FILTERING ALGORITHM FOR CLASSIFYING THE FEASIBLE SOLUTIONSPACE FROM DESIGN SPECIFICATIONS AND COMMITMENTS

1.0 Abstract

It has become common practice for engineering designers to adopt a concurrent engineering design approach whereby products are being designed with consideration for their whole life-cycle not just for their use phase. This means that fabrication and assembly issues, product servicing, product disposal and other issues of the product from conception to grave are also taken into account for the product to be competitive in the market during the early stages of design. Products are becoming very complex because of their minitiarization in size and incorporation of a number of disciplines, that are not only engineering related but include other disciplines. Thus engineering design has become a very knowledge-intensive activity, with designers requiring support when taking design decision commitments as they have become too complex for the designer to handle on an individual basis. Several computer aided Decision Support Systems (DSSs) have, as a result, been developed to aid designers in their work. However, a major shortcoming of these systems is that they first allow the designer to take a decision and *afterwards* provide support in the form of consequences of these decisions on the product life-cycle. This tends to result in a lot of iterations and consequently loss of design time until the designer arrives at a life-oriented solution. The aim of this invention is therefore to provide support to the designer *before* a decision is taken so that a lot of iterations are eliminated. The novelty lies in the approach that is adopted to assist the designer and its implementation in an algorithm and eventually in a framework for a DSS. The approach, together with its algorithm and framework implementation, are applicable not only in the field of engineering design but also in other areas involving the processing of specifications and related decision making such as software design, architecture design, service design, planning, etc.

1.1 Abbreviations

ABBREVIATION	DEFINITION
CAGM	Computer Aided Geometric Modelling
DSS	Decision Support System
LCC	Life-cycle Consequence
LCPE	Life-cycle Phase Element
PDE	Product Design Element
PDS	Product Design Specification
PLS	Product Life-cycle Specification
SLS	Sub-assembly Life-cycle Specification
CLS	Component Life-cycle Specification

Figure 10 demonstrates this graphically.

6.0 Application of invention

The approach, together with its algorithm and framework implementation, are applicable not only in the field of engineering design but also in other areas involving the processing of specifications and related decision making such as software design, architecture design, service design, planning, etc. This can be done by changing the Reusable Element Library to one that is related to the application at hand. In product design, the invention is particularly useful when applied to a system that supports designers that design multidisciplinary products which include i) the combination of more than one discipline and ii) life-cycle knowledge in their development. The reason for this is that since such products are complex, very knowledge-intensive and include many specifications for each of the disciplines, then it is difficult for the designer to handle all this without some form of guidance prior to taking commitments.

