Preventing design conflicts in distributed design systems composed of heterogeneous agents

Baris Canbaz, Bernard Yannou, Pierre-Alain Yvars

Preventing design conflicts in distributed design systems composed of heterogeneous agents

Baris Canbaz [a, 1], Bernard Yannou [a], Pierre-Alain Yvars [b]

[a] *Ecole Centrale Paris, Laboratoire Genie Industriel, Grande Voie des Vignes, F-92 295, Châtenay-Malabry, France*

[b] *Institut Supérieur de Mécanique de Paris (SupMeca) – LISMMA, 3 rue Fernand Hainaut, 93407, Saint Ouen Cedex, France*

[1] *Corresponding author: baris.canbaz@ecp.fr 0033 (0) 6 36 59 89 63*

Abstract

In distributed design systems, while designers are connected to each other through dimensioning couplings, they have limited control over design and performance variables. Any inconsistency among design objectives and working procedures of heterogeneous designers interacting in the design system can result in design conflicts due to these couplings. Modeling design attitudes can help to understand inconsistencies and manage conflicts in design processes. We extend the conventional bottom-up or design supervision approach through agent-based attitude modeling techniques to a more powerful level. In our model, design agents can set requirements directly on their wellbeing values that represent how their design targets are likely to be met at a given moment of the design process. Some design conflicts can in this manner be prevented at an earlier phase of the design process. Set-based design and constraint programming techniques are used to explore the overall performance of stochastic design collaborations on a product modeled with uncertainties at a given moment of the design process. Monte Carlo simulations are performed to evaluate the performance of our set-based thinking approach, providing a variety of agent attitudes. The results show that the number of design conflicts occurring during the design process and the intensity of design conflicts are both reduced through our collaborative design platform.

Keywords: Collaborative design; Distributed design; Set-based design; Conflict prevention; Constraint satisfaction problem; Agent attitude model; Heterogeneous agents

1. Introduction

Design processes of complex products currently involve considerable effort and expertise from different disciplines. Multiple designers from different disciplines are thus involved in performing collaborative design. The design model converges to a solution through a series of collaborative activities performed during the design process. Since the design problem has multidisciplinary boundaries, a distributed design approach can be adopted. In distributed design systems, the system is decentralized; the global problem is decomposed into sub-problems and distributed to subsystems consisting of one or several designers (Papalambros et al., 1997). Subsystems have limited control over the design variables because of their limited expertise and responsibility. In a sub-problem there are three main problem elements: design variables that can be controlled, design performances that are evaluated and constraints that must be respected. The rest of the global problem excluding a specific sub-problem does not concern the specific sub-system, but it can be only observed if it is shared and necessary. Distributed design tasks allocated

to sub-problems are executed concurrently by subsystems, the global problem converging to a global solution (Zheng et al., 2011).

In the ideal case, true concurrency is expected from distributed design systems where designers can perform their design activities independently. In reality, designers are related to each other through couplings between their sub-problems. Couplings can result in conflicts among designers if some inconsistencies are presented in the design system. Inconsistencies arise from design attitudes reflected by subsystems during the design process. The most significant inconsistency occurs between design objectives of subsystems. Typically, a design problem contains multiple conflicting objectives, so subsystems are forced to make trade-offs. Working procedures of designers influence the performances of others, and inconsistencies present in these working procedures can negatively impact the global solution (Zhao and Jin, 2003). For instance, a designer restricting the design model more rapidly or earlier than others could influence the model more. Subsequent designers are forced to deal with a restricted model which cannot satisfy their own design objectives. If the number of conflicts and intensity of the conflicts increase; the performance of the design process decreases, because individual design objectives are not satisfied in equilibrium. Some significant attempts have been made to coordinate and resolve existing conflicts in distributed systems. Zheng et al. (2011) propose to resolve conflicts by integrating resultant models of conflicting Boolean decisions in individual sub-problems of distributed computer-aided design. Kwon and Lee (2002) define a multi-agent based model that integrates a coordination mechanism. This can manage conflicting agents in a decentralized enterprise in order to resolve interdepartmental conflicts. Koulinitch and Sheremetov (1998) define a constraint-based dynamic design system model that includes facilitator agents which are responsible for coordination and conflict resolution during the design process. When a conflict occurs amongst design agents, facilitator agents send messages to relax some constraints until a consistent solution is obtained. Huang et al. (2006) develop a fuzzy interactive multi-objective optimization model for engineering design. The collaborative relationships among the objectives are improved with adjusting the threshold of satisfaction degree and weighting coefficients of objectives. The least conflicting solution is therefore selected among the generated set of Pareto optimal results. The selected solution gives the maximum satisfaction degree and the minimum divergence of the individual satisfactions of local objectives. Yvars (2009) proposes a collaborative design system where decisions of distributed designer agents are represented with constraints added to the model dynamically. Constraints restricting the design model restrict also the degree of freedom of agents, so that they cannot add anymore constraints to the design model. This results in conflicts that are represented as unfeasible models. Design conflicts are resolved by detecting a compromise solution that maximizes the number of accepted constraints by removing some constraints from the model. While these approaches focus on resolving conflicts that have already occurred, they overlook the idea of preventing and avoiding potential conflicts that have not yet occurred in the process. They interrogate the issue at a late phase of the problem, because the avoidance of a conflicting problem is usually more efficient and less time-consuming than the resolution of a conflicting problem. The approaches outlined above also fail to take into account attitude models of heterogeneous agents. Modeling design attitudes can help understand the design inconsistencies resulting in design conflicts, and as a result certain collaboration strategies can be defined with attitude models.

The technique chosen for modeling the design process significantly affects the collaborative solution emerging from different sub-problems. Devendorf and Lewis (2011) show that the stability of a distributed design system depends on how the process architecture is formed. Two main approaches can be adopted for global design process modeling. These are the top-down design approach and the bottom-up design approach (Fathianathan and Panchal, 2009). In the top-down design approach, decisions are made for parameterization of design variables in order to find detailed solutions that satisfy designer objectives. This approach is considered as a transition from an abstract level to a detailed level: in complex design problems, the effect of any parameter on the solution is usually abstract until the parameter is tested and a detailed solution is obtained. In contrast, the bottom-up design approach consists in defining detailed solutions to identify values of the design variables. With this bottom-up design approach, designers can make decisions on their design performances. The top-down design approach requires detailed decomposition of the problem where all the relations between variables are explicit. However, this may not be possible when the complexity of the design problem is very high and the problem contains too many couplings. Therefore, the effect of the decisions about design variables on design performances is highly uncertain, especially in early design phases. Engineering project failures can increase when it is not possible to predict the effect of the modifications because of the presence of intense couplings in complex design problems. Chanron and Lewis (2005) highlight the difficulty of allocating design variables to subsystems in a coupled problem where the same design variables influence the design performances of several subsystems. The allocation technique is critical, because it can influence the design quality (Kim et al., 2003) or the design process performance (Park et al., 2001). Fathianathan and Panchal (2009) propose the adoption of a bottom-up design approach when these limitations arise from a top-down design approach.

