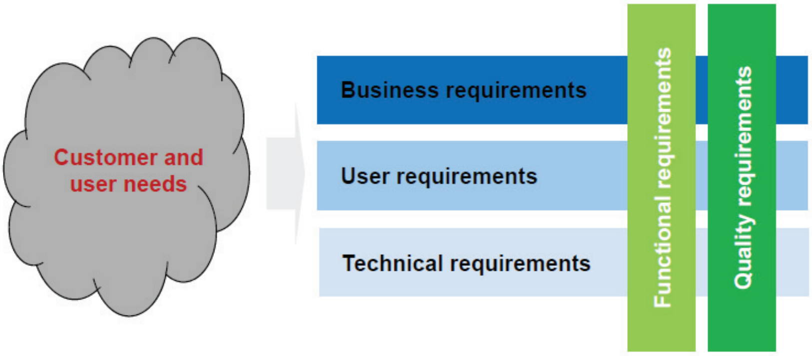


Figure 2.11. Classification of requirements according to different levels

The term requirement engineering is often used in product development and in engineering design in various industries from software engineering to food industry. Nevertheless despite the common use of the term in product development, it is useful to recall the basic objectives of requirement engineering also named sometimes requirements analysis. Requirements engineering in systems engineering, integrate tasks such as determining the needs or conditions to meet for a totally new product or a new version of a product. The task is taking into account the conflicting requirements of the different group of people involved in the project, such as the users. Traditionally the requirement engineering activity includes three fundamental actions. First the people in charge of the requirement engineering need to elicit the requirements which means that they need to communicate with the customers and future users to determine what their requirements are. They need to gather the requirements. Second they need to analyze the requirements in order to check if the stated requirements are unclear, incomplete, ambiguous, or contradictory, and they need to resolve these issues. Third, they need to record the requirements. The requirements can be recorded in the form of natural-language or more formally in the form of use cases for example in UML or SysML. In design the requirements can take the form of user stories also named brief. In process engineering it can take the form of specifications.

Nevertheless, in system engineering recording the requirements is not the end of the process indeed in the traditional vision of the V cycle, the development process is recursive and dynamic. Consequently, the requirement engineering is embedded in the system engineering and requires also considering some other phases such as the model of the system and the verification process associated with the system engineering approach. When the system is modeled we need to validate the requirements by doing a recursive analysis of the recorded requirements and last but not least we need also to manage the requirements in order to take into account the changes that can occur during the development process.

From this short summary we can derive some interesting aspects that will drive the development of this chapter. First, both the requirement elicitations as well as the requirement management during the development process require understanding properly the environment of the system that has to be developed. The definition of environment retained in this work is large and encompass different viewpoints and perspectives. For example this definition includes the consideration of the different stakeholders of the project as well as other aspects of the environment such as the climatic constraints, the usage of the system, the economic considerations and many other points of view. The extended environment of the system and the system are subject to potential dynamic changes during the project but also during the different phases of the life cycle. The perspective retains involve not only a system perspective but also a super-system perspective encompassing the environment. One practical manner to consider this super system perspective is to use modeling tools such as system dynamics or causal loops diagrams. This is the perspective that is first developed in this work.

It remains important to define the goals that can be achieved using a super-system perspective implemented via a system dynamics approach. The existing literature on System Engineering is not emphasizing very much on the nature of the causal relations existing between the requirements. Another aspect potentially resulting from the analysis of the different types of causal relations existing between requirements is the possibility to explore contradictions between requirements. The idea in this work is not to proceed in a classical engineering or scientific manner by selecting a strategy based on trade-off between conflicting requirements. Instead, the goal is to use a TRIZ inspired philosophy where the goal is to overcome the contradiction. This approach is potentially very fruitful because it might be a powerful manner to orient the design process in most potential innovative paths.

2.3.2 Environment representation with System Dynamics

Engineering design traditionally consists in modeling the interaction between the product or system to be designed and its environment. The general environment of the system can be seen at different levels such as: the entire lyfecycle of the product or each phase of the lifecycle of the product. Considering the use phase of the lifecycle of the product, the main concepts used to represent the future system within its environment are such as goals, functions or constraints. This vision is commonly used in requirement analysis to organize use cases or when a description in form of natural language is done. This approach can also be used when developing a brief but the concept of function is not as such a very operative concept for designers or marketers. They will most probably prefer concepts such as value, affordance or usage. Affordance or usage might be difficult to represent using system dynamics but a manner to measure the performance of these concepts is generally to be able to generate value for the different stakeholders. The concepts of value, energy, material or information can be easily represented by flows and stocks.

The term stock is a concept specific to system dynamics which represents containers of energy, material or information. In fact, System Dynamics has not been applied to Engineering Design. Nevertheless, it is noticeable that there is growing activity in International Council on Systems Engineering (INCOSE) to try to combine Systems Engineering body of knowledge with Systems Thinking and Systems Dynamics. An entire working group has been started in order to combine these concepts into System Science (Dee et al., 2011).

Stocks are characterizing the states of the systems. They also provide the systems with inertia and memory. The stocks accumulate past events and without changes in flows, stocks, the past accumulation in stocks persist. The stock is also the source of delays in systems. Stocks decouple rate of flows and create disequilibrium dynamics because flows are generally governed by different decision processes. For all these reasons a representation using system dynamics at requirement engineering level has the potentiality to provide insight compared with traditional approaches via brief, SysML, UML or natural languages. Another advantage is the possibility to easily communicate between different stakeholders using this representation because of the rather intuitive analogy of this language with hydraulics.

Finally, there is also the possibility to rapidly generate simulation of the requirement models using this approach and consequently options and analysis can be conjointly done by engineers and the other stakeholders.

Understanding the environment and the context of an engineering project is a task which is usually considered as to be the central task of engineers. Instead much more energy is traditionally put on the real engineering developments. There is nevertheless in the engineering tradition a series of tools that can be used to represent the environment of a service or product. Nevertheless the definition of the environment of a product can be extensive or narrow. The purpose of this chapter is to demonstrate that the environment of a product or service can be seen as an efficient tool to detect added value that can be provided to society, companies or customers. Extending the perimeter of this search might be an efficient manner to create value, to understand the user experiment and to understand the feasibility of the technical solutions.

System dynamics is a field of science that studies the behaviour of complex systems. By building models of dynamic systems we can easily study and reflect on their behaviour and the inside relations in dynamic systems (Sterman, 2000). Dynamic system models are an excellent method for breaking down complex systems and studying them in more detail. By creating visual models of dynamic systems we can easily see the different interrelationships that exist between the different parameters and variables of a system. The key aims of modelling dynamic system are:

- to show overall functional structure and flows,
- to identify functions, flows, and data stores,
- to identify interfaces between functions,
- to provide a framework for deriving system requirements.

A dynamic system contains stocks, flows and converters, and it is of these and their correlations that processes and dynamic systems are built. Stocks represent a level, a condition, or attribute of a variable at a given time. Stocks can also be used to represent buffers or delays. Flows represent activities or flows in motion. Converters represent constants, conversion tables, conditions or restrictions affecting the behaviour of the system (Gharajedaghi 1999). In product or project development processes stocks are usually represented by product variables. Flows represent the activity of defining the product specifications. Converters are those input values, variables, restrictions, standards and regulations that define the allowable interrelationships and have an influence on the flow. Stocks, flows and converters are used in models showing system behaviour and throughputs as shown in Figure 2.12. It is easier to grasp interdependencies between variables by using visual models such as those presented in Figures 2.13, 2.14, 2.15 and 2.16.

One key feature when modelling dynamic systems and showing the func-

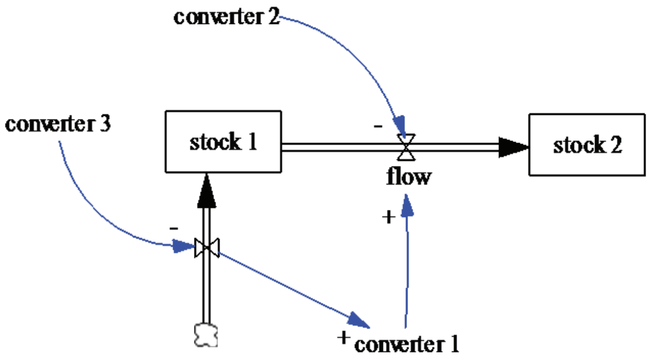


Figure 2.12. Basic concepts of Systems Dynamics: Stock, flow and converter

tional structure of the system is the discovery and representation of feedback processes and loops. There are two different types of loop: positive and negative. Figure 2.13 shows an example of a positive feedback loop. A positive feedback loop is where the system is self-reinforcing. Figure 2.13 is an example of a positive or self-reinforcing loop using the example of eggs and chickens. The more chickens there are, the more eggs there will be; the more eggs there are the more chickens there will be.

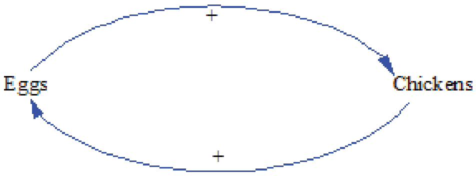


Figure 2.13. Positive feedback or self-reinforcing loop

If the chicken-egg loop were the only one in the system, the chicken population would grow exponentially. In real life no natural growth can continue forever and this is corrected by negative feedback. Figure 2.14 shows an example of a negative feedback loop, or self-correcting loop. The more chickens there are, the more road crossing they will attempt. More road crossing will lead to fewer chickens. An increase in chickens will lead to more road crossings, which will then lead to a decrease in the chicken population.

These two feedback loops shown above for Egg-Chickens and Chickens-Road Crossing can be combined into a system with multiple loops as in Figure 2.15. Depending on the type of system there can be numerous loops, both positive and negative. The dynamics of all the systems arise from the interactions between the different loops (Sterman, 2000).

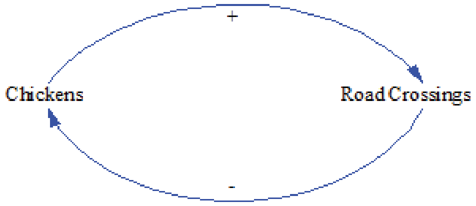


Figure 2.14. Negative feedback or self-correcting loop

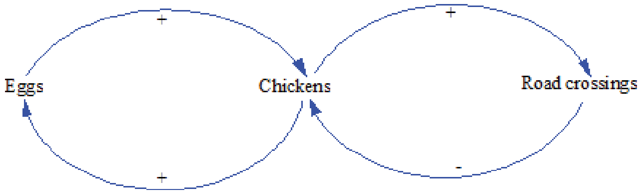


Figure 2.15. System with multiple dynamic loops

The strength of the relationship between two variables can be described by giving values to the interdependencies. If the dynamic model contains performance variables, dependency values can either be negative or positive. Depending on whether the relation is positive or negative, conflicts are produced. These emerging contradictions play an interesting part in the models, particularly in the product development and specification processes. System dynamic models are an excellent method for locating contradictions and performing a further study of their influences. It should be kept in mind that it is not always possible to assign positive or negative values to the relationships. Some systems may not contain performance values. The characteristics of the variables in these systems may only define the prevailing constraints and describe which attributes are allowed for the dependent variables.

Dynamic models can also be used to study indirect relationships, from one variable to another via a third variable. One requirement derived from one variable can become the input requirement for other variables (Hull et al., 2005). Figure 2.16 can be studied as an example of a dynamic system with performance values and derived requirements. The cruising range of a car is dependent on many factors such as, for instance, engine fuel consumption and fuel tank size. Fuel consumption is dependent on the total weight of the car as well as the engine size. Increasing the engine size causes a direct increase in the fuel consumption. An increase in engine size also affects the fuel consumption indirectly, as it increases the total weight, hence fuel consumption also increases. If we were to increase the fuel tank size this would cause a direct increase in cruising range, but an increase in tank size also increases the total weight, which has a negative impact on the cruising range. As the example in Figure 2.16 shows, visual dynamic models are a great tool for presenting performance contradictions, conflicts and the possibility to attain targets.