The implemented DSS is particularly useful for novice designers that have limited knowledge and expertise and who are not used to the traditional way of computer aided modeling. In the case of expert designers, it is helpful in that it encourages designers to experiment and use new alternatives rather than sticking to the usual commitments for a safer result.

7.0 Claims

What is claimed is:

Claim 1: A computational framework and its underlying algorithm comprising of four frames (Frame 1 to Frame 4) which collectively support users by informing them of the feasible solution space from product, sub-assembly and component specifications prior to taking any solution commitments.

Claim 2: Frame 1 according to Claim 1, further allowing the user to input product life-cycle specifications which could be or could not be inherited by the sub-assembly, sub-assembly specific life-cycle specifications which could be or could not be inherited by the component and component specific life-cycle specifications.

Claim 3: Frame 1 according to Claim 1, further allowing the user to determine whether each specification is obligatory or optional.

Claim 4: Frame 2 according to Claim 1, further mapping the specifications (depending on whether they are obligatory or optional) to the properties and values of the alternatives and classifies, by means of a tag, the alternatives in such a way that the user is informed about the 'feasible', 'non- feasible' and 'non-recommended but still acceptable' solution space.

Claim 5: Frame 2 according to Claims 1 and 4, further provided that the mapping occurs between the decision taken and the properties and values of the alternatives, to classify the alternatives for the next decision.

Claim 6: Frame 2 according to Claim 1 which also allows the user to view:
- brief guidance informing the user which specification will be violated if the alternative is chosen or which previous commitment will be in conflict with the alternative if it is chosen

- detailed guidance informing the user about the life-cycle consequences and giving recommendations on how to minimize the consequences
prior to selecting the alternative.

Claim 7: Frame 3 according to Claim 1 which generates three models in real time:
- the specification model which evolves as the designer takes specification commitments in Frame 1;
- the solution model which is made up of:
- a component model that evolves as the designer chooses alternative elements related to the component- a life-phase system model that evolves as the designer chooses alternative elements related to the life-phases of the component

Claim 8: Frame 4 according to Claim 1 which generates a geometric model corresponding to the alternative elements chosen.

8.0 Figures

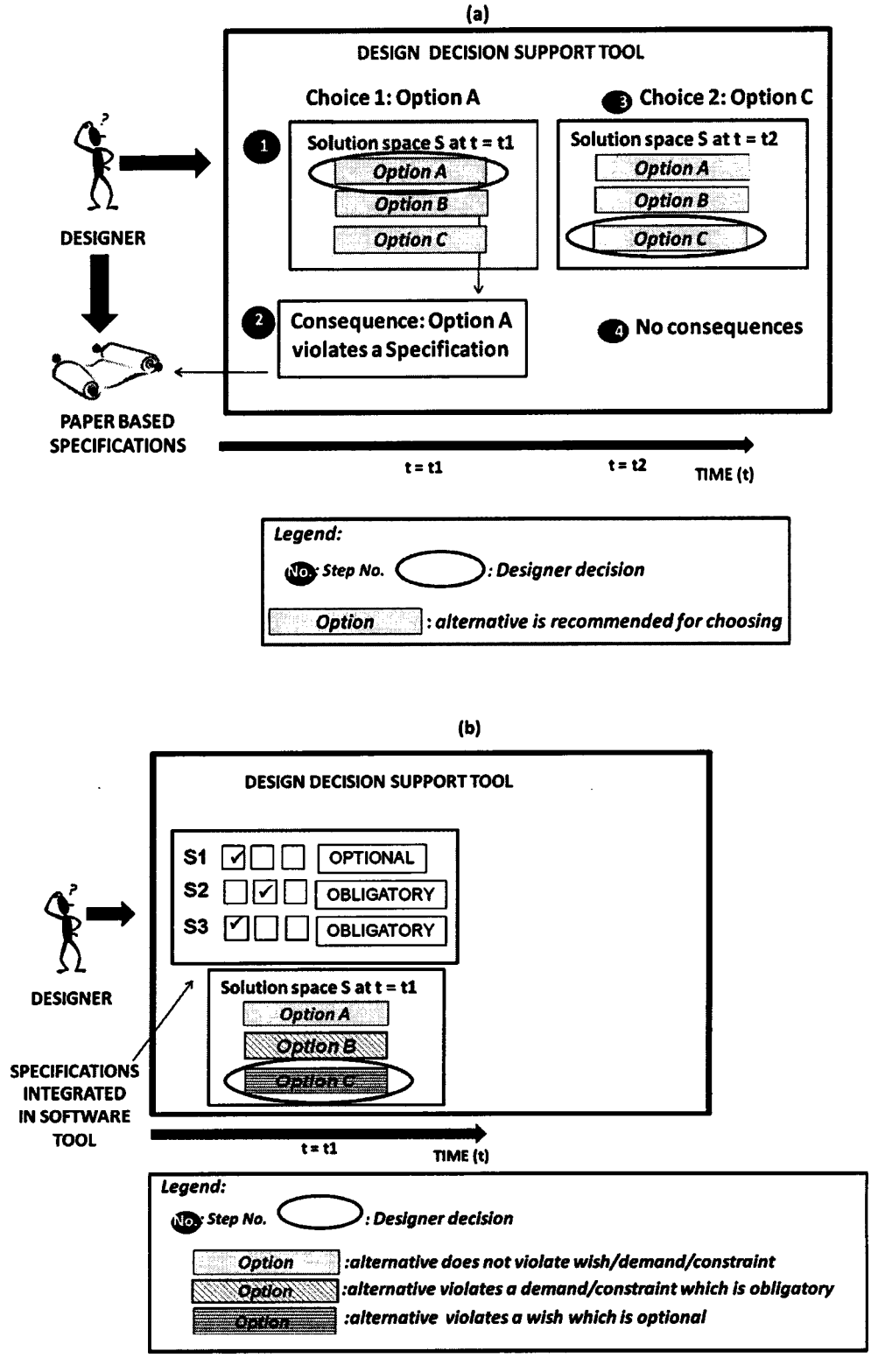
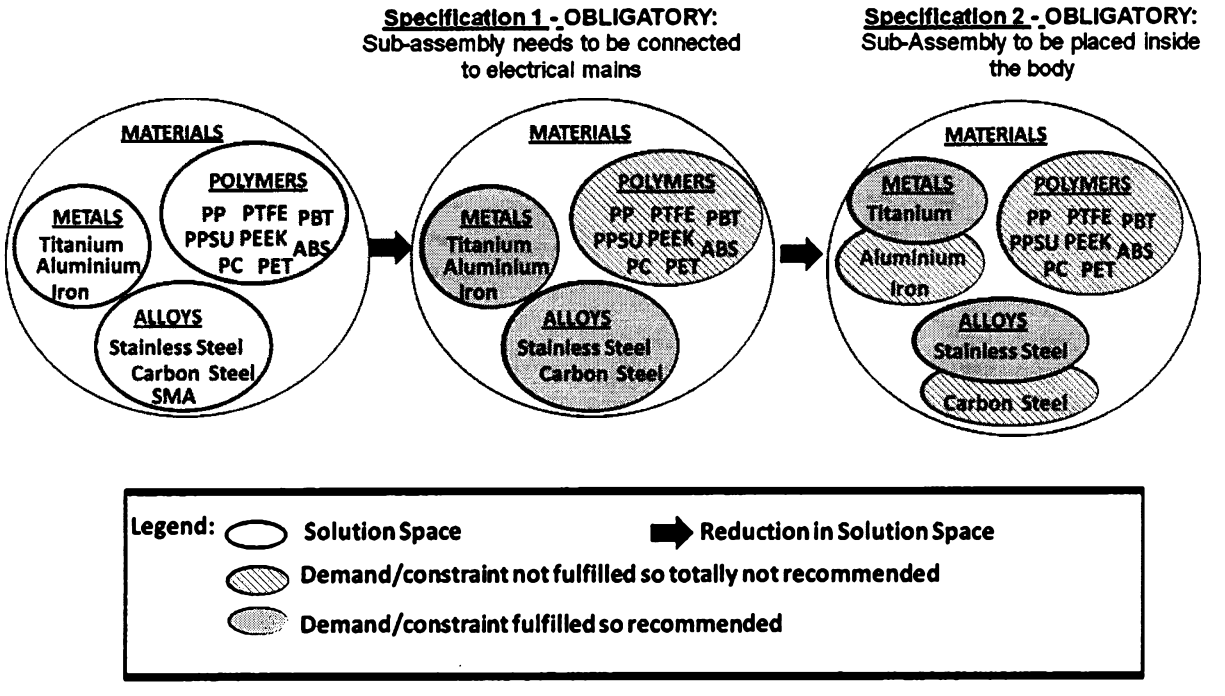