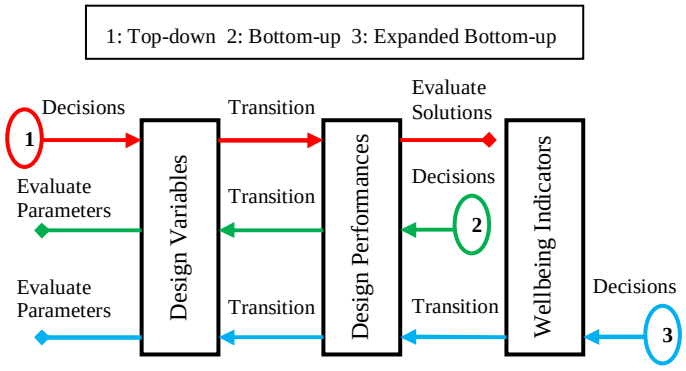

Figure 1. Comparison of process approaches

In this paper we extend the bottom-up design approach with agent-based attitude modeling techniques. A wellbeing indicator is presented that shows how the preference objectives of various designers are satisfied. Figure 1 shows the comparison of our extended bottom-up design approach with the traditional bottom-up approach and the top-down approach. In the top-down design approach, alternatives are generated first by making decisions on the design variables, and emerging solutions are subsequently evaluated considering design performances. In the bottom-up approach, solutions are generated by making decisions on design performance values and the parameters emerging from these values are evaluated to see if they are feasible or if they violate the problem constraints. Thus,

trade-offs are made on design performance values. Traditionally, the bottom-up design approach is modeled at the design problem level: it starts at the lowest level of the physical problem. However it does not include modeling the preferences of designers emerging from their design attitudes. We think that modeling design attitudes and including them at the bottom of the design approach will enable better control of collaborative convergence, because trade-offs can be made directly on satisfaction values of designer preferences. Therefore, the design conflicts can be reduced.

A common design issue, regardless of the design process approach used, is the presence of epistemic uncertainty due to the imprecision caused by the lack of knowledge about the final decision (Parry, 1996). According to Malak et al. (2009) this issue requires representing the uncertainty with imprecise intervals/sets and delaying uncertain decisions to later process stages when the information about the related decision becomes available. In this paper, we use the set-based design (SBD) concept to simulate the process performance of the extended bottom-up design approach modeled with agent attitudes. In section 2 we discuss the ability of SBD and constraint satisfaction problem (CSP) techniques to manage imprecision in design. In section 3 the attitudes of design agents in multi-agent systems and egoistic and altruistic agent characters emerging from dynamic attitudes are considered. Our agent-based SBD model is introduced in section 4 and the CSP simulation process of this model is presented in section 5. Monte Carlo simulations of our approach are performed on a design problem which involves variable agent characters composed of variable design attitudes that define how design agents react during the design process. The sequence of the agent reactions is stochastic. Problem definitions and simulation results are presented in section 6.

2. SBD and CSP techniques

In coupled and conflicting design problems, especially in preliminary design processes, variables cannot be crisply defined due to the lack of information about the decision consequences (Antonsson and Otto, 1995; Yannou, 2004). Even so, in deterministic design methods, crisp values are attributed to variables, so trade-offs are made on design point solutions. Hence, deterministic methods simplify and restrict the design problem in order to optimize it. However, this requires making radical decisions before the information about the decision becomes certain. Therefore, important uncertainty aspects are overlooked. Alternatively, SBD concept considers the design process as an ongoing evolution of non-crisp concurrent decisions (Sobek et al., 1999; Ward et al., 1994). Variables are represented with imprecise values in domains (intervals for real variables), so epistemic uncertainty can be propagated and evaluated. This concept allows information to be gathered before making decisions on the design model, and decisions to be delayed when the information is not certain. The delayed decisions are reconsidered at later process stages where more information has been gathered due the reduction of epistemic uncertainty through earlier decisions. This approach provides flexibility of modifications and higher adaptability to changes as shown by Wang and Terpenny (2003), as well as robustness to design errors as shown by Parsons et al. (1999). Process time is consequently reduced due to a decrease of repetitive design activities and loopbacks.

If SBD has been principally adopted at a managerial level, it is only recently that this concept has been adapted using CSP definitions at a technical solution level e.g. (Meyer and Yvars, 2012; Panchal et al., 2007; Yannou and Harmel, 2006). A CSP is defined with sets of variables, sets of domains that contain the allowable values of

variables and sets of constraints that restrict the problem (Montanari, 1974). The Cartesian product of the variable intervals defines a multidimensional space that contains the consistent values which respect the constraints. A decomposed design problem can be defined with three spaces: the design space defined by design variables, the performance space defined by design performance variables, and the solution space that contains both design and performance spaces. Design decisions are represented with constraints restricting the solutions space, so the epistemic uncertainty is reduced and the remaining solution space is precisely determined with domain reduction/filtering algorithms of constraint programming (CP) techniques. Yannou and Harmel (2004) show how CP techniques can compete with and outperform probabilistic and fuzzy methods on managing imprecision in design. CP techniques allow the bottom-up design approach with enabling constraint definitions on value occurrences. For instance X and Y are integer variables with domains $D(X) = [15, 25]$ and $D(Y) = [10, 20]$ and $Z = X \times Y$ is a value occurrence. If a constraint is defined on Z, its consistency is evaluated and the inconsistent values of X and Y are extracted from their domains. If $Z \leq 200$ the domains of the variables are reduced to $D(X) = [15, 20]$ and $D(Y) = [10, 13]$. Current CSP definitions are able to support a bottom-up SBD, but we are not aware of any CSP platform that includes collaboration indicators derived from design attitudes.

3. Attitudes of design agents in MAS

Through agent-based modeling, many complex phenomena can be considered as systems of autonomous agents that follow simple rules of repetitive, cooperative and competitive interactions. Thus multi-agent system (MAS) simulation is considered as an appropriate approach to investigate complex emergent systems. For instance, MASs have been used for social simulation (Caballero et al., 2011), for modeling bounded rational agents (Lin et al., 2008), and for organization of societies (Rodriguez et al., 2011). Agents are sub-systems that are situated in an environment, and in order to satisfy their design objectives they perform autonomous actions (Wooldridge and Jennings, 1995). In the environment they are social, so they can communicate and interact; they are reactive, so they can perceive the environment and respond to the change in the environment; and they are pro-active, so they can take initiatives by their goal-directed attitudes. In MAS, agents can reflect different attitudes that represent the reactions of agents to uncertainties of complex dynamic domains (Goyal, 2005). The widely deployed architecture of an agent, the belief-desire-intention (BDI) paradigm, is developed by Bratman et al. (1988). BDI views the system as it is emerging from agents with different mental attitudes. The emergent mental attitudes construct the system behavior and are important for the optimal performance of the system. Beliefs correspond to the information emerging from the analysis of the model. Desires correspond to the objectives of the agent and the tasks allocated to it. In a complex emergent system, agents are not able to satisfy all their desires at the same time, so they are forced to make trade-offs and compromise. Intentions correspond to the choices of the agent for some desires when compromise is necessary. Actions of choosing desires are intentions: an agent makes intentions until the desire is satisfied or until the agent believes that the intention is no longer feasible (Cohen and Levesque, 1990). Agents perceive their environment through sensors and act upon that environment through effectors. The system between perception and action consists of their attitudes. An agent is stimulated by the analysis of the model and through its belief, desire and intention architecture its attitudes are defined, so the agent performs actions (Fig. 2). Finally the new form of the model is synthesized following the actions.