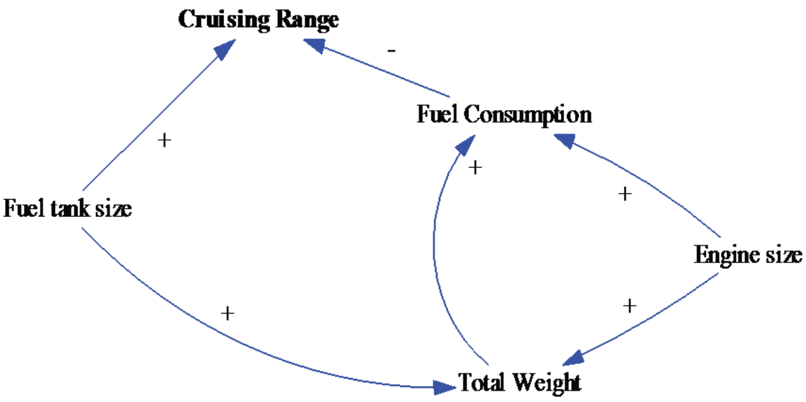


Figure 2.16. Dynamic system representing elements affecting the cruising range of a vehicle

In nature and engineering there are numerous situations where conflicts arise, e.g. electrical applications and water create a conflict, as their combination represents a hazard. Similarly, conflicts also arise when analysing requirements and establishing specifications. Coatanéa has shown that the analysis of causal loops enables finding contradictions or conflicts between requirement variables by applying analysis from graph theory to causal graphs. Finding these contradictions allows prioritization between design variables. Additionally, it allows making decision about potential trade-offs to make in design and enables targeting them at the very beginning of the design activity.

This section attempts to show that systems dynamics enables the analysis of causal loops in order to find contradictions within requirements. Such finding is very beneficial at this stage of engineering design as it would enable correcting such contradiction dynamically before going further into designing the artifact.

2.3.3 Elicitation and Representation of the initial design conditions

Intuitively, design is a human activity that aims to change an existing environment to a desired one by creating a new artefact into the existing environment. Environment-Based Design (EBD) is such a design theory that studies and supports this environment change process. The underlying principles behind the EBD are that design comes from the environment, serves for the environment, and goes back to the environment (Zeng, 2004). This section introduces the part of the EBD theory that is relevant to the problems to be addressed by this paper. The first subsection will be focused on the dynamics, in the form of design governing equation, underlying the design process. Based on the governing equation, an observation about design activities is explained in the second subsection. The third section will introduce the EBD process whereas the last subsection will introduce a question asking approach for implementing the EBD process.

Mathematically, the EBD process can be represented by structure operation, denoted by $\oplus$. Structure operation can be defined as the union ($\cup$) of an object O and the interaction ($\otimes$) of the object with itself.

$$\oplus O = O \cup (O \otimes O) \tag{2.1}$$

where $\oplus O$ is the structure of the object O.

Everything in the universe can be seen as an object. Interactions between objects are also objects. Examples of interaction include force, movement, and system input and output. Structure operation provides a means to represent a hierarchical system with a single mathematical expression. The application of structure operation can be found in the representation of conceptual graphs from Sowa (Sowa, 2000) and linguistic information in design. In the beginning of the design process, there was only environment. As the design progresses, any previously generated design concept can be indeed seen as an environment component for the succeeding design. As a result, a new state of design can be defined as the structure of the old environment (E_i) and the newly generated design concept (S_i), which is a partial design solution.

$$\oplus E_{i+1} = \oplus(E_{i \cup S_i}) \tag{2.2}$$

It has been shown that the environment structure, which is $\oplus E$, includes the description of the design solution at design stage i and the design requirements for the design stage $i+1$. Therefore, the recursive evolution process of design requirements and design solution can be mathematically formulated in the following design governing equation (Zeng, 2008):

$$\oplus E_{i+1} = K_i^s(K_i^e(\oplus E_i)) \tag{2.3}$$

where K_i^s and K_i^e are synthesis and evaluation operators, respectively.

The design governing equation makes design problem solving as a search for fixed points under the design function $K_i^s(K_i^e(\bullet))$. Different design methodologies indeed solve the design governing equation 2.3 under different assumptions. Considering this initial framework the impact of the initial conditions can be analysed. This is the role of the following subsection.

Impact of initial conditions on design

According to the recursive logic of design (Zeng, 2008), at most stages of conceptual design, the evaluation operator will be determined only after a preliminary design solution is generated, which will in turn trigger new synthesis operators. As a result, design is a non-linear process where a small change in the initial design problem may give rise to significant differences in the final design solutions, among which creative design solutions may exist.

Based on the non-linear characteristics underlying the design governing equation 2.3, it was observed that the initial conditions for design are in constant change during the conceptual design process. Every new design solution will lead to the redefinition of the original design problem, which presents new initial conditions for design. Zeng thus formulated mathematically three paths that may change the initial conditions in the design process, which are:

1. Formulating the design problem differently;

2. Extending synthesis knowledge; and

3. Changing the sequence of environment decomposition.

2.4 The synthesis process

The synthesis process is the stage that enables to go from "what the system should do?" to "How it is going to do it". It is the way from a conceptual viewpoint on the problem to a more tangible or concrete description of a potential solution to this problem. Methods have been developed to provide support for this transformation. TRIZ is one of them. Thie idea of TRIZ is to transform a specific problem into a generic problem for which we know generic solutions or solutions used in different disciplines. These generic solutions are then transformed into specific solutions answering the specific problem.

2.4.1 TRIZ

Synthesis in Conceptual design is related to creativity. Probably the most relevant work in terms of creativity and inventive design was proposed by Altshuller with TRIZ (Altshuller, 1984). The two main ideas of TRIZ are the following:

- many problems faced by engineers contain elements that have already been solved in a different context and industrial field

- patterns of technological change can be predicted and applied to any situation to determine the successful next steps in technological change

TRIZ is the theory of innovation applied in a systematic manner. The main idea behind this theory is that inventions could be organized and generalized by function. Therefore, generalizing the design problem in a functional manner could enable finding the essential physical principles which will solve the problem. TRIZ provides principles and rules to find general solutions to a specific problem. Nevertheless, this process of specialization is strongly context dependent and it is not possible to automate it. Therefore, scientists have developed many other conceptual design models adapted to a specific domain of product development, e.g. electronics, mechanics.

One approach used in TRIZ is to detect contradictions between different design objectives as a mean to improve the formulation of the design problem in order to make no compromise between these contradictions.

Finding contradiction can be a starting point towards more efficient synthesis process.

2.4.2 C-K theory

C-K theory defines design as (Hatchuel and Weil, 2003): “assuming a space of concepts C and a space of knowledge K, we define Design as the process by which a concept generates other concepts or is transformed into knowledge, i.e. propositions in K”.

In this theory, K, i.e. a piece of knowledge, is considered as a logical proposition having a status for the designer or for the customer (e.g. True or False in first order logic). On the contrary, a concept is defined as a notion or proposition without any logical status: “It cannot be said from a concept whether the concept by itself is right or wrong” (Hatchuel and Weil, 2008).

According to the authors, the design reasoning can be theorized as the co-evolution of these two spaces C and K. They call "capacity of expansion" the ability of the design process to generate novelty via a reasoning which begins by a disjunction K->C which is creating a concept and ends by a conjunction C->K transforming a concept into knowledge.

Authors defined the operators (C->C, C->K, K->C, K->K) which organize the co-evolution of the C and K spaces in the following manner (Masson and Hatchuel, 2006):

K->C: This operator adds or subtracts to concepts in C some properties coming from K. It creates “disjunctions” when it transforms elements from K into a concept. This also corresponds to what is usually called the “generation of alternatives”. Yet, concepts are not alternatives but potential “seeds” for alternatives. This operator expands the space C with elements coming from K: concepts cannot be imagined without knowledge. They call this K-relativity of a design process (Hatchuel and Weil, 2008).

C->K: This operator seeks for properties in K that could be added or subtracted to reach propositions with a logical status; it creates conjunctions which could be accepted as “finished designs” – when true. Practically, it corresponds to validation tools or methods in classical design: consulting an expert, doing a test, an experimental plan, a prototype, a mock-up are common examples of CK operators. A design solution is precisely what Hatchuel and Weil call a “conjunction”. They have reached a concept which is characterized by a sufficient number of propositions that can be established as true or false in K (Hatchuel and Weil, 2003).

K->K: This operator allows a knowledge space to have a self- expansion. This operator corresponds to an expansion of the knowledge space obtained by deduction and/or experiment. This operator is not fundamental for the design process to occur. This operator and the following one are corresponding to the exploration of the design space.

C->C: Finally, the operator C->C explains the expansion of the concepts space. The expansion of C (the addition of a new concept) can be done by removing a property to a concept, it is then an inclusion. Adding a property constitutes otherwise a partition. The partition is restrictive if the property already belongs to the concept. It is expansive in the case where a new property is added to the concept.

These mechanisms make the C space a tree structure (partitions correspond to the creation of new "branches", inclusions to their pruning). "We can only create new concepts (new sets) by adding or subtracting new properties to the initial concept.

As a summary, for authors of the CK theory, the mechanism of expansive partition is the elementary motor of design. This is contrary with the appampbroaches of problem solving. The mechanism of expansive partitions requires therefore two initial conditions:

- The set to partition is not completely specified. This set is expandable.

- The partition is activated using an external knowledge, outside of the CK-space.

The first interest in this theory is that Hatchuel separated two aspects of design thinking: the concept or idea and the knowledge, validated by humankind. Additionally, Hatchuel defined operators between these two aspects which can be considered as synthesis operators. To some extent, these operators can be related to mathematical reasoning and logic or, more generally, related to a scientific method.

Nevertheless, the theory does not provide a clearly defined process of the synthesis activity. The theory does not Also, it does not integrate the concept of function. Thus, as the cases are always expressed in term of structures or products, it is hard to perceive the limits between K and C. C = concept (idea: is not yet a part of the general knowledge) K = knowledge (knowledge acquired: validated by human community)

2.4.3 Formal synthesis in Engineering Design

Synthesis process happen not only in conceptual design but at different phases such as embodiment design for synthesis of forms or in detailed design for the integration of subsystems for example (Antonsson and Cagan, 2001) (Chakrabarti, 2002).

2.4.4 Conclusion on synthesis process

As seen in the previous descriptions of synthesis, there are multiple kinds of synthesis processes appearing at different places during the design process. What to consider with regard to our subject? In this thesis, we focus mainly on the functional analysis because it appears to us as the core activity of the synthesis in conceptual design (Chakrabarti et al., 2011). TRIZ and the derived tools are also considered for conceptual design synthesis.

In our focus of finding description of the knowledge needed for designers to achieve a synthesis of concepts of solution, it appears that there was consequent work achieved in the creation of taxonomies of functions, energies, flows. Probably the work summarizing it best and reconciling these taxonomies together is from Hirtz and Stone (Hirtz et al., 2002). Additionally, the functional repository (Stone et al., 2010) of products supported nowadays by Oregon State University and initiated by Pr. Stone permitted relevant work in direction of automating the synthesis process (Kurtoglu and Campbell, 2009).

The idea of this thesis is to organise the taxonomies under a common knowledge representation expressing formally the manner each of them integrate with another. Therefore, section 2.5 presents definitions of knowledge representation issued from Artificial Intelligence.

2.5 Knowledge Representation (KR): Definition and KRs for Conceptual Design

This section aims at defining what is Knowledge representation and the practical ways to implement such kind of representation by using Knowledge Engineering tools. First we present a summary of a definition of Knowledge Representation which is widely accepted in the field of Artificial Intelligence. Second, we provide a state of the art of the different semantic representations that are used in order to formalize knowledge and embed it into computer systems.

2.5.1 Knowledge Representation

Randall Davis provided a definition of a Knowledge Representation (Davis et al., 1993) as the following:

1. **A KR is a surrogate:** Any intelligent entity that wants to reason about its world encounters an important, inescapable fact: Reasoning is a process that goes on internally, but most things it wants to reason about exist only externally. A program (or person) engaged in planning the assembly of a bicycle, for example, might have to reason about entities such as wheels, chains, sprockets, and handle bars, but such things exist only in the external world.
   This unavoidable dichotomy is a fundamental rationale and role for a representation: It functions as a surrogate inside the reasoner, a stand-in for the things that exist in the world.