Figure 1. (a) Idea behind Approach 3 (b) Idea behind Approach 4



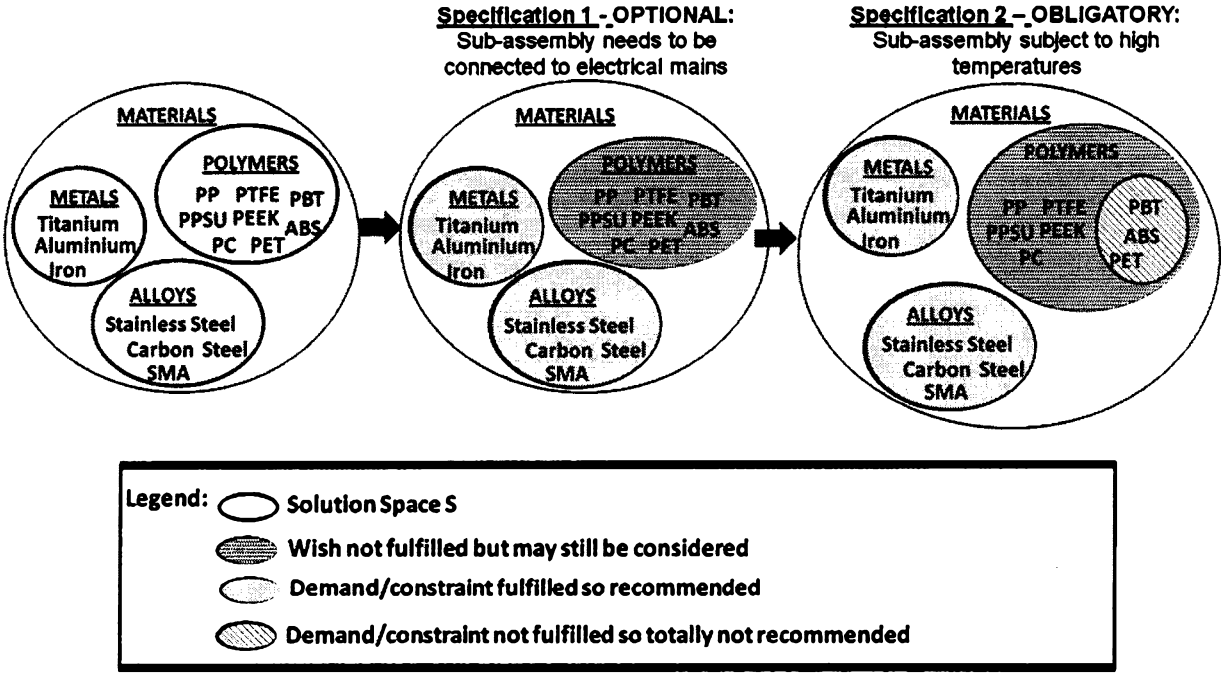


Figure 2: Reduction in design solution space by Approach 4

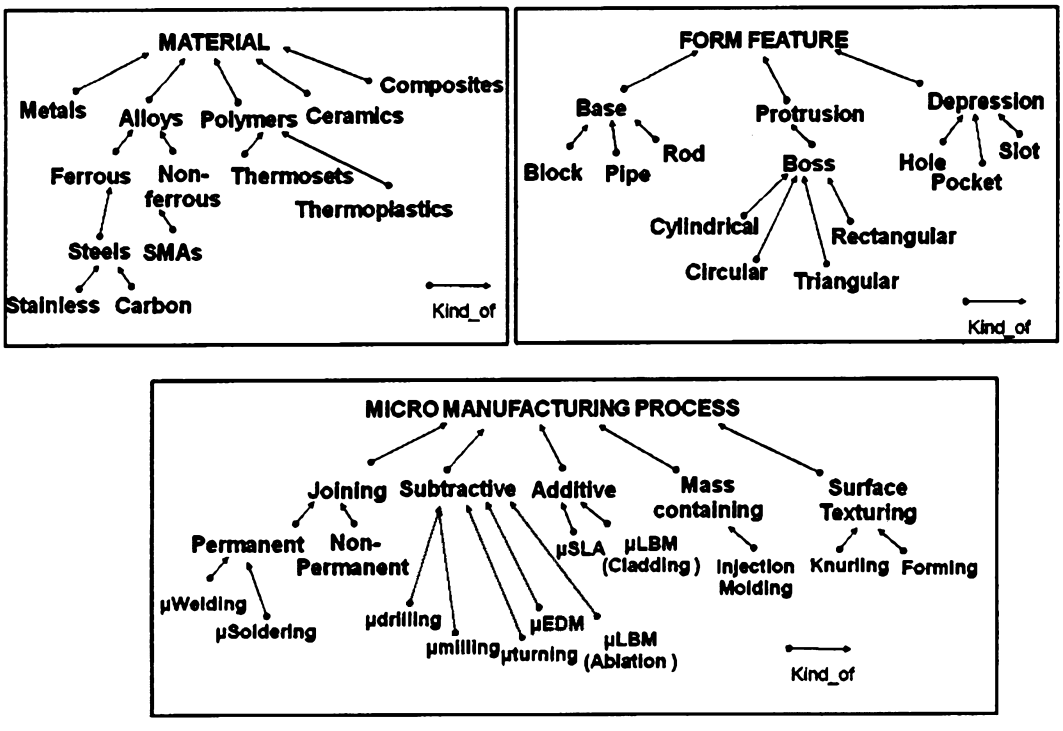


Figure 3. Parts of kind_of taxonomies

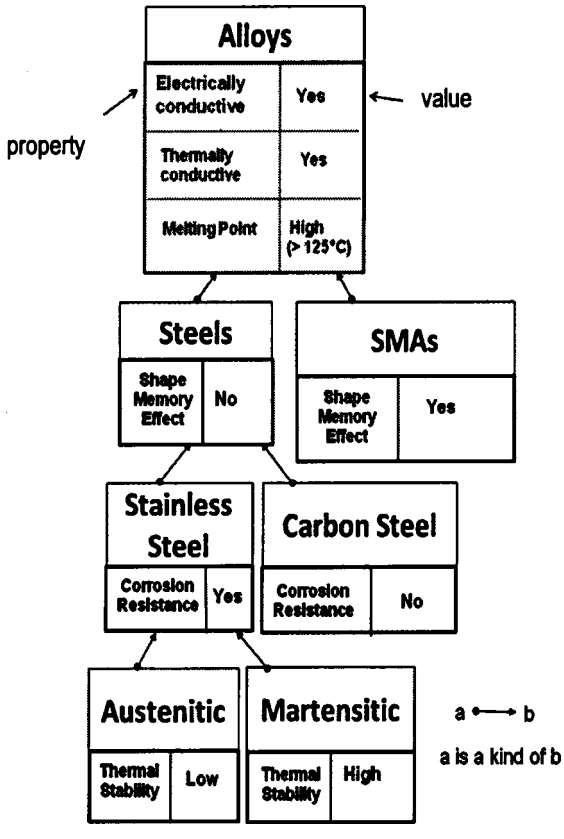


Figure 4. Examples of Frames

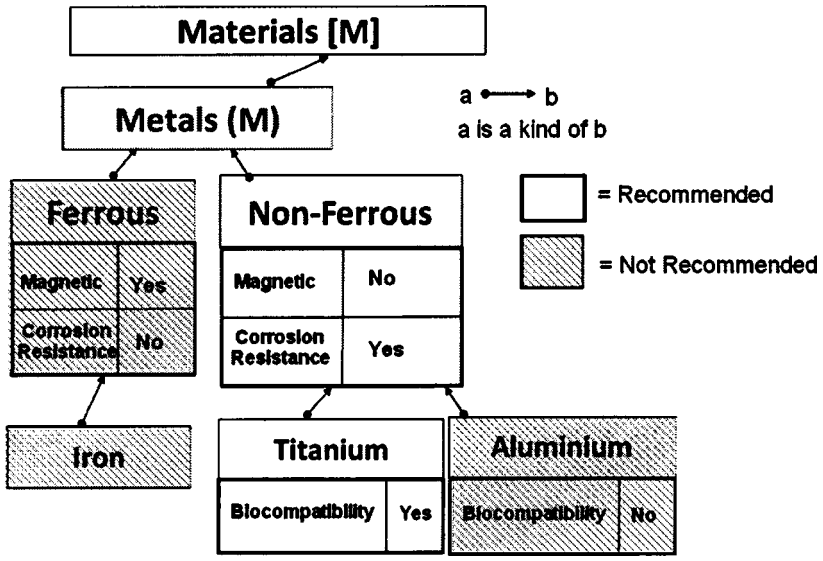


Figure 5. Specification Filter applied due to a match with the Specification 'Inside the Body'

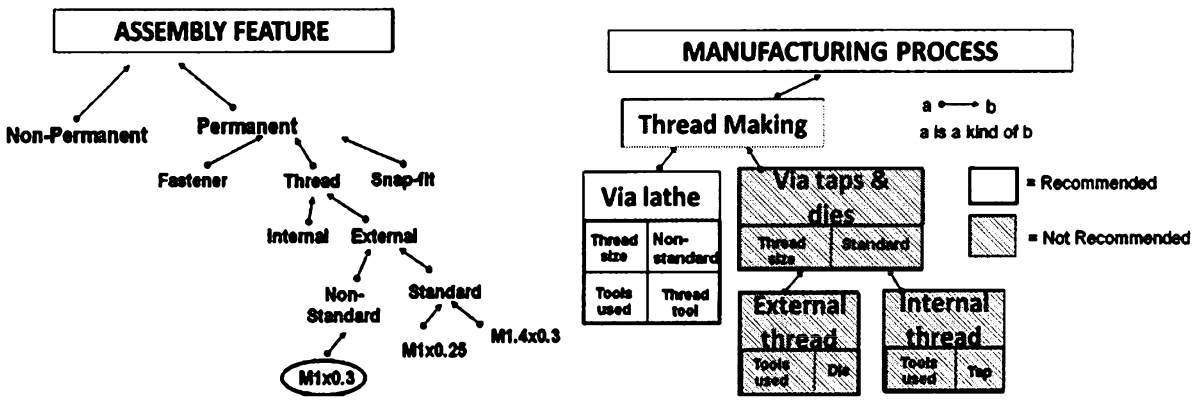


Figure 6. Commitment Filter applied to a match of the Commitment taken 'Assembly feature = M1x0.3 thread'