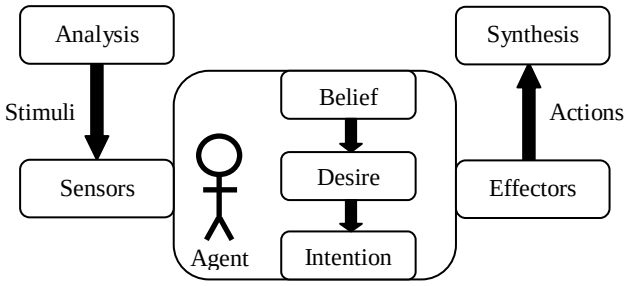

Figure 2. BDI paradigm

A distributed design system is an emergent system and it can be simulated as an MAS. In a distributed design system, the stimuli are sent to agents by the dynamic design model and agents react through defining decision constraints in the dynamic design model. Design attitudes are bounded and design agents need to interact and collaborate. Attitudes of design agents determine when and how their decision constraints are defined during the progress of the design process. This shapes the decision making process and collaborative convergence. The most widely employed decision making model in MAS is the multiple attribute utility theory (MAUT) which evaluates multiple performances. The decision maker agent attempts to maximize the utility function which aggregates all the performances. The utility is used to evaluate solutions while the analysis of trade-offs between alternatives is represented as weighted formulae. Decision makers can also rank alternatives and define preferences on one alternative over another. Preferences reflect agents' objectives and can be prioritized with constraints. Thus, constraints are used to make decisions either statically or dynamically. Therefore, a joint solution is generated by modifying the design model iteratively.

The design system is composed of different people each with different characters. The character of an agent is the combination of its attitudes, and it can be used to establish strategies in order to achieve optimal interactions between various agents (Castelfranchi et al., 1998). However, the attitudes of different agents can result in conflicting activities. This problem usually requires coordination and cooperation of agents' attitudes. MAS can simulate the coordination of different agents composed of different attitudes. The emergent behavior of the system consists of different compositions of altruistic and egoistic behaviors of every agent in the system (Pita and Lima Neto, 2007). Egoistic behaviors are actions that are motivated by self-interested gains, while altruistic behaviors are motivated by the gain of others, such as the pleasure obtained from others' pleasure. Altruism can also be viewed as sacrificing one's own good for the benefit of the group that one belongs to. While egoistic actions can cause harm to the other agents, altruistic actions help the others. A mutual defection may be the rational solution of the agents, but it is neither the most beneficial one for the global benefit nor even for individual benefits. Bazzan et al. (2002) simulate the effects of altruism among agents playing the Iterated Prisoner's Dilemma. They conclude that egoistic agents maximize their benefits only in the short term, but they compromise their performances in the long term. Xianjia and Weibing (2009) propose a method to investigate the evolutionary outcome of the behaviors of players with egoistic or altruistic preference in an iterated prisoner's dilemma. Their results show that egoism can cause defection, and altruism can increase the performance of cooperation. Jennings and Campos (Jennings and Campos, 1997) conclude that the overall performance of the system can be increased if agents are sometimes allowed to work

for the benefit of others. Since agents are autonomous and have different knowledge and resources, cooperation attitudes are conditional to the environment and are dynamic through the allocation of time and resources. Agents are therefore heterogeneous, and it is almost impossible to define optimal agent attitudes. To maintain cooperation among heterogeneous agents, social norms and collaborative strategies should be adopted upstream in the system.

4. Agent-based SBD model

We define the extended bottom-up design approach as an agent-based SBD model that considers the design attitudes of interacting agents. We first define the design process before presenting the design attitudes and the control indicators that derive from these attitudes.

4.1. Design process of agents

In the preliminary design phase, the solution space is very large. While the solution that designers find at the end of the design process is presented in the initial solution space, this solution is not known at the initial state. This implies a very high epistemic uncertainty. CSP definitions can be used to model designer actions. Designer actions are considered as decision constraints defined on the solution space iteratively. The design model is therefore dynamic, evolving with the actions representing decisions. Hence, a collaborative point solution emerges from the converging solution space while the epistemic uncertainty is reduced iteratively during the design process stages. We model an agent-based design process in order to understand both how the epistemic uncertainty is reduced, and how the solution space converges collaboratively. The design process model is shown in Fig. 3. Design agents make three decisions during the process: these are shown as D1, D2, and D3. D1 and D3 are Boolean decisions and D2 is a “how” decision.

D1: Define a decision constraint or not.

D2: How the decision constraint is defined.

D3: Accept the decision constraint or not.

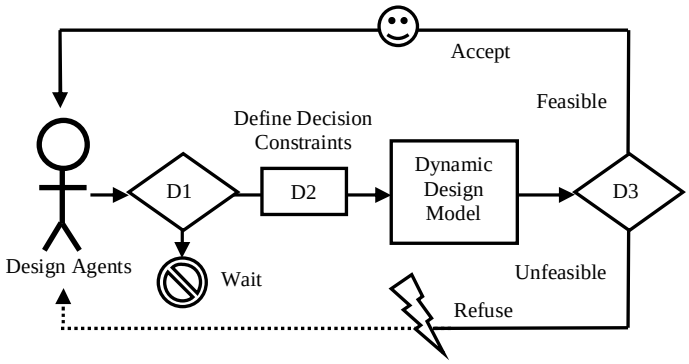

Figure 3. Design process of agents

During a process stage, agents evaluate the solution space to decide whether they will define a decision constraint, or wait. If they decide to define a decision constraint, next they decide how the constraint is defined. When the constraint is defined in the dynamic design model, the model's feasibility is tested. After the definition of the constraint, if there is at least one solution remaining in the solution space, the constraint is consistent for the model and it is accepted collaboratively. If the solution space is empty, then it is refused and rejected. The consistency of the constraint depends on the previously accepted constraints that have been defined by the process stage agent and other agents. In addition, the consistency of the constraint depends on how it is restrictively defined, and the nature of the initial problem. If the constraint is very restrictive, it is probably refused whether there is a previously accepted constraint or not. Therefore, D3 is a collaborative decision which has emerged from the collaborative behavior of the design system. In contrast, D1 and D2 are individual decisions defined by the individual design attitudes of the agents. When a constraint is refused, it is considered as a potential conflict because the agent's desires may not be sufficiently satisfied. The degree of the conflict can be evaluated by the divergence of the agents' individual solutions. The solution space is shared and design objectives are typically conflicting. If an agent can satisfy its desires, it results in dissatisfaction of another agent with conflicting objectives. When their satisfaction solutions diverge - for instance the solution of an agent with a very low satisfaction and the solution of another agent with a very high satisfaction - the conflicts increase in intensity. The conflict is reasonable if only the agent's desires are not sufficiently satisfied. Our proposition is to evaluate design attitudes with a BDI model and evaluate agents' states with control indicators called wellbeing indicators. Wellbeing indicators are derived from the desires of the agents reflected on the beliefs of the agents. They enable a bottom-up design process where convergence is controlled, with defining decision constraints impacting directly on the wellbeing intervals instead of on design variable intervals.