2. **A KR is a set of ontological commitments:** If, as we argue, all representations are imperfect approximations to reality, each approximation attending to some things and ignoring others, then in selecting any representation, we are in the very same act unavoidably making a set of decisions about how and what to see in the world. That is, selecting a representation means making a set of ontological commitments. The commitments are, in effect, a strong pair of glasses that determine what we can see, bringing some part of the world into sharp focus at the expense of blurring other parts.

3. **A KR is a fragmentary theory of intelligent reasoning:** The third role for a representation is as a fragmentary theory of intelligent reasoning. This role comes about because the initial conception of a representation is typically motivated by some insight indicating how people reason intelligently or by some belief about what it means to reason intelligently at all.
   The theory is fragmentary in two distinct senses: (1) the representation typically incorporates only part of the insight or belief that motivated it and (2) this insight or belief is, in turn, only a part of

the complex and multifaceted phenomenon of intelligent reasoning.

A representation's theory of intelligent reasoning is often implicit but can be made more evident by examining its three components: (1) the representation's fundamental conception of intelligent inference, (2) the set of inferences that the representation sanctions and (3) the set of inferences that it recommends.

Where the sanctioned inferences indicate what can be inferred at all, the recommended inferences are concerned with what should be inferred. (Guidance is needed because the set of sanctioned inferences is typically far too large to be used indiscriminately.) Where the ontology we examined earlier tells us how to see, the recommended inferences suggest how to reason.

4. **A KR is a medium for efficient computation:** From a purely mechanistic view, reasoning in machines (and, perhaps, in people) is a computational process. Simply put, to use a representation, we must compute with it. As a result, questions about computational efficiency are inevitably central to the notion of representation.

   This fact has long been recognized, at least implicitly, by representation designers: Along with their specification of a set of recommended inferences, representations typically offer a set of ideas about how to organize information in ways that facilitate making these inferences.

5. **A KR is a medium of human expression:** Finally, knowledge representations are also the means by which we express things about the world, the medium of expression and communication in which we tell the machine (and perhaps one another) about the world. This role for representations is inevitable as long as we need to tell the machine (or other people) about the world and as long as we do so by creating and communicating representations. Thus, the fifth role for knowledge representations is as a medium of expression and communication for our use.

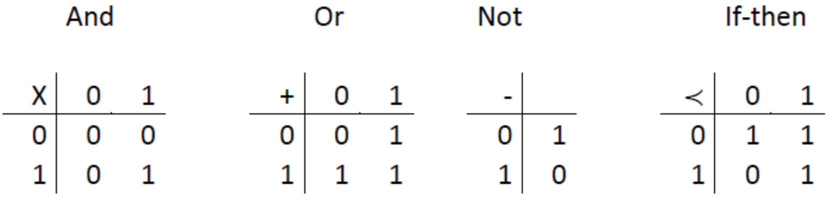
Mathematical Logic	Psychology	Biology	Statistics	Economics
Aristotle				
Descartes				
Boole	James		Laplace	Bentham Pareto
Frege Peano			Bernoullii	Friedman
	Hebb	Lashley	Bayes	
Goedel	Bruner	Rosenblatt		
Post	Miller	Ashby	Tversky,	Von Neumann
Church	Newell,	Lettvin	Kahneman	Simon
Turing	Simon	McCulloch, Pitts		Raiffa
Davis		Heubel, Weisel		
Putnam				
Robinson				
Logic PROLOG	SOAR KBS, Frames	Connectionism	Causal Networks	Rational Agents

Figure 2.17. Views of Intelligent Reasoning and Their Intellectual Origins (Davis et al., 1993)

2.5.2 Logic

Aristotle developed logic as a precise method for reasoning about knowledge. By inventing syllogism he presented three-part pattern of logical deduction containing major premise and minor premise which lead to the conclusion. To go further he presented systematic analyses with formal rules of inference to convert one pattern into another while preserving truth. Scholastic logic was developed in medieval time assigning A for universal affirmative: all a is b, I for particular affirmative: some a is b, E for universal negative: no a is b and O for particular negative: some a is not b. Besides the linear notations of logic, scientists involved in the field of Artificial Intelligence developed graphic notations called semantic networks. Already in the thirteenth century Ramon Lull invented the first mechanical device for automated reasoning with ten types of questions that may be asked: Whether? What? From what? Why? How much? What kind? When? Where? How and With what? (Sowa, 2000) Mathematical logic developed much by Leibniz and Newton who independently from each other invented differential calculus. Leibniz developed mechanical calculator with which it was possible to do multiplication and division. Furthermore in logic Leibniz used mathematics to formalize the patterns of syllogistic reasoning and formed the first system of binary arithmetic. In philosophy he introduced concepts of modality, identity and continuity. As Leibniz saw that his calculator could be used for mechanical reasoning he can be called as the grandfather of artificial intelligence (AI). (Sowa, 2000)

Boolean algebra was developed in “Investigation into the Laws of Thought” written by George Boole in 1854. Whereas Leibniz used numbers to represent categories, Boole used numbers to represent truth values, 0 for false and 1 for true. Besides multiplication for conjunction, Boole used addition for OR-disjunction and minus for the NOT-negation. Charles Sanders Pierce made extensions and modifications to Boolean algebra. Pierce’s truth tables (Figure 2.18) are well known:



And

X	0	1
0	0	0
1	0	1

Or

+	0	1
0	0	1
1	1	1

Not

-	
0	1
1	0

If-then

≺	0	1
0	1	1
1	0	1

Figure 2.18. Pierce’s truth tables for Boolean algebra

To represent knowledge in logic, Leibniz tried to present a universal language using mathematical principles so that it is precise enough and settle any dispute among persons. Not only modern logic is capable to represent in mathematical principles factual information that can be stated precisely in any language, natural or artificial. (Sowa, 2000)

Notation in propositional logic can be very simple. A sentence “Every trailer truck has 18 wheels” can be represented simply by a letter p. A set back of that representation is the loss of detail. For that reason it is not suitable to describe internal structure of propositions but implications between them. If the internal structure has to be described predicate logic can be used:

$$\begin{aligned}&(\forall x)((truck(x) \wedge (\exists y)(trailer(y) \wedge part(x,y))\\&\supset (\exists s)(set(s) \wedge count(s,18))\\&\wedge (\forall w)(member(w,s) \supset (wheel(w) \wedge part(x,w))))\end{aligned} \tag{2.4}$$

This may be read: For every x, if x is a truck and there exists a y where y is a trailer and x has y as a part, then there exists an s where s is set and the count of s is 18 and for every w is a member of s, then w is a wheel and x has w as a part. (Sowa, 2000)

Some observations from above can be made: It is easier to read natural language than the expression in predicate logic. Logic itself has only half a dozen basic symbols which are quite simple. The level of detail depends on the predicates like x, y, s and w which strictly do not belong to the logic but represent the ontology of the relevant things that exist in that particular domain. (Sowa, 2000)

Classical First Order Logic (FOL) developers, Frege and Pierce, were able to converge very divergent assumptions and different notations of logic into semantically identical systems and derive exactly the same theorems. Today FOL is often studied in six dimensions: syntax, subsets, proof theory, model theory, ontology and metalanguage(Sowa, 2000).

2.5.3 Conceptual Graphs

A conceptual graph (CG) is a graph representation for logic and corresponding ontology. Several versions of CGs have been designed and implemented. The simplest are the core CGs which are original Peirce's existential graphs but more common are the extended CGs. Sowa developed in 1976 a version of conceptual graphs to map natural language questions and assertions to a relational database. Figure 2.19 shows a CG for the sentence John is going to Boston by bus. The rectangles are called concepts, and the circles are called conceptual relations. An arc pointing toward a circle marks the first argument of the relation, and an arc pointing away from a circle marks the last argument. If a relation has only one argument, the arrowhead is omitted. If a relation has more than two arguments, the arrowheads are replaced by integers 1,...,n. (Sowa, 1976)

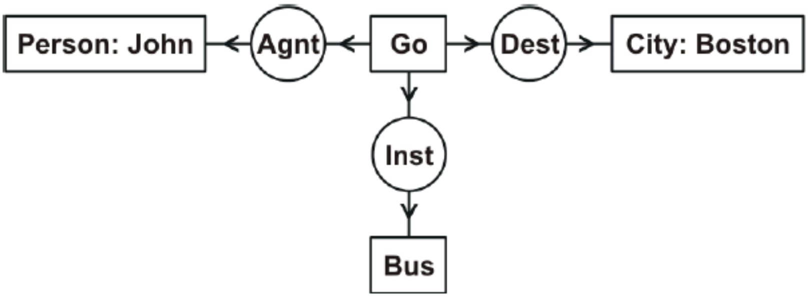


Figure 2.19. CG stands for John is going to Boston by bus

CG can be translated to the following formula:

$$\begin{aligned}&(\exists x)(\exists y)(Go(x) \wedge Person(John) \wedge City(Boston) \wedge Bus(y)\\&\wedge Agnt(x, John) \wedge Dest(x, Boston) \wedge Inst(x, y)\end{aligned} \tag{2.5}$$

Each of the four concepts has a type label. It represents the type of entity the concept refers to: Person, Go, City or Bus. Two of the concepts have names: John or Boston. Each of the three conceptual relations has a type label that represents the type of relation: agent (Agnt), destination (Dest) or instrument (Inst). [10]

2.5.4 Ontologies

The fundamental question of ontology can be expressed with a question: What is there? In logic the existential quantifier $\exists$ is a notation that something exists but in logic there is no vocabulary to describe a thing that exists. For that end ontology is needed. Ontology studies the existence of entities weather they are concrete or abstract. So the notation $\exists x$ can be expressed "there is something x such that" which combines logic and ontology. Ontology provides the predicates of predicate logic and calculus. The predicates in ontology are divided in two classes. Domain dependent predicates are domain specific to a particular application. In equation 2.4 domain specific examples are $truck(x)$, $trailer(y)$ and $wheel(w)$. Domain independent predicates like $part(x, y)$, $set(s)$, $count(s, n)$, and $member(x, s)$ can be used in many different applications. Predicates can be represented with help of conceptual graphs (Sowa, 2000).

The two sources of ontological categories are observation and reasoning. Observation provides information of physical world and reasoning generates the framework of abstractions called metaphysics. In database theory categories are called domains, in artificial intelligence (AI) categories are called types, in object oriented systems they are classes and in logic they are called types or sorts. Any incompleteness, distortions or restrictions in a category result inevitably in limitations to the generality of a program or database using such categories.

In the beginning of the 90's, researchers in Artificial Intelligence started to put their interest toward the notion of Ontology, commonly used in Metaphysics, in order to formalize knowledge. The main goal of this field is to "represent" what "exists". Within this context, they have defined an ontology as an artefact enabling the representation of the existing using a formal and consensual vocabulary. One of the first definitions of ontology, commonly admitted in Artificial Intelligence, was published by Gruber (Gruber, 1993) as the explicit specification of a conceptualization. This definition has then been refined by R. Studer (Studer et al., 1998) as the formal and explicit specification of a shared conceptualization:

- formal because an ontology needs to be read by machines, which excludes natural language,

- explicit because its definition explains the concepts being used and their constraints of use,

- conceptualization is the abstract model of a phenomenon from the real world with identification of the key concepts of this phenomenon,

- shared because an ontology is not an individual's property but represents a consensus accepted by a community of users.

Practically, ontologies give means to express concepts of a domain by organizing them hierarchically and by defining their semantic properties in a language of representation of knowledge helping the computer applications using this language to share a consensual view on that domain. Defining concepts and link them together with semantic relations corresponds to the first level of ontology, the conceptual model, inspired of semantic networks and moreover of conceptual graphs from Sowa (Sowa, 2000). The semantic web community is using the idea of ontology in order to express the semantic content of web pages in order to enable their exploitation by computer agents and not only by human users (Berners-Lee, 1998). In fact, Tim Berners-Lee presents the semantic web as an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation (Berners-Lee et al., 2001). In order to apply this vision practically, an architecture composed of a set of languages has been defined. This architecture is generally represented in the form of a pyramid. Each layer sits on results defined at the lower layers, thereby, each layer is progressively more specialized and more complex than the previous one. Additionally, each level is independent of the upper levels so as to be developed and operational in an autonomous manner according to the developments of upper levels. Figure 2.20 shows this pyramid of languages.