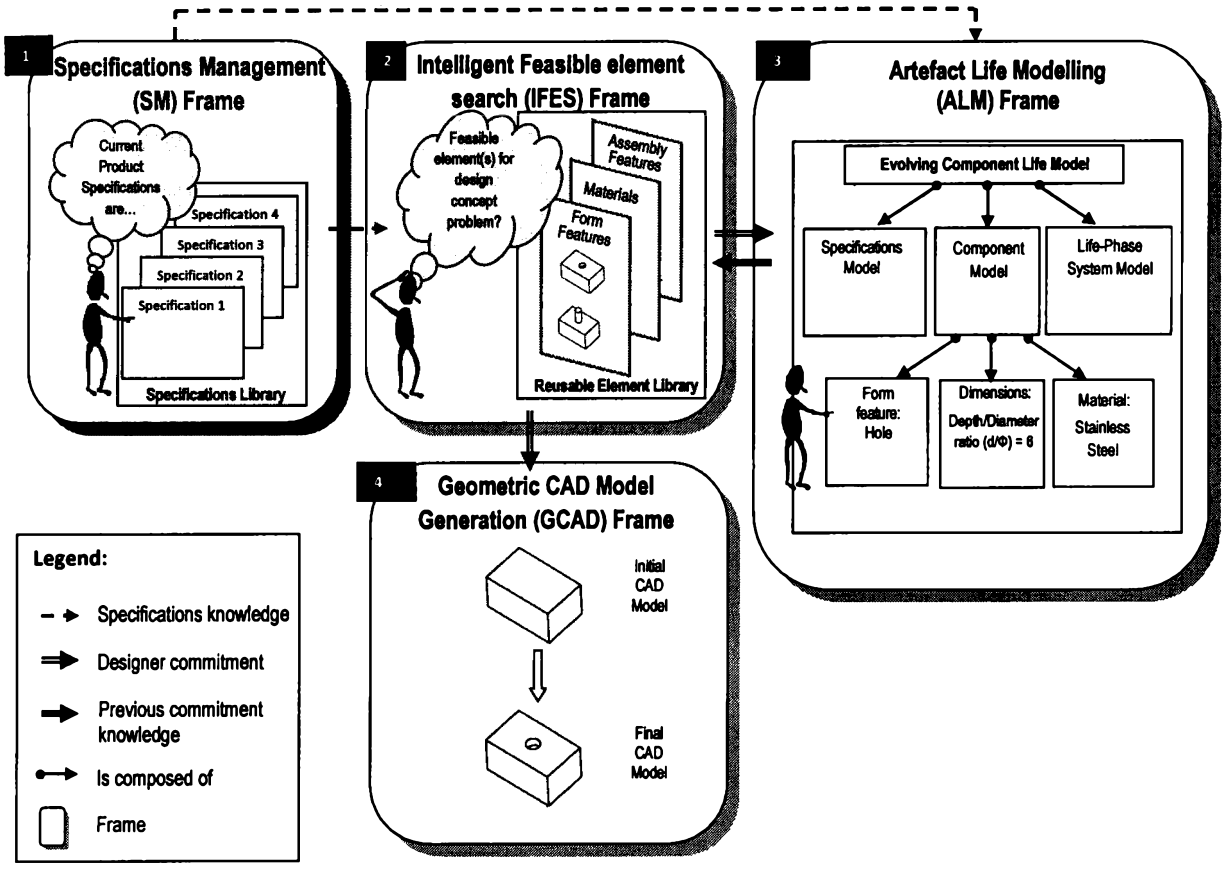


Figure 7. ICT Tool Framework

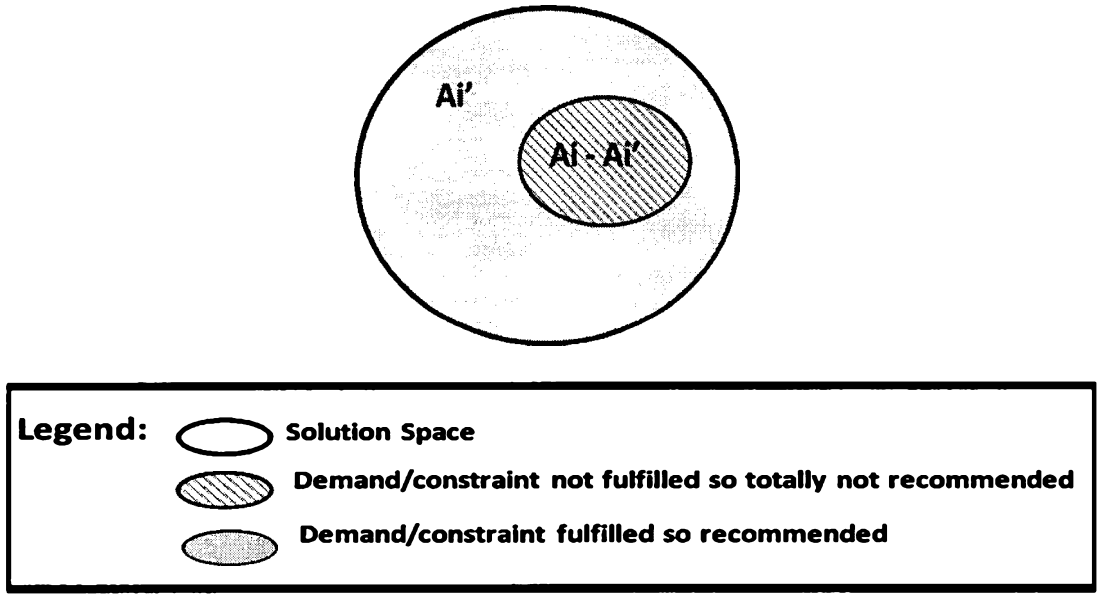


Figure 8. Solution Space restriction via specification filter in the case of obligatory specifications

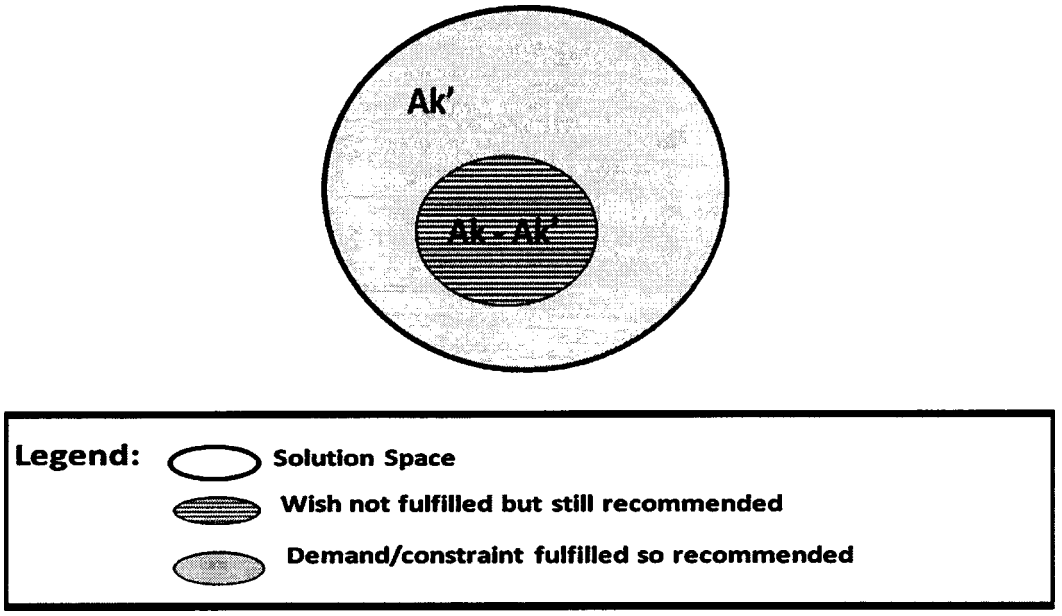


Figure 9. Solution Space restriction via specification filter in the case of optional specifications

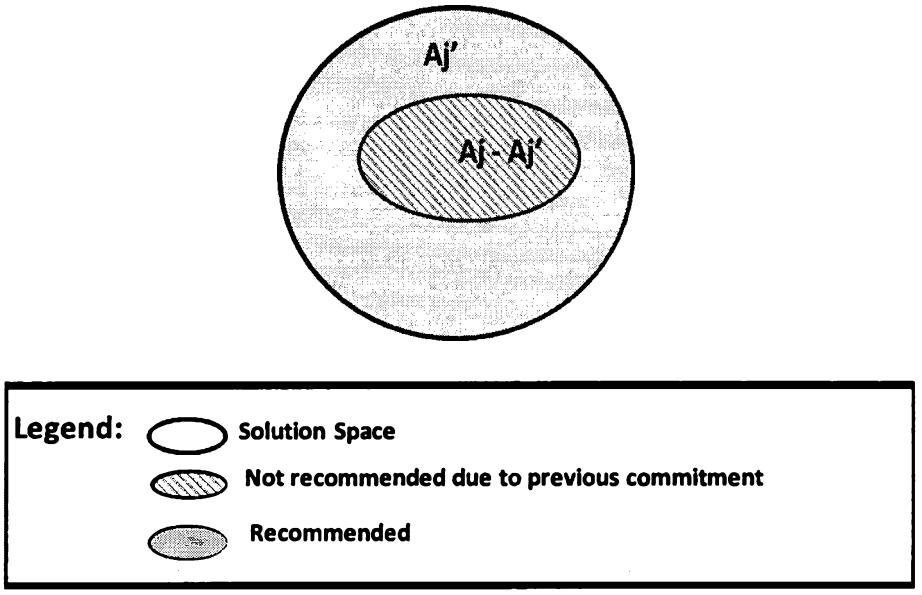


Figure 10. Solution Space restriction via commitment filter in the case of already taken decisions

9.0 References

Borg, J.C., 1999. Design Synthesis for Multi-X – A 'Life-Cycle Consequence Knowledge Approach'. Ph.D. University of Strathclyde.

Dai, X., Juster, N.P. and Qin, Y., 2006. The Development of a Design Tool for Micro Manufacture, *Proceedings of the ESAFORM Conference*, pp. 91-93.

Dooley R.L., Heimke, G., Dingankar, A., Berg, E. and Kimbrough, E., 1988. Automated design and analysis system for design of custom orthopedic implants, *Proceedings of the 1st international conference on Industrial and engineering applications of artificial intelligence and expert systems*, pp. 405-412.