4.2. Attitudes of agents and control indicators

Design space emerges from the intervals of design variables modified dynamically during the design process. This represents the dynamic design model. Analysis of the dynamic design model stimulates design agents and triggers their BDI mechanism. Figure 4 shows design agents' BDI mechanism. Beliefs of design agents are reflected with the intervals of the design performances emerging from the dynamic design model. The bounds of the intervals of the design performances represent the worst possible cases and the best possible cases for the corresponding design performances. Since actions of the design agents are bound through couplings, the intervals propagate some uncertainty. The worst possible cases and the best possible cases depend on the actions of the other agents. Therefore design agents define their desires to adopt the performance values. Desires are design agents' preferences on two factors: design performance alternatives and the satisfaction obtained by design performances. While preferences on design performances reflect agents' attitudes for satisfaction obtained from the alternatives, preferences on satisfaction represent agents' attitudes for compromise. Beliefs and preferences of design agents lead design agents to define their intentions in order to reduce the solution space by improving their worst cases. Intentions are reflected with how frequently and how restrictively their decision constraints are defined. Design agents react to the emerging performance space through defining decision constraints into solution space. These modifications synthesize the next design space in the dynamic process.

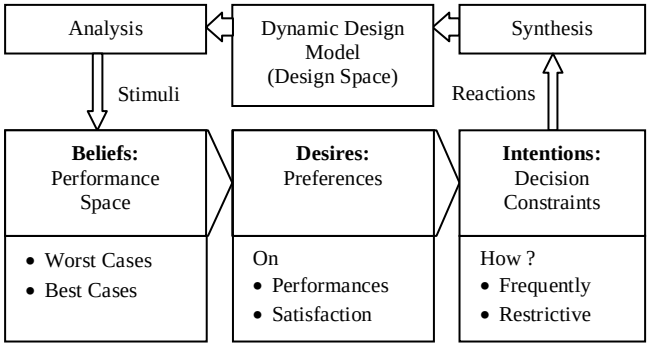

Figure 4. BDI mechanism of design agents

We define an agent *k*, A_k, as an entity with four different attitudes: $A_k\langle Pr_k, T_k, F_k, M_k\rangle$. Pr_k is the set of preferences of the agent on performance values. T_k is the compromise threshold value of the agent, representing the preference of the agent on the satisfaction values for compromise. F_k is the average frequency of the agent for defining constraints in the model and M_k is the coefficient of restriction of the constraints defined by the agent, reflecting the restrictiveness of the decision constraints defined by the agent.

4.2.1. Preferences and satisfaction

Preferences of an agent about design performances can be modeled as a satisfaction function. The list of preferences of an agent A_k on its performances creates the Pr_k attitude. Complete dissatisfaction is represented by 0 on the scale, while complete satisfaction is represented by 1. Design agents are moderately satisfied in the transition between fully satisfied and fully dissatisfied states. In this paper, we assume that the transition is linear; however nonlinear satisfaction functions can be adopted for different studies. We integrate piecewise constraints into the model in order to define information about performance preferences without restricting the solution space. For example, one objective of an agent *k* could be minimizing a performance *i*; the agent is fully satisfied by a performance value below or equal to P_1 and fully dissatisfied by a performance value above or equal to P_2. It is assumed that there is a linear transition between these two preference values. s_{ki} is the satisfaction value of the agent *k* obtained by the performance *i* and v_i is the performance value of the performance *i*. The corresponding integrated piecewise constraints are:

If $v_i \leq P_{1i}$, $s_{ki} = 1$ (1)

If $v_i \geq P_2$, $s_{ki} = 0$ (2)

If $P_1 < v_i < P_2$, $1 > s_{ki} > 0$ (3)

In the SBD framework, all the variables are defined with intervals instead of points. The design process progresses with time and the intervals are reduced through the decision constraints defined on the solution space during the progress. Thus the design process is composed of design stages where agents take actions. At process stage *t*, performance *i* has a minimum value x_i^t and a maximum value y_i^t, the interval of the performance *i* at process stage *t* being $[x_i^t, y_i^t]$. Since the performance is defined with an interval, we obtain an interval for the satisfaction of agent *k* from the performance *i* at stage *t*: $s_{ki} = [mins_{ki}^t, maxs_{ki}^t]$ where $mins_{ki}^t$ is the minimum satisfaction and

$maxs_{ki}^t$ is the maximum satisfaction obtained within the interval $[x_i^t, y_i^t]$. This phenomenon is illustrated in Fig. 5. Piecewise constraints are the same as above. Minimum satisfaction is obtained at point A and maximum satisfaction is obtained at point B. During the process while uncertainty is reduced, agents can observe the potential maximum and minimum satisfaction values from performances. When design agents have several design performances to evaluate, they can assign weights to their satisfaction values considering the importance of the performances for their job. As individual performance satisfaction values are aggregated, general satisfaction states of agents can be observed.

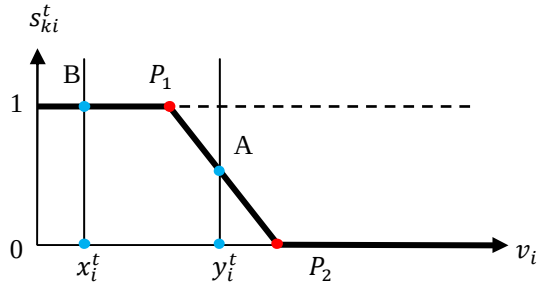

Figure 5. Intervals of satisfaction function

4.2.2. Compromise threshold and wellbeing

In a coupled design system, it is highly unlikely that all the design agents will be fully satisfied. Since design objectives are conflicting, a decision constraint defined to increase the minimum satisfaction value of an agent will decrease the maximum satisfaction value of another agent. Thus, the convergence of satisfaction intervals is bilateral, and design agents are forced to compromise at a certain level on their satisfaction values where maximum and minimum satisfactions are as close as possible. Figure 6 shows a clear example of this phenomenon where Agent 1 and Agent 2 have conflicting objectives, such as decreasing the mass and increasing the volume of a product. Agent 1 defines C_1 and C_3 and Agent 2 defines C_2 and C_4 at different process stages, in order to improve their satisfaction states. However, these constraints decrease the other agent's maximum satisfaction value. Since agents cannot be aware of the other agents' actions, the convergence propagates some uncertainty. Thus, design agents can reflect an attitude of desiring a value in which they may compromise. While preferences on design performances reflect the desires of agents on product specifications, preferences on satisfaction values obtained by these performances reflect the desires about process convergence. The preference about the satisfaction value is called compromise threshold T_k, and it defines the compromise attitude of an agent. This compromise threshold value represents the satisfaction value that an agent wants to guarantee. The agent wants the solution to converge at least to this value. The agent defines decision constraints considering the preferences Pr_k in order to increase the satisfaction obtained from the model until the minimum satisfaction value reaches the agent's satisfaction preference. This introduces a condition for making decisions during the design process. If the minimum satisfaction of an agent by the model reaches or exceeds value T_k, then the agent passes to the compromise state. In the compromise state, agents stop adding decision constraints to the model, so this leaves space to the other agents.