Our interest here is focused on the description of concepts used at the conceptual design of systems. Therefore we place our activity on the first level of ontology, which is the level of description of the conceptual model. We so concentrate on OWL and RDF languages of the pyramid Figure 2.20.

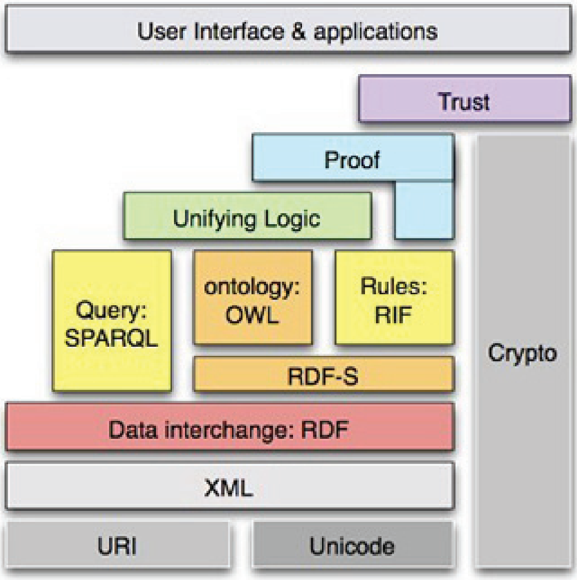


Figure 2.20. Pyramid of semantic web languages in 2006 (from Berners-Lee)

RDF enables the description of metadata in the form of triplets (resource, propriety, value) as specified in RDF schemes. Ontology Web Language, OWL, allows the description of more complex ontologies because it defines classes, attributes, relations and axioms. The biggest interest of using OWL is that it can then be combined with rule-based languages allowing reasoning about resources and inferring new knowledge about them.

2.5.5 Knowledge Representations for Conceptual Design

Researchers in engineering design have started to work on the necessary knowledge for the design of systems such as mechanical systems and multi-disciplinary systems like mechatronic systems. Kitamura, who has been involved in the early development of frameworks for conceptual design as stated earlier, developed a project called FOCUS, standing for a Functional Ontology for Categorization, Utilization and Systematization of functional knowledge (Kitamura et al., 2007). This project represents all the knowledge required for the functional representation of systems as shown Figure 2.21. This figure shows the organisation of different ontologies representing different knowledge about functions. The project contains an ontology of device and function, a functional concept ontology, descriptions of functional models of concrete devices and of the function achievement and a reference ontology of function. This work provides a very good insight on the development of functional architectures for describing the features of a device.

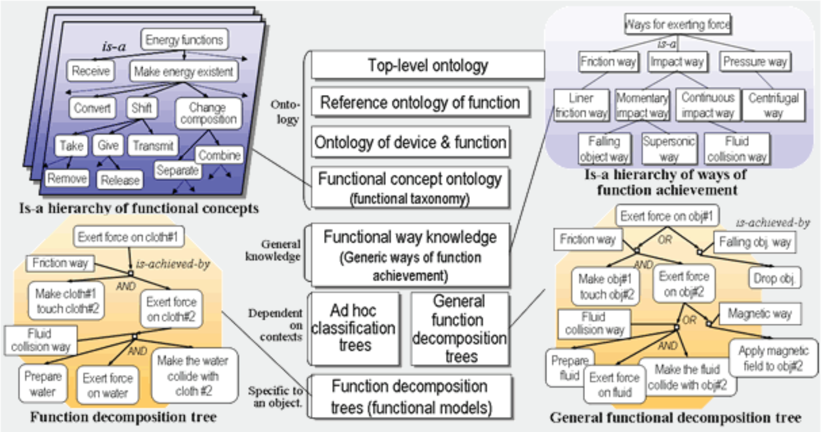


Figure 2.21. Layers of functional ontologies in the FOCUS project (Kitamura et al., 2007)

The work of Chakrabarti and his team also is a reference of automatic synthesis of mechanical systems (Liu et al., 2000). Their computer application called FuncSION helps synthesizing topological structures of mechanical objects which can then be transformed into spatial structures embodying the functions. The testing of an earlier version of FuncSION was reported in Artificial Intelligence in Design in 1996 (Chakrabarti and Tang, 1996) and it was found that the amount of generated solutions and their variety were always larger than those of the designers. The interest of our research in FuncSION is that it is also ontology based even though it is dedicated to mechanical products, further readings will show how we provide ontologies for systems engineering. In terms of systems engineering, Tudorache recently published ontologies representing conceptual design knowledge in her doctorate thesis (Tudorache, 2006). Her set of ontologies of engineering represents three ontologies dedicated to requirements, components and systems, and constraints. We will focus on her description of components and their organisation within a system as our aim is the synthesis of solutions. Figure 2.22 represents the taxonomy of components. Components may only be composite or atomic and composite components may contain other components whereas atomic components may not. Tudorache ontologies are published under Protégé server and are usable under General Public License. Figure 2.24 shows an example of the hierarchy of components and their relationships.

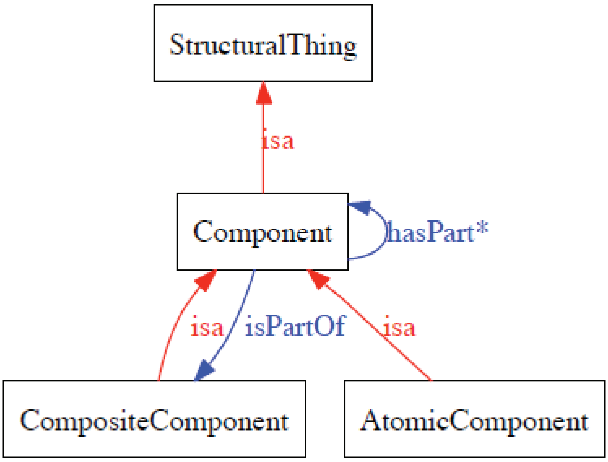


Figure 2.22. The generic description of the concept of component in Systems Engineering (Tudorache, 2006)

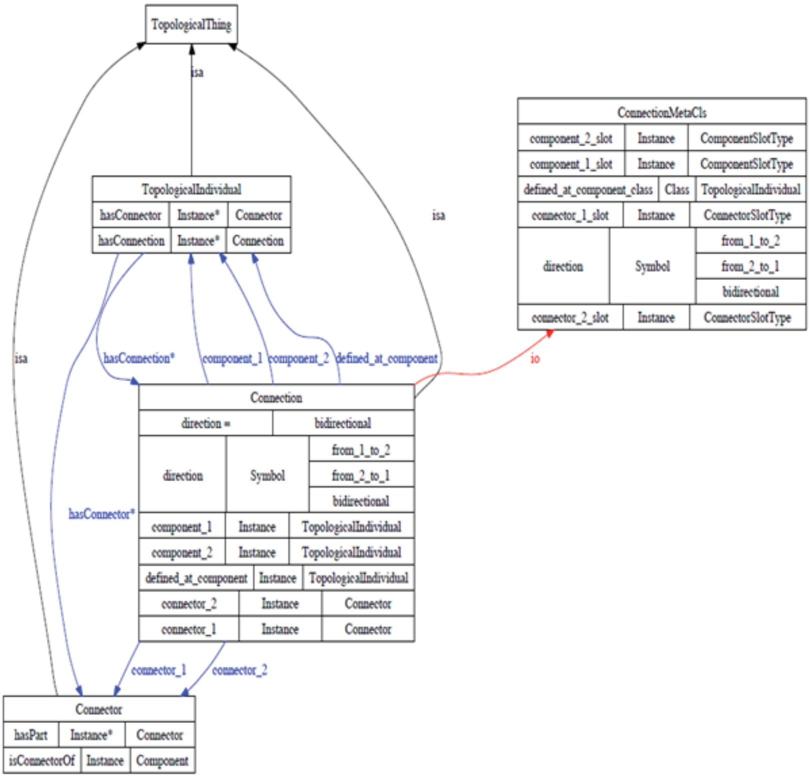


Figure 2.23. Connection and connector as topological concepts of a system (Tudorache, 2006)

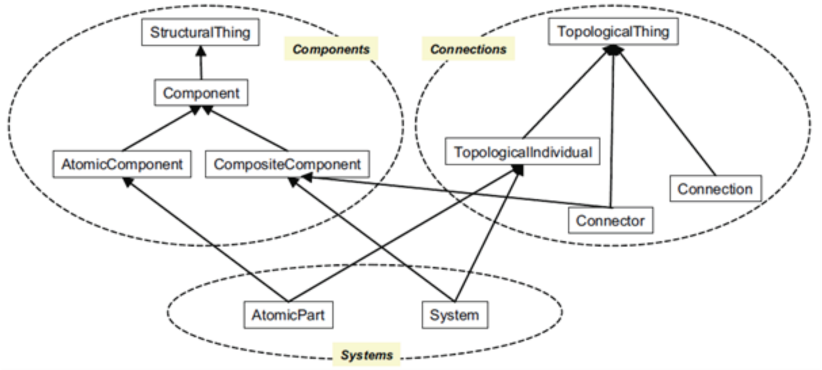


Figure 2.24. Ontology relationship between Systems, Components and Connections (Tudorache, 2006)

This approach fits systems engineering demands and provides a good basis for our research. Nevertheless, for her, a component can as well be functional or structural whereas it is important for us to reify the concept of function even if its representation has typically the structure of a component, e.g. a black-box.

2.6 Computational linguistics and Natural Language Processing

According to our problem, we consider the analysis of natural language.

Understanding a language implies understanding the grammar of this language. A grammar includes:

- morphology: the grammar of word forms

- syntax: the grammar of sentence structure

Understanding the syntax of a language implies understanding:

- semantics of the language

- its lexicon

- and pragmatics

The disambiguation in word selection and the interpretation of the context of a design problem description require analysing both the potential meanings of a word and the meanings of associated words inside a sentence.

The present section is summarizing the approach developed by Ploux et al. (Ploux, 1997) (Ploux and Ji, 2003) in order to develop a semantic map. The main characteristics of the semantic map are the following. The semantic map is able to compute the smallest unit if meaning and to represent words in a two dimensional map. Words are clustered by subgroups of meaning and a distance is computed between words. Before presenting the manner this semantic map is obtained and the use that can be made of it in this research, it is perhaps necessary to summarize some initial concepts associated with semantics. The first concept is the concept of seme. A seme is the smallest unit of meaning recognized in semantics; it refers to a single characteristic of a sememe. A sememe is a proposed unit of transmitted or intended meaning. A sememe is the semantic counterpart of an elementary particle in a substance. For example, a verb such as move can be conceived as the abstract representation (i.e. sememe) of verbs such as skate, roll, jump, and slide. In this section the authors are presenting in which manner a seme can be computed using the concept of clique introduced in graph theory. A clique is a complete sub-graph in graph theory. A complete graph is a simple graph in which every pair of distinct vertices is connected by a unique edge. The term seme has been introduced by Eric Buyssens (Buyssens, 1967) in the 1930s and developed by Bernard Pottier (Pottier, 1974) in the 1960s. New approaches have nowadays the ability to cluster contextually related words (Dagan and Itai, 1994) (Lin and Pantel, 2002). The availability of important resources in form of text corpuses and electronic dictionaries give the possibility to treat automatically or semi automatically the semantic description of lexical units. Two key research approaches exist. The first approach consists of using the electronic dictionaries available in order to use the synonym relations. The second one considers each semantic unit by analyzing the other semantic units having a syntax link with it. The present article is considering both of these approaches. First the authors consider the synonymy between words. Usually the sense of words is represented in a discrete manner by classifying in dictionaries in form of groups of synonyms. The research results of Ploux (Ploux and Ji, 2003) demonstrate that a discrete mathematic representation of synonymic relation is not sufficient to produce a semantic structure. A semantic structure should represent the different meanings of terms but also their overlapping. Ploux proposes a continuous representation (using data analysis) that enables a machine to produce for a term its semantic values. The semantic spaces are obtained automatically for each headword from a homogeneous list of synonyms. Ploux' work has lead to the creation of a semantic atlas synonym database, which is composed of several dictionaries and thesauri enriched by a symmetrical process [1]. The semantic atlas is constructed in the following manner. If a research is done on headwork, a set of cliques containing all the synonyms is calculated by the system. A clique is seen in this part as a set of terms related with each other by synonymy. The conjunction of all terms in the same clique filters and constrains the meaning given to the word. For example, for the headword good, three examples of cliques representing different perceptual values are presented below:

- Clique 11: adept, expert, good, practiced, proficient, skilful, skilled, skillful

- Clique 44: beneficial, good, healthy, salutary, wholesome

- Clique 85: dependable, good, reliable, safe, trustworthy

Correspondence factor analysis (Ploux and Victorri, 1998) is applied to the matrix composed of words in the columns and cliques in the lines to obtain coordinates for each clique (Bouroche and Saporta, 2005). The goal here is to define a neighborhood between cliques. This means that it is necessary to consider the semantic space associated with a semantic unit is a metric space. With n being the number of synonyms and p being the number of cliques, the general formula used to compute distance between cliques is:

$$d^2(c_k, c_l) = \sum_{i=1}^{n} \frac{x}{x_{\bullet i}} (\frac{x_{ki}}{x_{k\bullet}} - \frac{x_{li}}{x_{l\bullet}})^2 \tag{2.6}$$

with

$$x_{\bullet i} = \sum_{j=1}^{p} x_{ji}$$

,

$$x_{k\bullet} = \sum_{i=1}^{n} x_{ki}$$

, and

$$x = \sum_{i=1}^{n} \sum_{j=1}^{p} x_{ji}$$

where c_i and c_k are two cliques, n is the number of synonymous terms, $x_{i\bullet}$ and $x_{k\bullet}$ numbers of terms in c_i and c_k, $x_{\bullet j}$ the frequency of term j and x the sum of the frequencies of all terms (or the total number of terms in all

---

[1] available at http://dico.isc.cnrs.fr/

cliques). Using one of the examples used in (Bouroche and Saporta, 2005), the headword fast has many cliques, including:

- Clique 12: express, fast, quick, rapid, swift

- Clique 17: fast, fastened, fixed, secure

- Clique 23: fast, firm, lasting, stable, tight

The distance computed with the equation 2.6 is providing the following distances between terms:

$$d(c_{23}, c_{12}) = 1.7357$$
$$d(c_{17}, c_{23}) = 0.0170$$
$$d(c_{12}, c_{17}) = 1.17213$$

The analysis of these distance show that the distance is properly representing the semantic categorization.

In order to represent in a small amount of dimensions the semantic space, a principal component analysis is used by Ploux et al. (Ploux and Victorri, 1998) (Ploux et al., 2003). These coordinates are used to represent the semantic space of synonyms. To split the space into clusters, a hierarchical classification is obtained via the calculation of the Ward's distance of cliques' coordinates. A word belongs to a cluster if all the cliques that contain it belong to this cluster. Ward proposed in 1963 (Cornish, 2007) a clustering procedure seeking to form partitions in a manner that minimizes the loss associated with each grouping, and to quantify that loss in a form that is readily interpretable. At each step in the analysis, the union of every possible cluster pair is considered and the two clusters whose fusion results in minimum increase in "information loss" are combined. Information loss is defined by Ward in terms of an error sum-of-squares criterion, (ESS). In order to describe the idea behind Ward's proposal we can consider a small example of 10 cliques having distances of (2, 6, 5, 6, 2, 2, 2, 2, 0, 0, 0). The loss of information that would result from treating the ten distances as one group with a mean of 2.5 is represented by ESS given by:

$$ESS_{group} = (2-2,5)^2 + (6-2,5)^2 + ... + (0-2,5)^2 = 50.5$$

On the other hand, if the 10 cliques are classified according to their distances into four sets:

ESS_{group1} {0,0,0}

ESS_{group2} {2,2,2,2}

ESS_{group3} {5}

ESS_{group4} {6,6}

The ESS can be evaluated as the sum of squares of four separate error sums of squares:

$$ESS_{group} = ESS_{group1} + ESS_{group2} + ESS_{group3} + ESS_{group4} = 0.0 \quad (2.7)$$

As equation 2.7 shows, the clustering of the 10 cliques into 4 clusters is resulting in no loss of information. An example of semantic space for the synonyms of the headwork good is shown Figure 2.25.

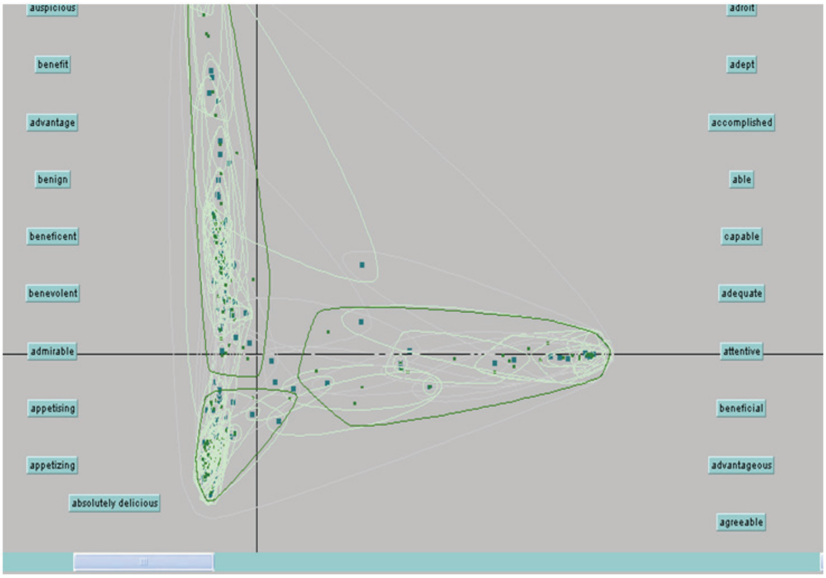


Figure 2.25. Example of semantic space for the head word *"good"*

Having defined the key characteristics of an analysis based on synonyms, the second research approach consists of analyzing the contexonyms. Ploux et al. (Ploux et al., 2003) define contexonyms: "Contexonyms are relevant contextually related words for a target word. By context, we mean a certain number of neighboring words of the target word (from a small-sized window to one or more paragraphs). Unlike synonyms or antonyms, contexonyms are not symmetric or transitive (i.e., when target word W has contexonyms c_1; c_2; ... ; c_n. W is not necessarily a contexonym of $c_{i(1 \leq i \leq k)}$, and this is also true between $c_i s$)." On the opposite of synonyms, contexonyms are often from mixed grammar categories. Contexonyms are more fuzzy than synonyms and they also evolve faster. For this reason, a contexonym requires a training corpus. More adequate the training corpus is, the more relevant and robust the contexonyms obtained from it will be. An adequate corpus can mean for example a corpus of scientific text in mechanical engineering if the goal is to design a mechanical device. The approach described above used to construct the semantic map also applies to the contexonyms. There are other steps necessary to develop a contexonym table (Ploux et al., 2003). First for a given corpus, co-occurrences of all types in a defined passage are counted and stored. A word-association table is constructed considering that each headword has children that are arranged in descending order of co-occurrence with the headwords. Children with small co-occurrences of the global frequency of the headwords are removed. In addition, portions of children of the headwork are selected and rarely co-occurring children are cut off. Lastly a filtering process is used to remove children that are considering the headwords. Using the resulting contexonyms table, cliques are computed followed by the computation of the distance between cliques, then the principal component analysis is applied followed by the clustering method of Ward. This section has provided a summary of the processes involved in the creation of semantic maps both for synonyms and contexonyms.

2.7 Summary of concepts reviewed in state of the art

This section aims at clarifying the connections between the different reviews provided in Chapter 2. Section 2.1 summarized relevant work on design methodologies: systematic approach (Pahl and Beitz, 2007), conceptual design (French, 1999), total design (Pugh, 1991). Engineering design methodologies are compared with Systems Engineering practices. A technical system is defined as a set of interrelated components which interact with each other to accomplish a common purpose. The issue of complexity is approached. Model-based Systems Engineering is viewed as a solution to handling complexity. Thus, the system modeling language, SysML, is described (OMG, 2010). It is shown that SysML fits the descriptions of function, behavior and structure, and therefore its use in engineering design is justified. Furthermore, because of the model centric aspect of the language, this chapter presents that this would allow verification and validation of the coherency of each phase of Systems Engineering, and as a consequence, of Engineering Design. The knowledge of conceptual design is described with Gero's FBS model (Gero, 1990) in Section 2.2. This model developed around the concepts of function, behavior and structure is aligned with the processes of conceptual design. Furthermore, the similarity between the concepts of Gero and SysML strengthen the application of this language to conceptual design.

Then, the scope is narrowed down from conceptual design to reviews related to the main issues of this dissertation, i.e. requirements engineering (Section 2.3) and the synthesis process (Section 2.4). Practices in requirements engineering and Recursive-Object-Model (Zeng, 2008) are presented as it approaches the issue of linguistic requirement clarification studied in this thesis.

Concerning the synthesis process, Section 2.4 provides reviews of two approaches of creativity: TRIZ (Altshuller, 1984) and the CK theory (Hatchuel and Weil, 2008). The taxonomy of functions from Hirtz and Stone (Hirtz et al., 2002), Stone's design repository (Stone et al., 2010) and the automating of the synthesis process from Campbell (Kurtoglu and Campbell, 2009) are reviewed as strong bases for the contribution of this dissertation.

Section 2.5 and 2.6 provide descriptions of the disciplines applied to requirements clarification and synthesis of solutions: Knowledge Representation and Computational Linguistics.

In Section 2.5, Knowledge Representation (KR) is defined and the main concepts used to build computer readable ontologies are expressed. Then, KRs applied to conceptual design are reviewed. This thesis considers the ontology of systems from Tudorache as the closest to the research problem of automating the synthesis process (Tudorache, 2006).

In Section 2.6, mostly presents the work of Ploux after a brief historical background of the discipline (Ploux and Ji, 2003). Ploux semantic atlas is considered for application in our research work because it adopts a radically new paradigm compared to other approaches treating issues of meaning. This paradigm expresses the idea of continuity between the multiple meanings of a word, continuity in polysemy, whereas other approaches consider meanings of a word as completely non-related, discrete polysemy. Moreover, the semantic atlas is based on the mathematical concept of clique from graph theory. This concept enables the construction of a distance of meaning between cliques of synonyms or contexonyms of a word, a metric.

This metric is used for clarifying the meanings of requirements. Additionally, the semantic atlas is used to create connections of meaning between functions and components during the synthesis process. Chapter 3 presents how this is implemented as contributions.

3. Contributions of this thesis

3.1 Model of Conceptual Design and Object - Oriented Design paradigm

According to our understanding of Conceptual Design of systems, Gero's FBS model contains some points which need to be updated according to current knowledge about model-driven engineering. Apart from the fact that Gero's model did not include a requirement phase and that the documentation phase has now become secondary when compared to the models themselves, a major point needs to be clarified concerning the synthesis phase of conceptual design. In fact, Gero states that the only possible link between function and structure is through the expression of behaviour (Gero and Kannengiesser, 2004). We argue here that it is possible to create "embryos" of structures out of functions only. We call these preliminary structures generic or abstract structures. Similarly to abstract classes in object oriented programming, their aim is only to encapsulate each atomic function of the system into one or more of the six families of organs from prior works in our research group (Coatanéa, 2005) and in agreement with the bond graph theory. Therefore, we propose the RFBS model shown in Figure 3.1. This model represents the conceptual design process in a practical way. It corresponds to the way conceptual solutions are formulated using SysML and our computer application prototype: OPAS, guide for the synthesis of conceptual solutions. This model also allows us to notice the strengths and limitations of the method. The RFBS model was first introduced in Publication II and then described more precisely in Publication III.