Gayretli, A. and Abdalla, H.S. 1998. A knowledge-based system for manufacturing process optimization, *Proceedings of 2nd International Synposium Tools and Methods for Concurrent Engineering (TMCE '98)*, pp. 419 - 427.

Gorji-Bandpy, M., Soleimani, S. and Jafari, B, 2008. Numerical analysis of fluid flow in a duct around perpendicular cylinder with triangle cross section obstacle, *International Journal of Dynamics of Fluids*, vol. 4, no. 2, pp. 93–107.

Lockett, H. and Guenov, M., 2007. A Knowledge Based Expert System for Moulded Part Design, *Proceedings of the International Conference on Engineering Design (ICED 07)*.

Nijhuis, W., 1984. Ergnomics, one of the disciplines in product development. In: Contemporary Ergonomics, *Proceedings of the Ergonomic Society's Conference, E.D.*, Megaw (ed.), London.

Pahl and Beitz, Engineering Design: A Systematic Approach, 1988, Design Council.

Pugh, S. and Ion, W.J. 1988. Total Design: The Systematic Activity necessary from the Identification of the Market/User Need to the Selling of the Successful Product to Satisfy the Need, Design Division, University of Strathclyde.

Rehman, F. and Yan, X-T., 2007. Evaluation of a Context Knowledge Based Tool to Support Decision Making in Conceptual Design, *Proceedings of the International Conference on Engineering Design*, Paris, France, pp. 188-120.

Roozenburg, N.F.M. and Eekels, J., Product Design: Fundamentals and Methods, 1995, John Wiley & Sons.

Sam Lazaro, A., Engquist, D.T. and Edwards, D.B. An Intelligent Design for Manufacturability System for Sheet-metal Parts, *Concurrent Engineering: Research and Applications*, vol. 1, pp. 117-123, 1993.

Tang, M.X., 1997. A knowledge-based architecture for intelligent design support, *The Knowledge Engineering Review*, vol. 12, no. 4, pp. 387-406.

Wang, L.B. and Tao,W.Q., 1997. Numerical analysis on heat transfer and fluid flow for arrays of non-uniform plate length aligned at angles to the flow direction, *International Journal of Numerical Methods for Heat & Fluid Flow*, vol. 7, no. 5, pp. 479-496.

Wood, R. M. and Bauer, S.X.S, 1998. A Discussion of Knowledge Based Design, in *7th AIAA/UUSAF/NASA/ISSMO Symposium on Multidiscipliary Analysis and Optimization,* Saint Louis, MO, United States.

1.2 Glossary

TERM	EXPLANATION
Alternative	A PDE or LCPE within a solution space that the designer can choose from
Commitment filter	A filter which matches the property and value of the alternatives to an already taken decision commitment and classifies the alternatives
Constraint	Restriction placed on the solution to achieve the requirement that must be met under all circumstances
Customer Requirement	'wants' of the customer which could either be a wish or a demand
Commitment	A decision that is taken by the designer when selecting one alternative from the solution space
Demand	Customer requirement that must be met under all circumstances
Engineering Characteristic	'hows' to achieve the 'wants'
LCC	A consequence on the life-cycle of the product which is caused by a design commitment
LCPE	System related to life-cycle phase such as machine or a tool/cutter in the manufacturing phase, an autoclave from the cleaning phase, an incinerator as part of the disposal phase
Life-Cycle	The consecutive and interlinked stages of a product system, from raw material acquisition or generation of natural resources to end of life
PDE	Component element such as form feature, assembly feature, surface texture, material, dimension
Property	Property of the alternative e.g. electrical conductivity of material, feasible product quantities that can be manufactured economically by a manufacturing process
Solution Space	The whole set of PDEs and LCPEs
Specification	List of wishes, demands and constraints
Specification filter	A filter which matches the property and value of the alternatives to a specification and classifies the alternatives
Value	Value of the property e.g. $63.0 \times 10^6\ S^{m-1}$, 10,000 parts
Wish	Customer requirement that should be taken into consideration whenever possible

1.3 Notation

SYMBOL	EXPLANATION
S	Solution Space
A_i	Alternative *i* in *S*
$(A_m)_i$	Alternative *i* from menu currently being loaded
$((A_m)_i)_{tag}$	Tag of Alternative *i* from menu currently being loaded
$((A_m)_i)_{LCCb}$	A brief description of a LCC which would result if alternative $(A_m)_i$ is chosen
$((A_m)_i)_{LCCd}$	The link to a detailed description of a LCC, including recommendations on how the LCC can be avoided, which would result if alternative $(A_m)_i$ is chosen
D_x	Document (which may be paper-based or computer-based) and which contains the current non-recommended alternatives. x may stand for: *assembly_features_ob* – document which contains the assembly features which currently violate an obligatory specification or are non-recommended due to a previous commitment *form_features_ob* – document which contains the form features which currently violate an obligatory specification or are non-recommended due to a previous commitment *manu_processes_ob* – document which contains the manufacturing processes which currently violate an obligatory specification or are non-recommended due to a previous commitment *materials_ob* – document which contains the materials which currently violate an obligatory specification or are non-recommended due to a previous commitment *assembly_features_op* – document which contains the assembly features which currently violate an optional specification *form_features_op* – document which contains the form features which currently violate an optional specification *manu_processes_op* – document which contains the manufacturing processes which currently violate an optional specification *materials_op* – document which contains the materials which currently violate an optional specification
$(A_d)_i$	Alternative *i* from currently open document (one from the eight D_x)
$(E_d)_i$	Reason E corresponding to $(A_d)_i$ (that is, to alternative *i* in currently open document) (Note: this will be displayed when LCCb is required)
$(G_d)_i$	Guideline G corresponding to $(E_d)_i$ (Note: this will be displayed when LCCd is required)

n	Total number of alternatives in solution space S
m	Total number of properties for a given alternative A_i
p	Total number of chosen specifications and previous commitments which are limiting the solution space S
q	Total number of alternatives in the menu currently being loaded in the user interface
r	Total number of non-recommended alternatives in currently open document (one from the eight D_x)
S_{ob}	An obligatory specification (demand/constraint)
S_{op}	An optional specification (wish)
d_i	A previous decision commitment
p	Property
p_v	Particular value of the property p
Ep_v	Reason E corresponding to why a particular p_v is making A_i non-recommended
=>	implies
$\in$	is a member of
$\exists$	there exists
$A \subseteq B$	A is a subset of B (part_of)
$A \cup B$	either A or B (union)
$A \cap B$	A and B (intersection)
$A \mid B$	A given B

2.0 Field of invention

Customers are requesting complex products that are smaller in size, multifunctional and including other disciplines. Nowadays, it is also common practice to adopt a concurrent engineering design approach whereby products are being designed for the life-cycle and thus the life-cycle phases need to be kept in mind during the early stages of design. Thus design has become a very knowledge-intensive activity and too complex to handle on an individual basis. As a result, designers need to be supported when taking decisions, especially since it is at this point that they spend a lot of time alone in the design office. There exists a number of computer aided DSSs that aid designers in their work. A good number of them (such as SMAART (Sam Lazaro et al 1993), DAS (Dai et al 2006) and Molding Advisor (Lockett and Guenov 2007)) make use of *feature recognition* to link the geometric model to a knowledge based system. In these type of DSSs, the designer is free to take any decision related to component geometry (that may be 'good' or 'bad') and draws the component in a computer aided geometric modeling (CAGM) software. Then via feature recognition, the geometric features are extracted from the computer aided drawing and sent in the form of a feature file to the knowledge-base which has a number of guidelines stored and which via an advisory window informs the designer of any negative life-cycle consequences (LCCs) that may arise due to wrong decisions

that were taken. The drawback with this type of DSSs is that advice is given to the designer *AFTER* decisions would have been taken. *Process planning* support systems (such as PROACTIVE DFM (Gayretli and Abdallah, 1998)) are used to guide the designer on the sequence of manufacturing processes that need to take place, following the use of feature recognition where manufacturing features are extracted from the model, but once again, guidance is given too late in the design process. Tools that make use of *numerical analysis* (such as ORTHOPERT (Dooley et al, 1988), and the systems developed by Wang and Tao (1997) and Gorji-Bandpy et al. (2008)) to inform the designer about the behavioural properties of the product (without having the need to manufacture a physical prototype) such as stresses and strains when forces are applied to it, also provide guidance too late – in the solution evaluation activity of design. In other DSSs, that guide the designer at earlier activities in design (when the designer is still deciding on solutions) such as FORESEE (Borg 1999), the designer chooses from a set of predefined features and based on the selection and is informed of the LCCs, but once again *AFTER* having taken the design commitments. Therefore the designer would have to retract the commitment and choose another alternative until the consequences are eliminated. The problem with these systems is that such approaches result in a lot of time that is wasted due to much repetitive iteration. Hence, there is the need for a different type of DSS - that guides the designer and forwards advice regarding possible upcoming consequences *BEFORE* commitments are taken.