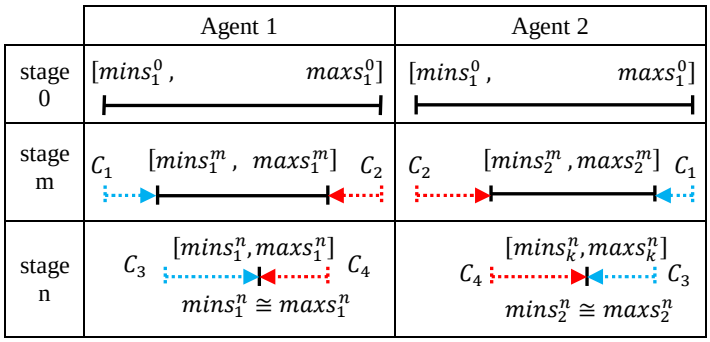

Figure 6. Bilateral convergence

Satisfaction values are normalized through dividing them by the compromise threshold value; this provides wellbeing states Eq. (4, 5). Wellbeing states represent global states of design agents; they show if an agent suffers from not being able to approach the compromise state or if an agent could have performed modifications to the model and thus approached the compromise state. Wellbeing is defined with an interval $wb_k = [minwb_k^t, maxwb_k^t]$ where the minimum value is the minimum wellbeing indicator and the maximum value is the maximum wellbeing indicator. The wellbeing interval converges through the progress of the design process. If the minimum wellbeing value is larger than or equal to 1, then the agent is in a perfect wellbeing state. The worst wellbeing state is when the value is equal to 0.

$$minwb_k^t = \frac{mins_k^t}{T_k} \quad (4)$$

$$maxwb_k^t = \frac{maxs_k^t}{T_k} \quad (5)$$

4.2.3. Frequency

Design agents define decision constraints at an average frequency F_k per process stage. This attitude, dependant on agent character, reflects if agents intend to restrict the solution space more frequently or less. Ph_k is the phase of the decision frequency of A_k. Phases of frequencies can differ from one agent to the other agent depending on their availability and their time zone. A_k defines decision constraints at each process stage t where $(t - Ph_k)$ value is an integer multiple of $1/F_k$. In order to define a consistent function, we assume that $1/F_k$ is integer.

4.2.4. Coefficient of restriction

Any variable of the design problem can be improved with constraints. This improvement increases the minimum satisfaction and wellbeing of the agent. M_k is the coefficient of restriction for the constraints defined by A_k. This attitude defines the restriction effect of the constraints defined on the solution space. M_k is used as an improvement coefficient for the minimum values of the intervals. The constraint defined by A_k at process stage t is C_k^t: $v_k \geq minv_k^t \times (1 + M_k)$ where v_k can be any variable of the design problem and $minv_k^t$ is its minimum value at process stage t. However, $minv_k^t$ value and M_k value should be larger than 0.

If the constraint is consistent for the design model, which means that there is at least one feasible solution after propagating the constraint, then the constraint is accepted. If the constraint is inconsistent, which means that it

returns an unfeasible solution space, then it is refused and rejected from the model. The consistency of the constraint depends on the nature of the initial problem and the earlier constraints defined during progress. Thus it depends on the emerging attitudes of the design agents and propagates an uncertainty.

4.3. Characterization of agents

Depending on their attitudes, agents can have different characters. We consider T_k, F_k, M_k attitudes for characterization of A_k. Pr_k is not considered for characterization, because performance values of different agents may not be the same, and they may not have the same unit of measurement. Besides, T_k attitude reflects the fondness of agents for their preferences. Design agents may be more egoistic or more altruistic compared to the others. More egoistic agents try to satisfy their needs at the highest levels without considering other agents. More altruistic agents have an opposite character, taking other agents into consideration. The solution space is shared between design agents, so any restriction performed by an agent on the solution space will decrease the degree of freedom of the other agents and leave less space to them. As Fig. 6 shows, the reduction of the degree of freedom is on the favorable side of the satisfaction intervals due to the conflicting objectives. Hence, agents with relatively restrictive design attitudes are considered as more egoistic and agents with less restrictive design attitudes are considered as more altruistic.

Figure 7 represents egoistic and altruistic characteristics of agents. When two agents are compared, if F_k and M_k attitudes are identical, the agent with the larger T_k is more egoistic than the other, because it will compromise at a higher satisfaction value. Thus, it will restrict the solution space more than the other, until its objective is satisfied. If T_k and M_k are identical, the agent with larger F_k is more egoistic than the other, because when an agent defines decision constraints more frequently, it will restrict the bounded solution space more rapidly during the process. Consequently, it leaves less space to the other agents. If T_k and F_k are identical the agent with larger M_k is more egoistic than the other, because its decision constraints will be more restrictive than the other agent's decision constraints. This will reduce the solution space for the egoistic agent's benefit.

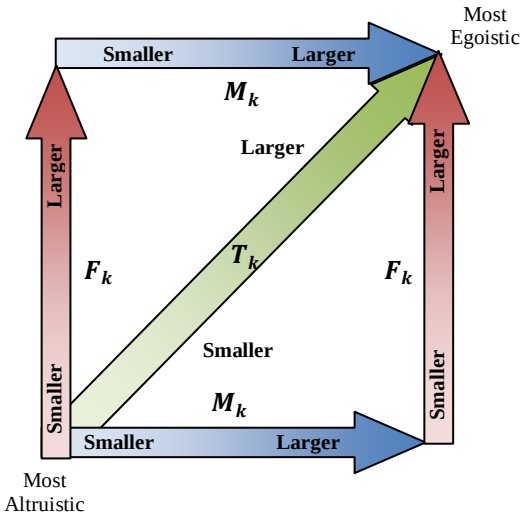

Figure 7. Egoism and altruism in design agents

The desires of design agents and their intentions should be rational. If an agent has egoistic desires its intentions are also egoistic. Therefore, more egoistic agents tend to define decision constraints more frequently with a larger coefficient of restriction, and they do not accept to compromise easily. In contrast, more altruistic agents tend to define decision constraints less frequently with a smaller coefficient of restriction, and they can compromise more easily. However the structure of the rationality between the desires and the intentions can be different from agent to agent, since they model human beings. For example, an agent can have a larger M_k value but a smaller F_k value than another agent with the same T_k value. As seen in Fig. 7, in the extreme case, the most egoistic agent in the design system has the largest T_k, F_k, M_k attitude values while the most altruistic agent has the smallest T_k, F_k, M_k attitude values. The process performance and the design solutions can be influenced by the characters of designers. When the design system consists of heterogeneous agents with different design attitudes, the results may diverge where one agent has a very low wellbeing and another has a very high wellbeing. Process time and the number of conflicts that occur during the design process can also increase due to the non-converging design characters.