- R is the set of constraints and performance criteria required

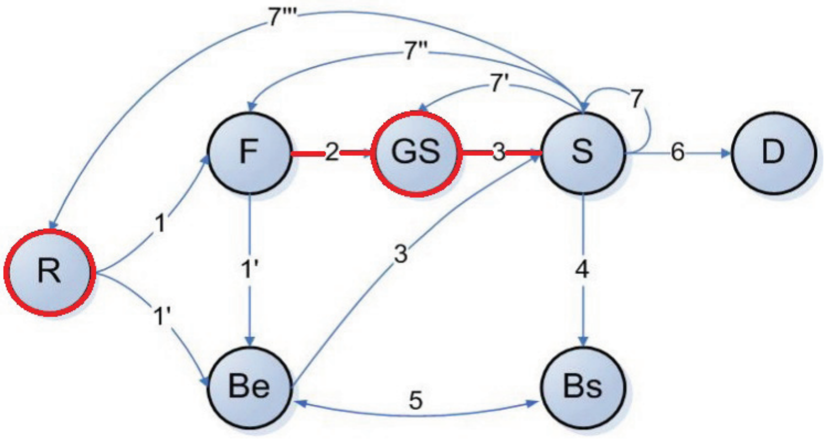


Figure 3.1. The RFBS Model

- F represents a set of functions, the necessary knowledge in order to be able to explain what the system should do according to requirements, thus F is derived from R

- B_e is the expected behaviour of the system, specifically the set of variables showing how the system should work, Be is set according to Requirements and Functions,

- GS is the representation of Generic Structure, specifically abstract classes encapsulating function and their intrinsic attributes, GS is derived from F,

- S is the set of variables representing the physical structure of the system, S specializes GS according to B_e

- B_s is the set of variables enabling the representation of the effective behaviour of the system, thus its "actual" behaviour,

- D represents the transfer of the models to the next stage of design: detailed design

We have defined seven types of processes occurring during conceptual design:

1. (a) Formulation type 1 (process 1): transforms the design problem, expressed in requirements (R), into functions (F) that the system should

   provide

   (b) Formulation type 2 (process 1’): transforms the design problem, expressed in function (F) and requirements (R), into behaviour (B_e) that is expected to enable this function with the performance criteria set by the requirements

2. Pre-synthesis: transforms the functional architecture of the system (F) into a generic structure (GS) using abstract organs

3. Synthesis: specializes GS according to the expected behaviour (B_e) into a solution structure (S) that is intended to exhibit this desired behaviour

4. Analysis: derives the “actual” behavior (B_s) from the synthesized structure (S)

5. Evaluation: compares the behavior derived from structure (B_s) with the expected behaviour to prepare the decision if the design solution is to be accepted

6. Detailing: prepares all drawn models for the detailed design phase (from work classes into technology involvement)

7. (a) Reformulation type 1 (process 7): addresses changes in the design state space in terms of structure variables or their ranges of values

   (b) Reformulation type 2 (process 7’): addresses changes in the design state space in terms of abstract organs or generic structure variables or their ranges of values

   (c) Reformulation type 3 (process 7”): addresses changes in the design state space in terms of function variables or their ranges of values (this reformulation induces automatically changes in the expected behaviour)

   (d) Reformulation type 4 (process 7”’): addresses changes in the design state space in terms of requirement variables or their ranges of values (this reformulation involves discussion with the client for finding agreement)

Preliminary study of Gero's FBS model (Gero, 1990) (Gero and Kannengiesser, 2004) and SysML was realsied in Publication I.
As this model represents needed knowledge during conceptual design, we shall consider matching concepts of this model with SysML, modeling language for designing systems. SysML diagrams can be mapped with each state of the RFBS model. The first stage of conceptual design involves an important part of problem solving issues: the refinement of ill-defined problems with the refinement of informal requirements into formal requirements represented with SysML requirement diagrams. This type of diagram represents requirements hierarchically and in order of importance. Requirements diagram correspond to state R of the RFBS model. Concurrently, the verbal requirements are analyzed in order to extract the expected functionalities of the future product. This analysis allows the creation of use case diagrams presenting the system services and its surrounding environment. More detailed insight into the needed functionalities enables the creation of functional architectures developing different functional structures represented as "black boxes" with SysML block definition diagrams.

Following the problem refinement phase of conceptual design, the synthesis of concepts phase has the aim to transform functional models into structural models enabling the system's organisation in sub-systems and components. Generic solutions are in the form of block definition diagrams and then are specialized into physical components in the form of block definition diagrams and internal block diagrams.

The evaluation phase of conceptual design can be realised by comparing behavioral diagrams of different types: activity diagrams, sequence diagrams and state-machine diagrams. These diagrams are derived from both requirements and functional architectures and represent the behavior expected from the system. Similarly, the same set of diagrams represents the structural behavior of the system and is derived by designers from the block diagrams representations. Additionally, systems engineers represent the connections between variables involved in the structure of the system with parametric diagrams. Table 3.1 summarises the correspondance between RFBS concepts of Knowledge and SysML notations.

RFBS states	Corresponding SysML diagrams	Description of the diagram type
R	Requirement diagram (req)	Requirements made with or coming from stakeholders are first defined by req. Stereotypes of requirements can be defined in order to classify requirements into, for example, functional and non-functional.
	Block Definition Diagram (bdd)	The description of the boundaries of the system “as-is", its environment and the relations between the system and actors of its environment can be represented with bdd
F	Use Case diagram (uc)	Services provided by the system to actors are described in a uc. uc is at the interface between R and F as a use case should $<< refine >>$ a functional requirement and $<< trace >>$ non-functional requirements for traceability reasons. It is also possible to describe the functional decomposition of services with technical functions in uc with relation $<< include >>$
GS	Block Definition Diagram (bdd)	Each technical functions at lowest level of the functional tree decomposition is encapsulated into an abstract organ.
B_e	Activity diagram (act)	act represents the behaviour of the System and the order in which technical functions should appear to realise the service.
	State Machine diagram (stm)	stm represents the different states in which the System can be during its operation

S	Block Definition diagram (bdd)	System physical structure is defined by bdd. Each block of this bdd is derived from GS and specialises GS with the states variables discovered while modeling B_e. Such block contains internal variables (attributes of the block) and defines the necessary input variables for the function to be achieved as well as the provided output variables (prototype of operation of the block).
	Internal Block diagram (ibd)	Whereas bdd defines the hierarchical structure of the system, ibd represents the interactions between its components and the way they exchange matter, energy or information flows.
B_S	Activity diagram (act)	Similar than for B_e but for the analysis of the behavior of the concepts of solution.
	State Machine diagram (stm)	
	Sequence diagram (seq)	Analysis of the sequence of flow exchange between components of the system
B_S	Parametric diagram (par)	When all variables are known par represents their use in the physical equations governing the system in a graph. The analysis of the variables involved in most of these equations as well as the causal analysis of interaction between variables helps finding the variables of most importance for the efficiency of the system (performance variables).
D	every diagram	The entire project modeled with SysML should be used for the detailed design phases of each discipline involved in the design (e.g. mechanical, electrical and software engineering).

Table 3.1. Correspondance entre les concepts RFBS et les diagrammes SysML

3.2 Requirements Engineering

3.2.1 Protocol for semantic disambiguation

The first step of the semantic disambiguation process consists in gathering all synonyms and contexonyms for each term composing the sentence. For each term, the set of synonyms represents all possible meanings of this term. This is the divergent part of this protocol. It aims at searching for all possible meanings of the terms used in the sentence. The set of contexonyms represents the different words that are frequently associated with the term according to a certain corpus of texts as presented in section 4.2. It is important to note that the corpus for which Ploux's algorithm for finding contexonyms have been trained is based on an English corpus maintained by Project Gutenberg, which includes literature, essays, and other writings (Ploux et al., 2003). This database extracted from Project Gutenberg was then combined with a separate database trained on the British National Corpus. Even though this corpus contains over 300 million words, one should notice that it is not scientifically oriented. The matter of the relevance of this corpus or whether a specific corpus dedicated to science and physics is discussed further in the conclusion of this section.

In a second step, cliques, representing the minimal unit of meaning a word, are formed from both the set of synonyms and the set of contexonyms for each term contained in the definition of the pressure regulator.

The third step of this protocol represents the core of the disambiguation process. This phase consists on the elimination of irrelevant cliques. This elimination is done by comparing synonyms of each term of the sentence. If common synonyms are found between terms in the sentence, then the cliques where these synonyms appear will be kept as potential meaning of the terms of the sentence. Otherwise, the cliques will be eliminated. The same principle is then applied to contexonyms. This is the convergent part of this protocol.

The following paragraph addresses this protocol through the concrete example of the analysis of the definition of a pressure regulator. This example should provide a concrete application and understanding of how this process is being held.

Following this protocol we have gathered the lists of synonyms and contexonyms related to each term of the definition of a pressure regulator. These lists can be seen in Table 7, Annex A of Publication IV. This represents the divergent part of this protocol as the lists should be as exhaustive as possible. Our sources for this collecting process are available at $dico.isc.cnrs.fr$ from (Ploux and Ji, 2003) and at $thesaurus.com$.

The next stage consisting in computing the cliques of synonyms and the cliques of contexonyms in order to obtain a metric and a relative distance between cliques and the term subject of the request is shown Figure 5a and 5b through the example of the term “pressure”. This stage is neither divergent nor convergent as it consists in sorting the data obtained in a manner that can be further analyzed, i.e. by placing the metric on words thanks to cliques.

The third stage is presented through Figure 6 of Annex A and involves the creation of links between different terms of the definition through their common synonyms or contexonyms. Figure 6 shows that these links can be done through words considered in one part as synonyms and in the other part as contexonyms. This makes sense as we are looking for disambiguation of terms within the specific context of this definition sentence. Thus, a contexonym of one term of this sentence (e.g. a word often associated with this term in the corpus), should be found as the synonym of another term of this specific sentence.

As a result of this protocol we obtain Table 8 of Annex A which contains the remaining relevant cliques of synonyms and contexonyms. These cliques form the remaining meanings of the term employed within the context of this sentence. Table 8 also presents an auto-evaluation of the application of this protocol through the calculation of the ratio between the remaining cliques and the initial number of cliques associated with a term.

As we noticed through the example of the definition of the pressure regulator presented in section 5.3.2. of Publication IV, the semantic disambiguation process allows ruling out irrelevant meanings of a word within the specific context of the sentence. In the chapter we are now addressing the question of the sufficiency of this process. Does this protocol provide no more confusion possible for the meaning of a word in its context?

From the analysis of the results of the case study presented in Table 8 of Annex A, we notice that in most of the cases, a small number of cliques remain considered as relevant compared with the initial number of cliques. In fact, analyzing this ratio shows that in the major cases the ambiguity is reduced of an average of 90%. Except for the terms “automatically” and “gas” which had initially only one possible meaning, more precisely one clique where this ratio is indeed of 100% (this unique unit of sense is kept as relevant).

Nevertheless, this analysis is conducted in this case on a small amount of terms which cannot validate empirically this protocol but it is noticeable that rather relevant results were obtained. Therefore we argue that this method has strong potential for avoiding misunderstandings within the designing team due to multiple possible meanings of a word. We discuss these potential benefits in the following and concluding section as well as the benefits from combining both ROM and semantic disambiguation approaches.

Disambiguation has a lot of importance at the early stages of design and particularly during the first phases of requirements engineering: elicitation and representation. The approach proposed with Recursive Object Modeling constructs a formal representation of requirements expressed initially in natural language. This enables a preliminary processing of the information provided at the phase of elicitation of the requirements. Moreover, ROM provides the possibility to prioritize and focus on the most important concepts, objects, of the requirements and to precise these concepts through a question/answer process.

Publication V provides an implementation for clarifying requirements and gathering information about the environment to which the system should be related to. This implementation is realized with a question/answer process. Questions on the meaning of specific terms used in a requirement are asked automatically from ROM. Answers to these questions are sent as requests to search engines and patent search website. After a selection of relevant answers according a metric of comparison in the similarity of keywords, the remaining answers provide information about existing systems and the environment. Such information should be used to model the environment of the “system to-be”.