In design, the designer moves from a need (the design problem) that is depicted by customer 'wants' to a solution with the aim of satisfying the need. The 'wants' of the customer are converted into technical specifications via tools such as the Quality Function Deployment (QFD). The designer needs to keep these specifications in mind, while designing, so as to end up with a product that satisfies the customer needs. But many times, designers refer back to the QFD, when they are well into the design. It is only after having taken a lot of commitments that they realize that they have forgotten one or more of the specifications and thus this results in high changing effort and wastage of time to redesign. Some DSSs have been developed in such a way so as to integrate requirements into the system. Examples include PROACTIVE DFM (Gayretli and Abdallah, 1998), DAS (Dai et al, 1996) and the system described in the patent by Sebastian et al (2000). However these DSSs focus on only one life-phase - the manufacturing phase by allowing the designer to input the constrained manufacturing time or cost or product quantity at the start of the design. However, the product undergoes more phases throughout its life apart from the manufacturing phase and these too need to be considered.

There is thus a need for a DSS that, informs the designer of the feasible solution space from the design specifications – that is, which makes a distinction between those alternatives that are recommended and those that are not and which result in LCCs – *BEFORE* commitments are taken. This algorithm should be capable of linking the life-cycle oriented specifications to the feasible solution space.

3.0 Background of invention

Customers have a number of 'wants' which they require for a product. These requirements are of two types: *wishes* and *demands*. Demands are requirements that need to be fulfilled by the

solution whereas wishes are ideally fulfilled but the solution is still valid if they are not. *Constraints* on the other hand are restrictions that are placed on the solutions to fulfill the requirements usually from other life-cycle stakeholders. So demands and constraints are both 'obligatory' whereas wishes are 'optional'. These are recorded in a Product Design Specification (PDS) which can take various forms. For example, Pahl and Beitz (1988), Roozenburg and Eekels (1995) and Pugh and Ion (1988) represent it in the form of a checklist with headings that relate to the product e.g. limited weight and cost, aesthetics, ergonomics. Nijhius (1984) describe a different type of PDS known as a process tree that is life-cycle oriented rather than product oriented. To end up with a life-oriented solution, one needs to consider all the product life-phases and for this reason, the invention adopts the second type of PDS so as to include life-cycle specifications. It is also important that the specifications are identified as being either obligatory or optional. Every product has a breakdown structure consisting of the components that are assembled together to form sub-assemblies that are assembled together to form the product. The *Product Life-cycle Specifications* (PLS) may be different to the *Sub-assembly Life-cycle Specifications* (SLS) due to the different environments that the sub-assemblies may be found in. For example, if the product is submerged in sea water, it is important that the outer casing is made from a material that can withstand a salty environment. However the sub-assemblies that are found enclosed within the casing that are not subject to the same environment, need not be made from the same material. Similarly the Component Life-cycle Specifications (CLS) may be different from the sub-assembly specifications.

Engineering design is essentially a *decision making* process with the designers being the 'decision-makers' taking a number of decision commitments that need to satisfy the customers' wants or constraints. This process starts off with a design *issue* (the 'need' to design or redesign a product) which is constrained by a number of *criteria* (specifications) which limit the available *design solution space*. Data, information and knowledge support the criteria and give rise to a number of *alternatives* (solution options to address the problem). The designer then needs to take a number of *decisions* related to the component form features, material, dimensions, surface texture, assembly features and manufacturing and assembly processes to be used to fabricate the product) by selecting from the alternatives. Decisions are dynamic; they may later be changed as criteria and preferences change, and as new alternatives are generated. With time, as the designer and customer communicate with each other and the design develops, the specifications become more concrete and detailed.

3.1 Incorporation of specifications in the design process

From literature it was observed that there exist four different approaches as to how design specifications can be incorporated in the design process. A description of these four approaches is detailed below:

Approach 1- Using the traditional "paper-based" method:

Using this method, the designer has to go back and manually refer to the design criteria whenever the need arises. As mentioned in Section 2, this is generally done much later during the design, and therefore high changing efforts will be required if some of the criteria are not met.

Approach 2 - Using autonomous design tools:
Autonomous design tools produce designs automatically after given the design specifications. Although at first glance these tools may seem more supportive during the design, this is generally not the case. In fact many design researchers agree that while computers are an essential tool in engineering design, it is a critical mistake to view them as the heart and soul of design as this would result in less creativity and innovation (Wood and Bauer 1998). Therefore, a better approach would be to go between the "paper-based" approach and the autonomous design tool approach – that is, to use intelligent design support tools.

Approach 3 - Using intelligent design support tools that provide support *after* a design decision is taken:
In this approach, the designer, after taking a "bad" decision (a decision that violates one or more design specifications), as in Figure 1 (a) Step 1, is informed that the decision taken has certain undesired consequences on the other product life-cycle phases (Figure 1 (a) Step 2) and therefore the designer has to choose another option (Figure 1 (a) Step 3) until no consequences result (Figure 1 (a) Step 4). This approach is a big improvement with respect to the first two approaches as while in the first one a lot of redesigns tend to be needed until all the specifications are met, in the second one, design creativity is limited as the design is produced automatically. However, as illustrated in Figure 1 (a) this approach may still require a number of iterations (which result in loss of design time) until a solution which has minimal undesired consequences is found.

Approach 4 - Using intelligent design support tools that provide support *before* a design decision is taken:
In this approach, which is the one being adopted by the invention being proposed (illustrated in Figure 1(b)), the designer is made aware of alternatives that i) should not be selected as they violate an obligatory specification (demand/constraint) ii) can be selected as they do not violate any specification (demand/constraint or wish) iii) can be selected even though they go against the customer's wish (optional specification), before actually taking the decision.

As illustrated in Figure 2, depending on whether the specification is obligatory or optional, the solution space is reduced. As can be seen in the case of demands/constraints the solution space is reduced much faster than in the case of wishes. Therefore, compared to the previous approach, where several iterations may be required until the ideal solution is found, in this case no iterations are required as the designer is made aware of the "bad" decisions from an even earlier stage - *before* the design decision is actually taken. Figure 1(b) shows the reduction in design time compared to the previous approach.

3.2 Displaying recommended and non-recommended alternatives to the designer

When providing support to designers regarding alternatives (approach 4 in Section 3.1), it is not sufficient to present only the recommended and non-recommended alternatives. Designers

generally need to know the motivation behind the guidance given together with recommendations on how to reduce or eliminate any resulting life-cycle consequences. Therefore, apart from being aware of any life-cycle consequences, designers will be motivated to explore different alternatives which have no or less consequences on the product life-cycle. This means that designers require a degree of detail from the tool to be motivated to employ it in their daily work.

On the other hand, if the tool provides excessive detail and annoys the designer with too much guidance, then designers, especially experienced ones, may refrain from using it. Therefore, a balance should be found between *detail* and *fast access* of the required information. To meet this need, two ways of presenting LCCs to designers are therefore suggested:

i) LCCb – which provides a brief description of LCCs
ii) LCCd – which provides a detailed description of LCCs including recommendations on how the LCCs can be avoided

4.0 The invention

The invention concerns the process of making the designer aware of the 'feasible', 'non-feasible' and 'not recommended but still acceptable' solution space from the product, sub-assembly and component specifications.