5. CSP simulation process

We present an automatic constraint propagating simulation of our model where the solution space is reduced iteratively considering design agents' BDI mechanism. The objective is to simulate some top-down and bottom-up design processes with different combinations of design agent characters, and compare the results that emerge from these processes. Two practical top-down simulation cases are defined. Case 1 represents the design process where a designer can modify only one design variable after the modification of another designer. Case 2 consists of an all-at-once approach where designers can modify all the design variables after the modification of another designer. Next, two bottom-up simulation cases are defined. In Case 3, designers can modify their design performances. Case 4 is our extended bottom-up design process where designers can modify their wellbeing indicators derived from the performances. In the simulation process we used a split mechanism similar to the round-robin strategy that loops on all the variables at process iteration (Granvilliers, 2012). The objective is to obtain an upper value and a lower value that are as close as possible for each interval. Intervals are reduced until a good degree of precision is obtained. The simulation algorithm is shown in Fig. 8. We make some assumptions when defining the simulation process:

- If $(t - Ph_k) \times F_k \in \mathbb{Z}$ and $mins_k^t < T_k$ each agent can define (a) decision constraint(s) only once at any iteration and constraints are defined sequentially. If all the agents are processed in iteration then the process passes to the next iteration: t++.
- Decision constraints are defined for improving the worst case scenarios with a coefficient of restriction $M_k > 0$ or $M_{ki} > 0$ or $M_{kj} > 0 \quad \forall k, i, j$. Initial worst cases are larger than 0: $minVn_j^0 > 0 \ \forall j$, $minPn_i^0 > 0 \ \forall i$, $minwb_k^0 > 0 \ \forall k$. If $mins_k^t \geq T_k$ then the compromising agent is extracted from the splitting loop (Cases 1 and 2: $M_{kj} = 0 \ \forall j$, Case 3: $M_{ki} = 0 \ \forall i$, Case 4: $M_k = 0$).
    - Cases 1 and 2: $Vn_j \geq minVn_j^t \times \left(1 + M_{kj}\right)$ where Vn_j is the normalized design variable j, $minVn_j^t$ is its minimum value at iteration t and M_{ki} is the coefficient of restriction on the performance i defined by agent k.

- Case 3: $Pn_i \geq minPn_i^t \times (1 + M_{ki})$ where Pn_i is the normalized performance i, $minPn_i^t$ is its minimum value at iteration t and M_{ki} is the coefficient of restriction on the performance i defined by agent k.
- Case 4: $wb_k \geq minwb_k^t \times (1 + M_k)$.

- If a constraint is rejected, its related coefficient of restriction value is reduced by half. If the coefficient of restriction value of a variable reaches a precision value (P), then the splitting is stopped for this variable, because the upper and lower bounds are as close as possible considering the precision value. If all the coefficient of restriction values reach the precision, then the simulation process stops.
- Agents' attitudes are defined at the initial state of the process. M_{ki} and M_{kj} values are equal to M_k at the initial state. Pr_k and T_k attitudes do not change during the simulation process because they represent desires.

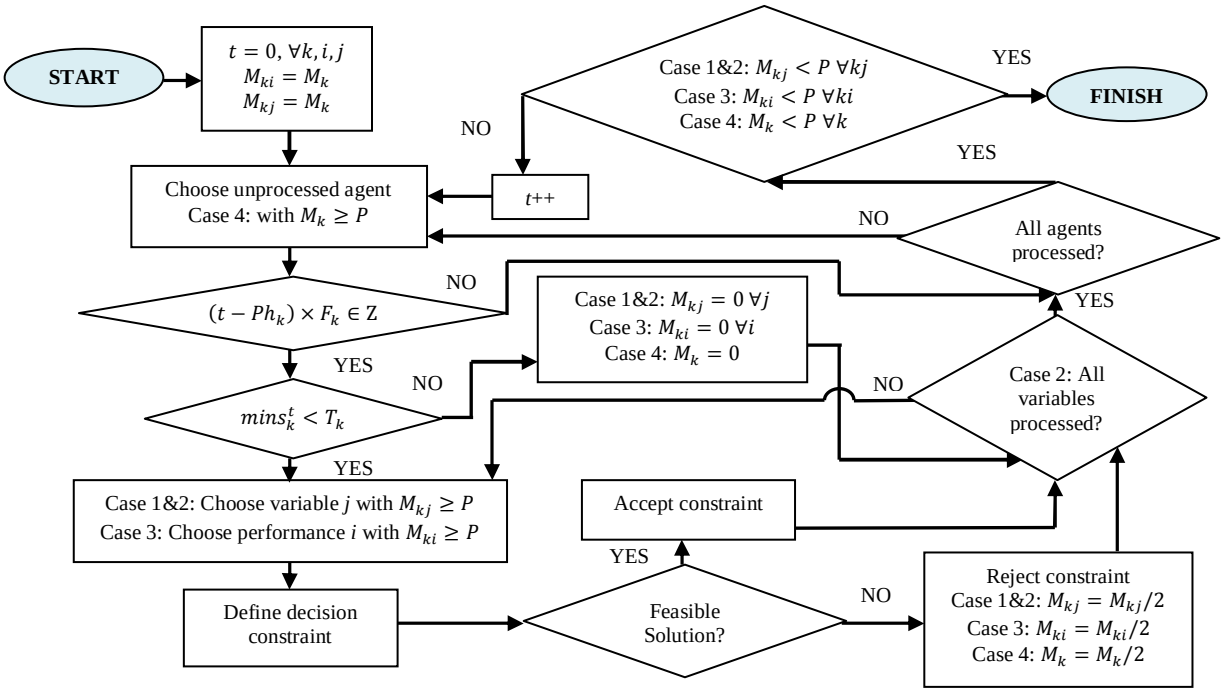

Figure 8. CSP simulation algorithm

The simulation process is evaluated by four performance criteria: number of iterations, number of failures, total wellbeing and divergence of individual solutions. Four simulation cases are compared regarding these process performances. A smaller number of iterations means a faster convergence of intervals and a rapid design process. This should not however be evaluated alone, because when there are less failures, the coefficient of restriction is split later, which leads to the number of iterations increasing. When a decision constraint is rejected, it is a process failure. Each failure is a potential conflict among designers. Therefore less failures means a design convergence with less conflict. The total number of failures represents the number of conflicts occurring in a design process. The objective is to maximize agents' wellbeing values while minimizing their divergence. Divergence is defined as the

difference between their individual wellbeing states. In the ideal case, agents should obtain the same wellbeing values, and each value should be larger than 1. Absolute differences of the wellbeing values of either two element combination represent a vector $D(d_1, \dots, d_n)$. The Euclidian distance of this vector solution to the ideal case solution gives the divergence of the individual solutions: $Divergence = \sqrt{(d_1)^2 + \dots + (d_n)^2}$. More divergent solutions lead to more intense conflicts, because the divergence is caused by agents with a relatively low wellbeing value. However, the divergence cannot be evaluated alone. A zero divergence is not desirable if the total wellbeing is zero.