3.3 Steps towards the automated synthesis of design

In this section, we place our focus on the explanation of our vision of the synthesis process. We consider two phases in the process: pre-synthesis and synthesis. Pre-synthesis is related with the linkage of functions with generic components which would serve as the basis for concepts of solution. Our viewpoint on assisting synthesis with a computer application is explained in Publication II and Publication III.

3.3.1 Knowledge Representation of the concepts used for the synthesis

The main point of this section is to show what we consider as the necessary knowledge needed for a machine to be able to interact with humans at the same level of understanding than them. In this part we provide an ontology of the needed concepts during the synthesis of conceptual design solutions. Figure 3.2 shows the concepts contained in this ontology and their relationships.

This ontology can be integrated with the work of Tudorache. It is a frame-based ontology developed in OWL/RDF(S) with Protégé, an ontology editor. It is planned to publish this ontology and make it available for the community. This chapter presents the core concepts of this ontology. As the synthesis of solutions starts from functional architectures, it is important to have knowledge about function. A function is, from our viewpoint, represented by a verb and is a connection between an initial situation and a final situation. Hirtz have established a taxonomy of standard functions (Hirtz et al., 2002) and according to empirical studies, 94% of the functions could be described using the functional basis proposed by Hirtz et al. (Saeema, 2005). These studies are based on a total of 207 descriptions of functions representing various sub-assemblies of an aero-engine and an aircraft design taken from two different companies. Our approach is to use this taxonomy at its full potential but with leaving freedom of vocabulary to designers. Therefore, we will transform the functional description made by designers with their own vocabulary into a standard functional description. In order to do this standardization we send requests to an online semantic atlas (Ploux and Ji, 2003) based on the concept of contexonyms. After this step, the next is to associate each standard function of the lowest level of description to one or more of the six families of abstract organs (Coatanéa, 2005). These organs can be then specialized according to the type of energy they involve. In fact, we also use the taxonomy of fields from Hirtz in order to determine the type of variables involved in the input/output of physical components. Our ontology is a mid-level ontology and integrates Hirtz taxonomies of functions and fields of energies. The concept of standard function has the elements of the functional taxonomy as instances of our ontology. Similarly, the concepts of energy and variables integrate the taxonomies of generalized variables according to fields of energies.

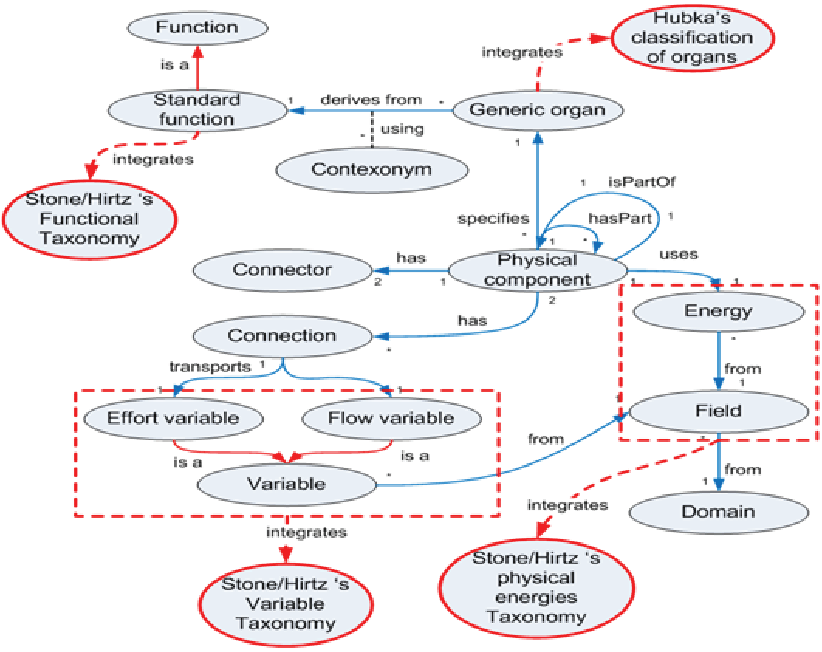


Figure 3.2. Representation of the ontology of concepts involved in the synthesis of conceptual solutions

3.3.2 Library of components

Off-the-shelf components are concrete physical components implementing the classification of generic organs (Karnopp et al., 1990) (Top, 1993) (Coatanéa, 2005) according to a specific physical field. For instance, if we use the classification of organs in the field of electronics, we obtain the following:

- effort variables are expressed as an electric potential difference with Volts as SI unit.

- flow variables are expressed as an electric current with Amperes as SI unit.

- storage mechanisms are expressed in the form of capacitors or inductances

4. Conclusion and Perspectives

The research presented in this manuscript shows:

- that applying semantic analysis from the domain of computational linguistics can be of great use in the following conceptual design phases:
  - Elicitation and Representation of Requirements, in the fact that it ensures a clear understanding of the design problem
  - Synthesis of concepts of solution, because it enables the mapping, through their shared meanings, between a technical function and generic components fulfilling this function
- the possibility of enhancing such analysis by computer assistance using tools from computational linguistics such as: syntactic, lexical or semantic analyzers,
- the importance of using a formalism for knowledge representation and, moreover,
- the importance of the digital representation of this knowledge in a computer-readable form in order, for example at the synthesis stage of conceptual design, for computers to be able to assist designers by using the same key concepts of knowledge.

This thesis covers an important part of the conceptual design process, since only the evaluation of concepts of solution is not reviewed here in detail. Indeed, the purpose of this thesis is primarily to observe the preliminary phases of design with the viewpoints of semantics and knowledge formalization for the development of concepts of solution. The evaluation phase is more a complex mathematical problem of multi-criteria optimization and decision support. Indeed, in order to achieve this evaluation phase, the meaning of the problem/solutions data must be analyzed and formulated as to facilitate this assessment. It is for these reasons that this thesis mainly covers the phases of requirements elicitation and representation as well as the synthesis phase of solution concepts. Moreover, these phases contain many textual elements that can not be reduced to a mathematical formalism because of their multiple meanings. In this sense, this research tends to facilitate and prepare the evaluation phase of concepts in two ways:

- first, in focusing on the critical points contained in requirements; points in which ambiguities related to the multiple possible meanings persist despite the addition of details to clarify the requirements. This makes possible to give special attention to variables derived from this part of the problem during the confrontation between concepts of solution and their validity with regard to the design problem.

- second, in assisting the design team during the exploration of the solution space by a systematic and partly automated synthesis of multiple concepts of solution.

The concept of function takes a central role in this thesis. Nevertheless, the description of function as interface between initial and final situations does not fit perfectly with the description of Stone (Hirtz et al., 2002). The viewpoint of this thesis is that both descriptions are not contradicting.In fact, it is possible to describe non-transformative functions with both initial and final situations. For example, a garbage bin is basically a storage organ as it does not necessarily transform material or energy. Nevertheless, when in use, the initial situations of the garbage bin can be either full or not. In the case it is not full; the user will want to put a certain volume of trash from the environment into the garbage bin. As a result, the final situation after use, in this specific case, should be that the garbage bin contains this certain volume of trash in addition to what it was containing initially. On the range of applicability of the approach proposed in this thesis, it is noticeable in Publication II that OPAS was able to find semantic links between function and storage organs. Nevertheless, the range of applicability of OPAS needs to be further tested. The design repository (Stone et al., 2010) offers an interesting platform for such tests and it is integrated in the perspectives for future research.

This work paved two ways for future research:

- the first one concerns deepening the directions proposed here for the automated synthesis of concepts of solution,

- the second one is on expansing the use of computational linguistics for the analysis of requirements.

In the case of the first research direction, the synthesis of embryos of solution with generic components could be continued with the study of the possible relationships between components. Generic components obtained during the synthesis phase should be further specialized by applying these components to various domain specific areas. This specialization should be possible by the creation of libraries of components-of-the-shelf being specific to a certain domain of application (e.g. electronic, hydraulic, mechanical). Then, this would enable the combinatorial synthesis of many bricks solutions integrated with each other. However, an analysis of the relationships between components would limit the number of concepts generated, as the study of incompatibility between the output variables of a component and input variables of another component to be connected to the previous would eliminate considering such combination as a concept of solution.

In the case of the second research direction, it seems that the semantic analysis of requirements based on the context expressed in textual requirements could provide many other benefits than the clarification and disambiguation of these requirements. Indeed, the clarification process and the similarity metric between texts proposed in this research could be used across the entire requirement document. This document is usually composed of different categories under which similar requirements are grouped (e.g. ergonomics, safety, security). Then, the use of the contextual analysis of meaning and the similarity metric could detect requirements belonging to several categories at once but mentioned only in one of these categories in the requirements document. This would avoid forgetting to consider this requirement when analyzing the document with a focus related to a specific category of requirements.

In fact, it appears that many industrials now spend part of their resources and show interests in the sense of these two directions.

Bibliography

Altshuller, G., *Creativity as an exact science*, Gordon & Breach, Luxembourg, 1984.

ANSI/EIA-632-1998, *Processes for engineering a system*, Approved 7 January 1999, Reaffirmed 2003, 2003.

Antonsson, E. K. and Cagan, J., eds., *Formal Engineering Design Synthesis*, Cambridge University Press, 2001.

Asimow, M., *Introduction to Design*, Englewood Cliffs, NJ: Prentice-Hall, 1962.

Bayazit, N., *Investigating Design: A Review of Forty Years of Design Research*, Design Issues, vol. 20(1), pp. 16–29, 2004.

Berners-Lee, T., *Relational Databases on the Semantic Web*, W3C, URL `http://www. w3.org/Design Issues/RDB-RDF.html`, September 1998.

Berners-Lee, T., Hendler, J. and Lassila, O., *The Semantic Web: A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities*, Scientific American Magazine, URL `http://www.scientificamerican.com/article.cfm?id=the-semantic-web`, May 2001.

Bouroche, J. M. and Saporta, G., *L'analyse des données*, Presse Universitaire de France, 2005.

Buyssens, E., *La communication et l'articulation linguistique*, Bruxelles: Presses Universitaires / Paris: Presses Universitaires de France, 1967.

Chakrabarti, A., ed., *Engineering Design Synthesis: Understanding, Approaches and Tools*, Springer Verlag, London, ISBN 1852334924, 2002.

Chakrabarti, A., Shea, K., Stone, R., Cagan, J., Campbell, M., Hernandez, N. V. and Wood, K. L., *Computer-Based Design Synthesis Research: An Overview*, Journal of Computing and Information Science in Engineering, vol. 11(2), p. 10, doi:10.1115/1.3593409, June 15 2011.

Chakrabarti, A. and Tang, M. X., *Generating Conceptual Solutions on FuncSION: Evolution of a Functional Synthesiser*, Artificial Intelligence in Design '96, pp. 603–622, 1996.

Coatanéa, E., *Conceptual Modelling of Life Cycle Design - A Modelling and Evaluation Method Based on Analogies and Dimensionless Numbers*, Ph.D. thesis, Helsinki University of Technology and Université de Bretagne Occidentale, Espoo, Finland, October 2005.

Cornish, R., *Mathematics Learning Support*, chap. 3.1. Cluster Analysis, 2007.

Cross, N., *Forty years of design research*, Design Studies, vol. 28(1), pp. 1 – 4, ISSN 0142-694X, doi:10.1016/j.destud.2006.11.004, 2007.

Dagan, I. and Itai, A., *Word sense disambiguation using a second language monolingual corpus*, Computational Linguistics, vol. 20(4), pp. 563–596, 1994.

Dardy, F., Teixido, C., Brissard, J. L. and Polizzi, M., *Guide de la Compétitivité Industrielle: De la Conception à la Production*, Paris, 1st edn., ISBN 978-2-206-08662-0, 5 october 2003.

Davis, R., Shrobe, H. and Szolowits, P., *What is a Knowledge Representation?*, AI Magazine, vol. 14(1), pp. 17–33, 1993.