Sub-assemblies may inherit specifications from the product or else be specific to the sub-assembly. Similarly components may inherit specifications from the sub-assembly or else be specific to the component. Thus the product, sub-assembly and component specifications need to be defined separately.
Once the specifications are defined, two processes occur - mapping and classification, using two types of filters:

i) the *specification filter* – that maps the specification (whether product/sub-assembly/component)with the properties and property values of the alternatives and classifies the alternatives depending on whether the specification was obligatory or optional.
The filter distinguishes between the alternatives as follows:
a) those alternatives that fulfill the specification (whether it is obligatory or optional) - feasible;
b) those alternatives that do not fulfill an obligatory specification – not feasible;
c) those alternatives that do not fulfill an optional specification but are still recommended as possible options which the designer can choose from.

ii) the *commitment filter* – that maps a commitment that has already been taken by the designer with the properties and property values of the alternatives for the next commitment, and classifies the alternatives as feasible or not feasible.

The filter makes use of a technique known as *pattern matching* whereby it checks for a match between a specification or a commitment and the properties and values of alternatives as explained in the next sections. If there is a match between these two then the alternative is

considered feasible (and the designer is informed about the recommended alternative) and if not, then the alternative is not considered feasible (and the designer is informed that the alternative is not recommended) and guidance is given via LCCb and/or LCCd. Whereas in PROCONDES (Rehman and Yan 2007) a mapping is made between the function and the PDE, this concept makes a mapping between the life-cycle specification or previous commitment and the solution space of PDEs and LCPEs.

The alternative options of PDEs and LCPEs are stored in libraries that are structured in *kind_of taxonomies.* These taxonomies are hierarchies of the alternative options and their parents (classes) and children (sub-classes). Each child of the taxonomy is a 'kind_of' of the parent (which is found at a level above it). Each child automatically inherits the properties of its parent by a technique known as *inheritance*. An example of the material hierarchy is shown in Figure 3. Other hierarchies are those for the other properties that a designer can manipulate such as form features, assembly features and surface textures. Each alternative option within the hierarchy is stored in a *frame* together with its properties and values as shown in Figure 4.

Examples of the two filters are given below:

Example i) utilizing a *specification filter:*

As demonstrated in Figure 2:

If SLS = Obligatory = Sub-assembly needs to be placed inside the body

Filter distinguishes between feasible and non-feasible material alternatives as follows:

Biocompatible and anti-corrosive materials = feasible
(i.e. those materials that have their values of the property values 'yes' for the biocompatibility property and corrosion resistance property)

Materials that are non-biocompatible and/or corrosive = not feasible as they will result in the body rejecting the material or poisoning the body (i.e. those materials that have their values of the property values 'no' for either the biocompatibility property and/or the corrosion resistance property). This is explained in Figure 5.

If SLS = Optional, then all materials can be chosen, whether biocompatible or not, however the filter still distinguishes between those materials that are recommended because they fulfill the wish and those that may still be chosen even though they do not fulfill the wish.

Example ii) utilizing the *commitment filter:*

If already taken commitments include:
Assembly feature = external thread
and
Size = non-standard

Then some manufacturing processes are feasible and others are not

Thread making via lathe = feasible

Thread making via taps and dies = not feasible as taps and dies come in standard sizes and can only be used to make threads of a standard size. This is explained in Figure 6.

4.1 The filtering algorithm

The filtering algorithm which is used to distinguish between feasible and non-feasible solution spaces will be described in this section. The algorithm is made up of two main parts – the first part is needed to perform the *mapping* from the specification space, or from previous commitments made, to the design solution space and the second part is required to *classify* the solution space in recommended and non-recommended alternatives to the designer. Also, since the motivation for a particular alternative being non-recommended is generally required, as discussed in section 3.2, the classifier also aims to display these motivations to the designer using the different ways mentioned. Therefore, the designer is proactively supported to choose a feasible alternative *before* actually taking a decision.

For the notation description of the filtering algorithm, refer to the beginning of this document.

The first part of the filtering algorithm is described below:

Part 1: Mapping of specification space (or previous commitments) to solution space

(1) For i = 1 to p
(2) For j = 1 to n
(3) For k = 1 to m
(4) If $((p_v)_k \in A_j)$ AND $((S_{ob}$ *OR* $d_i) \Rightarrow$ NOT $((p_v)_k))$ then
(5) If ($A_j \in$ assembly features) then write (A_j AND $(E(p_v)_k)$ in $F_{assembly\_features\_ob}$ End If
(6) If ($A_j \in$ form features) then write (A_j AND $(E(p_v)_k)$ in $F_{form\_features\_ob}$ End If
(7) If ($A_j \in$ manufacturing processes) then write (A_j AND $(E(p_v)_k)$ in $F_{manu\_processes\_ob}$ End If
(8) If ($A_j \in$ materials) then write (A_j AND $(E(p_v)_k)$ in $F_{materials\_ob}$ End If
(9) End If
(10) If $((p_v)_k \in A_j)$ AND $(S_{op} \Rightarrow$ NOT $((p_v)_k))$ then
(11) If ($A_j \in$ assembly features) then write (A_j AND $(E(p_v)_k)$ in $F_{assembly\_features\_op}$ End If
(12) If ($A_j \in$ form features) then write (A_j AND $(E(p_v)_k)$ in $F_{form\_features\_op}$ End If
(13) If ($A_j \in$ manufacturing processes) then write (A_j AND $(E(p_v)_k)$ in $F_{manu\_processes\_op}$ End If
(14) If ($A_j \in$ materials) then write (A_j AND $(E(p_v)_k)$ in $F_{materials\_op}$ End If
(15) End If
(16) Next k
(17) Next j
(18) Next i

The following is a description of what is happening in Part 1 above of the algorithm. Given a chosen specification or previous commitment (i), the algorithm traverses each property (k) of each alternative available (j) in the solution space *(lines (1) – (3))*. Then *(line (4))*, if a chosen obligatory specification or previous commitment *(S_{ob} OR d_i)* implies that a particular value for a given property is not allowed (NOT $((p_v)_k)$) and this same property with the respective value is a member of the current alternative $((p_v)_k \in A_j)$, then the current alternative together with the corresponding reason why this alternative is not recommended are written in the appropriate document *(lines (5) – (9))*. This procedure is repeated for optional specifications, where the current alternative together with the corresponding reason why an optional specification is violated are also written in the appropriate document *(lines (10) – (15))*. This procedure is repeated for every alternative available *(line (17))* for each chosen specification or previous commitment *(line (18))*.

To avoid having to write a lot of iterations, which make programming quite complex and which also reduce the processing efficiency of the program, a declarative programming paradigm is recommended. In this paradigm, as opposed to the procedural programming style adopted by the imperative programming paradigm, *what* the program should do is described rather than *how* it should do a particular task. Therefore the program only needs to be provided with the data and then it performs the necessary pattern matching internally to arrive at the same end-state of the above algorithm.

After the above algorithm is performed, the final output is a collection of documents (for assembly features, form features, manufacturing processes and materials) which have the non-recommended alternatives, together with the reason for each, resulting from the currently chosen specifications (obligatory and optional) and commitments. These documents are required for the second part of the algorithm, to classify the solution space, which will be described next.