6. Monte Carlo simulation

We ran a Monte Carlo simulation with the design problem of the pressure vessel in (Karandikar and Mistree, 1992; Lewis and Mistree, 1998). We define three agent characters as shown in Table 1: Egoistic, Moderate, and Altruistic. We consider that there are no frequency phase differences. All four simulation cases are repeated 1000 times for permutations of these characters generated randomly from their attitude sets. Design agents and their design variables and performances are also chosen randomly in process iterations, so the process sequence is completely independent from agent characters.

Table 1 Definitions of random characters

	Egoistic	Moderate	Altruistic
T_k:	(0.6, 0.65, 0.7, 0.75, 0.8, 0.85, 0.9, 0.95, 1)	(0.45, 0.5, 0.55)	(0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4)
F_k:	(0.5, 1)	(1/3, 0.5, 1)	(1/3, 0.5)
M_k:	(5, 6, 7, 8, 9)	(3, 4, 5, 6, 7)	(1, 2, 3, 4, 5)

In the simulation process, the worst cases are improved by increasing the lower bounds of the intervals. Therefore, for minimization objectives the larger bounds are normalized to 0 and the smaller bounds are normalized to 1, and for maximization objectives the smaller bounds are normalized to 0 and the larger bounds are normalized to 1.The precision value is defined as 0.001. This means that if the interval of a variable X: $[minX, maxX]$ does not contain $minX \times 1.001$, it is extracted from the loop. Dynamic CSP is defined in C++ computer language and a CP solver library called IBM ILOG CP V1.6 (IBM, 2012) is used to find solutions through its domain reduction and constraint propagation algorithms. The solve function of IBM ILOG CP is used to examine the feasibility of the model.

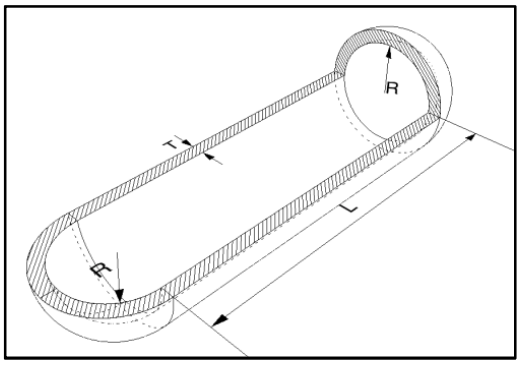

Figure 9. Thin-walled pressure vessel

The design problem consists of a cylindrical thin walled pressure vessel with hemispherical ends as shown in Fig. 9. Problem nomenclature and constant values are given in Table 2. There are three design variables (*R, T, L*) and two design performance variables (*W, V*). Design performance formulas are given in Table 3 and initial constraints are given in Table 4. With given constraints, the weight and volume bounds are determined using CP techniques (Table 4). The design problem is divided into two sub-problems assigned to two designers (Agent 1 and Agent 2). The objective of Agent 1 is to minimize *W* by controlling *R, T* and *L* while satisfying the related constraints; the objective of Agent 2 is to maximize *V* by controlling *R* and *L* while satisfying the related constraints. Their design activities are coupled because of the shared information in constraints and performance formulas. While Agent 1 minimizes the weight, the volume is minimized; while Agent 2 maximizes the volume, the weight is maximized. Since their objectives are inconsistent, their design activities are conflicting.

Table 2 Problem nomenclature and constants

W	Weight of the pressure vessel, lbs.
V	Volume of the pressure vessel, in.^{3}
R	Radius, in.
T	Thickness of the vessel wall, in.
L	Length of the cylinder, in.
P	Pressure inside the cylinder, 3.89 klb.
σ_{UTS}	Ultimate tensile strength of the vessel material, 35 klb.
d	Density of the vessel material, 0.283 lbs./in.^{3}
σ_{circ}	Circumferential stress, lbs./in.^{2}
s_k	Satisfaction of agent *k*
wb_k	Wellbeing of agent *k*
nV	Normalized volume
nW	Normalized weight
nR_k	Radius normalized for agent *k*
nL_k	Length normalized for agent *k*
nT	Normalized thickness
	k=1 for Agent 1 and k=2 for Agent 2

Table 3 Design performance formulas

$W = d\left[\frac{4}{3}\pi(R+T)^3 + \pi(R+T)^2 L - \left(\frac{4}{3}\pi R^3 + \pi R^2 L\right)\right]$
$V = \frac{4}{3}\pi R^3 + \pi R^2 L$

Table 4 Initial constraints

Stress Constraint:	$\sigma_{circ} = \frac{PR}{T} \leq \sigma_{UTS}$
Geometric Constraints:	$5T - R \leq 0$
	$R + T - 40 \leq 0$
	$L + 2R + 2T - 150 \leq 0$
Bounds:	$0.1 \leq R \leq 36$
	$0.1 \leq L \leq 140$
	$0.5 \leq T \leq 6$
	$13.7288 \leq W \leq 56907.6$
	$67.4138 \leq V \leq 480404$

Table 5 Preferences

If $W \leq 8000\ lbs$ then $s_1 = 1$
If $W \geq 31000\ lbs$ then $s_1 = 0$
If $8000\ lbs < W < 31000\ lbs$ then $1 > s_1 > 0$
If $V \geq 380000\ in.^3$ then $s_2 = 1$
If $V \leq 120000\ in.^3$ then $s_2 = 0$
If $120000\ in.^3 < V < 380000\ in.^3$ then $0 > s_2 > 1$

Table 6 Normalizations

$13.7288 \leq W \leq 56907.6$:	$1 \geq nW \geq 0$
$67.4138 \leq V \leq 480404$:	$0 \leq nV \leq 1$
$0.5 \leq T \leq 6$:	$1 \geq nT \geq 0$
$0.1 \leq R \leq 36$:	$1 \geq nR_1 \geq 0$
	$0 \leq nR_2 \leq 1$
$0.1 \leq L \leq 140$:	$1 \geq nL_1 \geq 0$
	$0 \leq nL_2 \leq 1$

Agents define their performance satisfaction functions with piecewise constraints as shown in Table 5. All the transitions between the preferences are considered linear as shown in Fig. 5. Design performances and design variables are normalized using their bound values. Piecewise constraints are defined for normalizations and are shown in Table 6. Agent 1 minimizes *W* through minimizing *R*, *T* and *L* and Agent 2 maximizes *V* through maximizing *R* and *T*. Supplementary initial constraints are defined in order to avoid non-zero worst case scenarios for enabling fruitful constraint propagations ($s_1, s_2, nT, nR_1, nR_2, nL_1, nL_2 \geq 0.01$). These very small constraint values do not affect the performance of the simulation process.