Dee, M., Martin, J. and Group, D., *Better SE for INCOSE*, URL `http://sites.google.com/site/syssciwg/projects/better-se-for-incose`, 2011.

Dekker, D. L., *Engineering design processes, problem solving and creativity*, in *Frontiers in Education Conference Proceedings*, vol. 1, pp. 3a5.16 – 3a5.19, IEEE, Atlanta, USA, ISBN 0-7803-3022-6, 1-4 Nov. 1995.

Deneux, D., *Méthodes et modèles pour la conception concourante*, Habilitation à diriger des recherches, University of Valenciennes and Hainaut Cambrésis, Jan. 2002.

Dorst, K. and Cross, N., *Creativity in the design process: co-evolution of problem–solution*, Design Studies, vol. 22(5), pp. 425–437, 2001.

Eisner, H., *Essentials of Project and Systems Engineering Management*, John Wiley & Sons, Inc., New York, NY, USA, 1st edn., ISBN 0471148466, 1996.

Estefan, J. A., *Survey of Model-Based Systems Engineering (MBSE) Methodologies*, Jet Propulsion, vol. 25, pp. 1–70, 2008.

French, M., *Conceptual design for engineers*, Springer, 3rd edn., ISBN 9781852330279, 1999.

Gero, J. S., *Design Prototypes: A Knowledge Representation Schema for Design*, AI Magazine, vol. 11(4), pp. 26–36, 1990.

Gero, J. S. and Kannengiesser, U., *The situated function-behaviour-structure framework*, Design Studies, vol. 25(4), pp. 373–391, ISSN 0142-694X, doi: 10.1016/j.destud.2003.10.010, 2004.

Gruber, T. R., *Toward principles for the design of ontologies used for knowledge sharing*, International Journal of Human-Computer Studies, special issue on the role of formal ontology in the information technology, vol. 43(5-6), pp. 907–928, originally in N. Guarino and R. Poli (Eds.), International Workshop on Formal Ontology, Padova, Italy, August 1993.

Hatchuel, A. and Weil, B., *A new approach of innovative design: An introduction to CK theory*, in *ICED'03*, p. 124, Stockholm, Sweden, August 19-21 2003.

Hatchuel, A. and Weil, B., *Entre concepts et connaissances : éléments d'une théorie de la conception*, in Hatchuel, A. and Weil, B., eds., *Les nouveaux régimes de la conception : langages, théories, métiers*, pp. 115–131, Vuibert, Cerisy, 2008.

Hirtz, J., Stone, R. B., McAdams, D. A., Szykman, S. and Wood, K. L., *A functional basis for engineering design: Reconciling and evolving previous efforts*, Research in Engineering Design, vol. 13(2), pp. 65–82, 2002.

Hubka, V. and Eder, W. E., *Design Science*, Springer, 1996.

Hull, E., Jackson, K. and Dick, J., *Requirements Engineering*, Springer Science, 2nd edn., 2005.

INCOSE-SEH-WG, *Systems Engineering Handbook*, INCOSE, version 2a, June 2004.

Karnopp, D. C., Margolis, D. L. and Rosenberg, R. C., *System Dynamics: A unified Approach*, Wiley & Sons, New York, second revised edn., 1990.

Kitamura, Y., Takafuji, S. and Mizoguchi, R., *Towards A Reference Ontology for Functional Knowledge Interoperability*, in *Proceedings of ASME IDETC/CIE*, 2007.

Kurtoglu, T. and Campbell, M. I., *Automated synthesis of electromechanical design configurations from empirical analysis of function to form mapping*, Journal of Engineering Design, vol. 20(1), pp. 83–104, 2009.

Kurtoglu, T., Campbell, M. I., Bryant, C. R., Stone, R. B. and McAdams, D. A., *Deriving a component basis for computational functional synthesis*, in *Proceedings of International Conference on Engineering Design, ICED'05*, p. 15, Melbourne, August 15-18 2005.

Lin, D. and Pantel, P., *Concept discovery from text*, in *Proceedings of the 19th international conference on Computational linguistics*, pp. 1–7, 2002.

Liu, Y. C., Chakrabarti, A. and Bligh, T. P., *Further Developments of FuncSION*, Artificial Intelligence in Design '00, pp. 499–519, 2000.

Lotter, B., *Manufacturing Assembly Handbook*, Butterworths, Boston, 1986.

Lusch, R. F., Vargo, S. L. and O'Brien, M., *Competing through service: Insights from service-dominant logic.*, Journal of Retailing, vol. 83(1), pp. 2 – 18, 2007.

Masson, P. L. and Hatchuel, A., *Les processus d'innovation : conception innovante et croissance des entreprises*, Hermès, Lavoisier, 2006.

Micouin, P., *Propositions pour la définition et la mise en place de processus d'ingénierire de systèmes : application au cas de la conceptin concourante dans le secteur de l'automobile.*, These, Arts et Métiers ParisTech, Sep 2006.

Miles, L. D., *Techniques of Value Analysis and Engineering*, McGraw-Hill Book Company, New York NY, 1961.

Motte, D., *A Review of the Fundamentals of Systematic Engineering Design Process Models*, in *Proceedings of the 10th International Design Conference, DESIGN 2008*, vol. 48, pp. 199–210, Design Society, Dubrovnik, Croatia, 19-22 May 2008.

Motte, D., *On the Fundamentals of the Engineering Design Process*, Ph.D. thesis, Lund University, Department of Machine Design, Lund, Sweden, 10 June 2011.

NF-X50-151, *Analyse de la valeur, analyse fonctionnelle - Expression fonctionnelle du besoin et cahier des charges fonctionnel*, AFNOR, December 1991.

OMG, *SysML specification V. 1.2*, Object Management Group, URL `http://www.sysml.org/docs/specs/OMGSysML-v1.2-10-06-02.pdf`, June 2010.

Otto, K. N. and Wood, K. L., *Product Design : Techiniques in Reverse Engineering and New Product Development*, Upper Saddle River, EUA : Prentice-Hall, 2001.

Pahl, G. and Beitz, W., *Engineering Design: A Systematic Approach*, Design Council, London, 1st edn., 1984.

Pahl, G. and Beitz, W., *Engineering Design – A Systematic Approach*, Springer, London, 3rd edn., 2007.

Ploux, S., *Modélisation et traitement informatique de la synonymie*, Linguisticae Investigationes, vol. XXI(1), pp. 1–128, 1997.

Ploux, S. and Ji, H., *A model for matching semantic maps between languages (French/English, English/French)*, Computational Linguistics, vol. 29(2), pp. 155–178, June 2003.

Ploux, S., Ji, H. and Wehrli, E., *Lexical Knowledge Representation with Contexonyms*, in *Proceedings of the 9th Machine Translation*, pp. 194–201, 2003.

Ploux, S. and Victorri, B., *Construction d'espaces sémantiques à l'aide de dictionnaires de synonymes*, Traitement automatique des langues, vol. 39(1), pp. 161–182, 1998.

Pottier, B., *Linguistique générale*, Paris: Klincksieck, 1974.

Pugh, S., *Total design: integrated methods for successful product engineering*, Pearson Education, Addison-Wesley Pub. Co., ISBN 9780201416398, 1991.

Ramo, S., *Definition of Systems Engineering in System Engineering Manual*, Federal Aviation Agency, U.S.A., May 2007.

Saeema, A., *Encouraging reuse of design knowledge: a method to index knowledge*, Design Studies, vol. 26(6), pp. 565–592, November 2005.

Schön, D. A., *The reflective practitioner - How professionals think in action*, Basic books, New York, 1983.

Simon, H. A., *Bounded Rationality and Organizational Learning*, Organization Science, vol. 2(1), pp. 125–134, 1991.

Simon, H. A., *The Sciences of the Artificial*, The MIT Press, 3rd edn., october 1996.

Sowa, J. F., *Conceptual Graphs for a Data Base Interface*, IBM Journal of Research and Development, vol. 20(4), pp. 336–357, July 1976.

Sowa, J. F., *Knowledge Representation: Logical, Philosophical, and Computational Foundations*, Brooks Cole Publishing Co., Pacific Grove, CA, ISBN 0-534-94965-7, 2000.

Sterman, J. D., *Business Dynamics: Systems Thinking and Modeling for a Complex World*, Irwin/McGraw-Hill, 2000.

Stone, R., McAdams, D. and Wie, M. V., *Design Repository*, URL `http://function2.mime.oregonstate.edu:8080/view/index.jsp`, 2010.

Studer, R., Benjamins, V. R. and Fensel, D., *Knowledge engineering: Principles and methods*, Data & Knowledge Engineering, vol. 25(1-2), pp. 161–197, March 1998.

Tomiyama, T., Gu, P., Jin, Y., Lutters, D., Kind, C. and Kimura, F., *Design Methodologies: Industrial and Educational Applications*, Keynote, CIRP Annals—Manufacturing Technology, vol. 58(2), pp. 543–565, 2009.

Tomiyama, T., Meijer, B. R., Holst, B. H. A. V. D. and Werff, K. V. D., *Knowledge Structuring for Function Design*, CIRP Annals STC Design, vol. 52(1), pp. 89–92, 2003.

Top, J. L., *Conceptual modelling of physical systems*, Ph.D. thesis, University of Twente, 1993.

Tudorache, T., *Employing Ontologies for an Improved Development Process in Collaborative Engineering*, Ph.D. thesis, Berlin, November 2006.

Ullman, D. G., *The mechanical design process*, McGraw-Hill, 3rd edn., ISBN 9780072373387, 2002.

Ulrich, K. T. and Eppinger, S. D., *Product Design and Development*, McGraw-Hill/Irwin, 5th edn., ISBN 978-0073404776, May 2011.

Umeda, Y. and Tomiyama, T., *FBS modeling: modeling scheme of function for conceptual design*, in *Proceedings of the 9th International Workshop on Qualitative Reasoning about Physical Systems*, pp. 217–278, Amsterdam, 1995.

VDI, *VDI-Guideline 2221: Systematic Approach to the Design of Technical Systems and Products*, VDI-Verlag, Düsseldorf, 1987.

Weilkiens, T., *Systems Engineering with SysML/UML: Modeling, Analysis, Design*, The MK/OMG Press, Morgan Kaufmann OMG Press/Elsevier, ISBN 9780123742742, 2008.

Yannou, B., *Préconception de Produits*, Mémoire d'habilitation à diriger des recherches. discipline: mécanique, Institut Nationale Polytechnique de Grenoble (INPG), 2000.

Zeng, Y., *Environment-based formulation of design problem*, Transaction of SDPS: Journal of Integrated Design and Process Science, vol. 8(4), pp. 45–63, 2004.

Zeng, Y., *Recursive Object Model (ROM) - Modeling of Linguistic Information in Engineering Design*, Computers in Industry, vol. 59(6), pp. 612–625, 2008.

Aalto-DD 90/2012

This thesis suggests the use of tools from the disciplines of Computational Linguistics and Knowledge Representation with the idea that such tools would enable the partial automation of two processes of Conceptual Design: the analysis of Requirements and the synthesis of concepts of solution. The viewpoint on Conceptual Design developed in this research is based on the systematic methodologies developed in the literature. The evolution of these methodologies provided precise description of the tasks to be achieved by the designing team in order to achieve successful design. Therefore, the argument of this thesis is that it is possible to create computer models of some of these tasks in order to partially automate the refinement of the design problem and the exploration of the design space. In Requirements Engineering, the definition of requirements consists in identifying the needs of various stakeholders and formalizing it into design specifications. During this task, designers face the problem of having to deal with individuals from different expertise, expressing their needs

ECN
Centrale
Nantes

9 789526 046969

ISBN 978-952-60-4696-9
ISBN 978-952-60-4697-6 (pdf)
ISSN-L 1799-4934
ISSN 1799-4934
ISSN 1799-4942 (pdf)

Aalto University
School of Engineering
Department of Engineering Design and Production
www.aalto.fi

BUSINESS +
ECONOMY

ART +
DESIGN +
ARCHITECTURE

SCIENCE +
TECHNOLOGY

CROSSOVER

DOCTORAL
DISSERTATIONS