Part 2: Classification of solution space

(1) For j = 1 to q
(2) If $((A_m)_j \in$ assembly features) then $D_x = D_{assembly\_features\_ob}$ End If
(3) If $((A_m)_j \in$ form features) then $D_x = D_{form\_features\_ob}$ End If
(4) If $((A_m)_j \in$ manufacturing processes) then $D_x = D_{manu\_processes\_ob}$ End If
(5) If $((A_m)_j \in$ materials) then $D_x = D_{materials\_ob}$ End If
(6) Open D_x
(7) Matchfound = False
(8) MenuLoadedAgain = True
(9) For k = 1 to r
(10) If $((A_m)_j = (A_d)_k$ Then
(11) $((A_m)_j)_{tag}$ = Red
(12) MatchFound = True
(13) End If
(14) If MatchFound = True Then

(15) If MenuLoadedAgain = True Then
(16) $((A_m)_j)_{LCCb} = (E_d)_k$
(17) Else
(18) $((A_m)_j)_{LCCb} = ((A_m)_j)_{LCCb} + (E_d)_k$
(19) End If
(20) If $\exists\ (G_d)_k$ Then
(21) $((A_m)_j)_{LCCd}$ = Visible
(22) End If
(23) MatchFound = False
(24) MenuLoadedAgain = False
(25) End If
(26) Next k
(27) Close D_x
(28) Next j

The following is a description of what is happening in Part 2 above of the algorithm. For each alternative (j) in the menu currently being loaded *(line (1))*, it is checked whether the alternative is an assembly feature, form feature, manufacturing process or a material and the corresponding document is opened *(lines (2) - (6))*. Initially, no matches have been found so variable Matchfound is set to False *(line (7))* while variable MenuLoadedAgain is set to True *(line (8))* to keep track that the menu has been loaded. Then, for each non-recommended alternative (k) in the currently open document *(line (9))*, the following 5 steps are performed:

- If alternative j from the menu currently being loaded matches alternative k from the currently open document, the tag of alternative j from the menu is set to red (which means that the alternative is absolutely not recommended as it violates an obligatory specification or is in conflict with a previous commitment). Furthermore, since a match has been found, variable Matchfound is set to True *(lines (10) - (13))*.
- If variable Matchfound is True and variable MenuLoadedAgain is also True, then LCCb of alternative j in the menu $((A_m)_j)_{LCCb}$ is set to Reason E corresponding to alternative k in the document $((E_d)_k)$ *(lines (14) - (16))*.
- If variable Matchfound is True and variable MenuLoadedAgain is False (for example in the case when more than one match is found for alternative j i.e. there are several reasons why alternative j is not recommended), then LCCb of alternative j $((A_m)_j)_{LCCb}$ is set to Reason E corresponding to alternative k in the document appended to the current LCCb $((A_m)_j)_{LCCb}$ *(lines (17) - (19))*.
- If variable Matchfound is True and there exists a Guideline G corresponding to reason $(E_d)_k$, then LCCd of Alternative j is set to visible so that the designer can view the corresponding guideline *(lines (20) - (22))*.
- Variable Matchfound and variable MenuLoadedAgain are both set to False so that they are ready for the next iteration of k *(lines (23) - (26))*.

The document is closed and the procedure is repeated for the next alternative (j) in the menu until no more alternatives are present *(lines (27) - (28))*.

The same algorithm is used to display alternatives that violate an optional specification but with the following modifications:

- Lines (2) – (5) are modified as follows so that the appropriate documents are opened:

  (2) If $((A_m)_j \in$ assembly features) then $D_x = D_{assembly\_features\_op}$ End If
  (3) If $((A_m)_j \in$ form features) then $D_x = D_{form\_features\_op}$ End If
  (4) If $((A_m)_j \in$ manufacturing processes) then $D_x = D_{manu\_processes\_op}$ End If
  (5) If $((A_m)_j \in$ materials) then $D_x = D_{materials\_op}$ End If

- Line (11) is modified as follows to classify the solution space. Note: Orange means that the alternative does not fulfill a wish but is still an option which the designer can choose:

  (11) $((A_m)_j)_{tag}$ = Orange

4.2 ICT Tool Framework

To implement the algorithm discussed in the previous section in an ICT tool, an approach framework is required to describe the generic functions which should be performed by the tool and how the knowledge present can be modeled and maintained. The framework, which is made up of four frames as illustrated in Figure 7 will be described next.

- *Specifications Management (SM) Frame:* In this frame the designer enters the product specifications that come out from the PDS into the design tool and determines whether they are *obligatory* (demands/constraints) or *optional* (wishes). Examples of product specifications include, for example that a product or one of its subassemblies or components should be durable, recyclable and corrosion resistant. Use quantities which will affect the manufacturing process to be used are also defined here. This frame should also allow the designer to change the specifications when required, as changing of specifications is quite common in many design situations. The specifications together with the associated knowledge (that is, restrictions on the design solution space) that come out from this frame are then transferred to the IFES Frame to display to the designer the recommended and non-recommended alternatives.

  What is advantageous in this approach framework is that if initially the customer is very vague in describing his/her wants but with time, the specifications are made more concrete, by this approach, the specification can be chosen at a later stage in design and the filter still can be applied. In the case of other approach frameworks such as that of PROCONDES (Rehman and Yan 2007) that use a numerical constraint-based approach, all the specifications need to be defined at the start as otherwise it would be incapable of forwarding results to the designer.

- *Intelligent Feasible Element Search (IFES) Frame*: In this frame, the designer, given a specific problem, refers to the Reusable Element Library, which consists of Product Design Elements (PDEs), such as form features, materials, assembly features, etc. and Life-Cycle Phase Elements (LCPEs) such as manufacturing and assembly processes and chooses the preferred element(s) for the problem. The term "Intelligent" is used here

because the alternatives are provided in an intelligent way to the designer. This means that although the designer can choose any element present, those alternatives that would have undesired consequences, given the product specifications set in the previous frame and previous commitments from the ALM Frame are tagged, to differentiate them from those alternatives that are recommended, and their corresponding LCCb and LCCd can be viewed.

- *Artefact Life Modelling (ALM) Frame*: Here, the library elements related to the component and its life phase system which are selected by the designer in the IFES Frame and the specifications which are selected in the SM Frame are built into an Evolving Component Life Model. These are processed by the tool so that any restrictions on the design solution space resulting from these elements are reflected immediately in the IFES Frame.
- *Geometric CAD Model Generation (GCAD) Frame*: While the ALM frame is generating models from the designer's choices and inferring restrictions on the solution space, the GCAD Frame updates the geometric model in the CAGM Interface drawing panel to correspond to the model in the ALM Frame. Updating of the geometric model includes, amongst others, the automatic creation of form features and the setting of materials and surface textures.

Here it should be noted that the idea of interfacing the design support tool with a geometric CAD model building frame already exists. It is included as part of the framework because of the fact that ideally a designer should not be forced to leave the "natural" work environment in order to acquire support. Rather, the designer should be motivated to employ the tool in the daily design work, by having it interfaced/embedded in a CAGM environment (such as Pro-E, Inventor, Solidworks, etc.). Therefore, what is claimed here is the idea behind this framework to provide support to the designer before taking a decision together with the algorithm to implement this idea in an ICT tool.

5.0 Formal Description

<u>For Case i) of Section 4.0 for the specification filter</u>

For a given life-cycle specification that is *obligatory*, the solution space of alternatives S changes to a restricted Ai' (the feasible alternatives) and Ai-Ai' (the non-feasible alternatives) as explained in Figure 8.

$$S \mid S_{ob} = A_i' \cap (A_i - A_i') \qquad \text{(eq 1)}$$

where Ai' and Ai-Ai' are subsets of S

$$A_i' \subseteq S$$
$$(A_i - A_i') \subseteq S \qquad \text{(eq 2)}$$

For a given life-cycle specification that is *optional*, Ak' are the recommended alternatives to fulfill wish and Ak-Ak' are the non- recommended alternatives that do not fulfill wish but still may be chosen by the designer.

$$S \mid S_{op} = A_k' \cap (A_k - A_k') \qquad \text{(eq 3)}$$

where Ak' and Ak-Ak' are subsets of S

$$A_k' \subseteq S$$
$$(A_k - A_k') \subseteq S \qquad \text{(eq 4)}$$

Figure 9 demonstrates this graphically.

<u>For case ii) of Section 4.0 for a commitment filter:</u>

For a given commitment that has already been taken by the designer, the solution space S for the next commitment may be restricted by Aj' (the feasible alternatives) and (Aj-Aj') (the non-feasible alternatives that result in interacting LCCs).

$$S \mid d_i = A_j' \cap (A_j - A_j') \qquad \text{(eq 5)}$$

where Aj' and Aj-Aj' are subsets of S

$$A_j' \subseteq S$$
$$(A_j - A_j') \subseteq S \qquad \text{(eq 6)}$$