All the permutations of egoistic (E), moderate (M) and altruistic (A) characters of Agent 1 and Agent 2 are simulated for each case 1000 times through a Monte Carlo simulation approach and the average results are shown in Fig. 10. Case 1, the process which enables modifications on design variables only one agent at a time, requires the longest process time because the number of iterations is the largest for every character combination. One of the bottom-up approaches outperforms the second top-down design approach, Case 2, for every character permutation except EM. When there is at least one altruistic agent, Case 4 outperforms Case 3 except AE. The process time should be evaluated with the number of failures, because when the number of failures decreases, M_k is split less, and the convergence continues during subsequent process stages. The number of failures can be considered as the number of conflicts. Case 1 and Case 2 result in the highest number of failures. Case 3 and Case 4 generate significantly less conflicts. Case 4 outperforms Case 3 except when one of the agents is moderate and the other is egoistic or both of them are egoistic. The intensity of the conflicts also needs to be evaluated. A conflict is more reasonable when its intensity is relatively high, because one agent covers more space, resulting in the blocking of the other agent in order to satisfy preferences. When the final wellbeing values are compared, it is significant that in Case 1 and Case 2, Agent 1 dominates Agent 2 regardless of their characters. In Case 3, Agent 2 generally dominates Agent 1 except when Agent 1 is more egoistic than Agent 2 (MA and EA). In Case 4, when agents reflect the same characters, no domination occurs, except when one is more egoistic than the other. These findings are reflected in the divergence results. Case 4 generates significantly the least divergence regardless of agent characters. These conflicts are relatively less intense. However, the reduction of divergence is obtained by compromising some of the total wellbeing value for some character combinations. This compromise of the total wellbeing is due to a slight concaveness of the wellbeing space. The total wellbeing of two agents - one is over-satisfied by exceeding its compromise threshold value, while the other is under-satisfied - can be larger than the total wellbeing of two satisfied agents with wellbeing values equal to 1. This is observed with character combinations that include at least one altruistic agent, because an altruistic agent has a lower compromise threshold value and has greater potential to be over-satisfied.

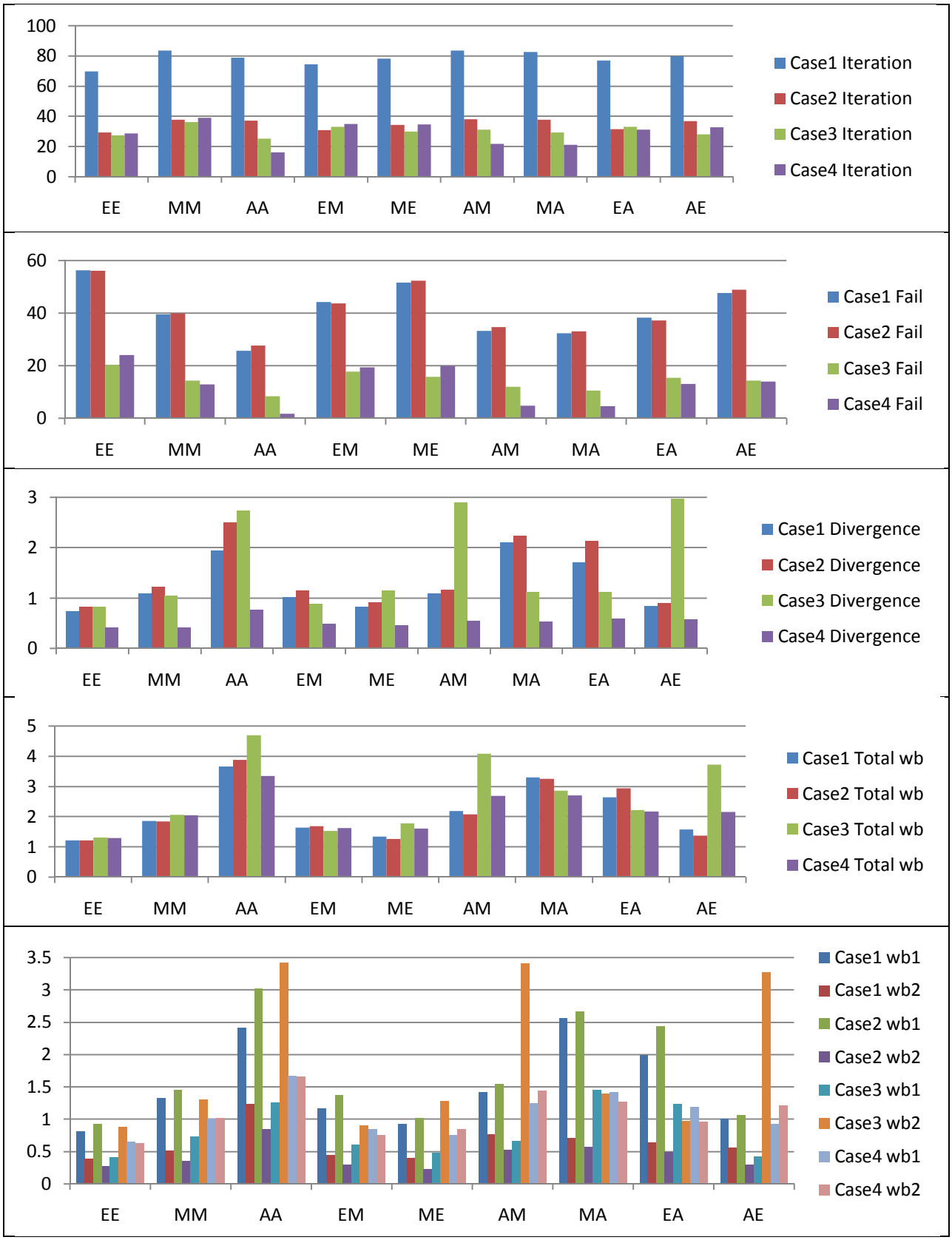

Figure 10. Simulation results

Figure 11 shows the average total satisfaction values and the average absolute differences of the individual satisfaction values obtained by our approach for Case 4. Optimal results are obtained when both agents are moderate, because the absolute difference of satisfaction values is minimal while the total satisfaction value is maximal. Egoistic agents overestimate their desires represented as compromise threshold values when they work with an egoistic agent or a moderate agent, because the total satisfaction values of these situations are not greater than the total satisfaction of the situation where both agents are moderate. Also, the absolute differences of satisfaction values of EE, EM and ME situations are larger than the MM situation. This shows that the individual satisfaction values diverge more because of the egoistic attitudes reflected during the design process. Altruistic agents underestimate their desires when the other agent is altruistic or moderate. Even if altruistic agents can be over-satisfied, the total satisfaction values of AA, AM and MA situations are smaller than the total satisfaction values of the other character situations.

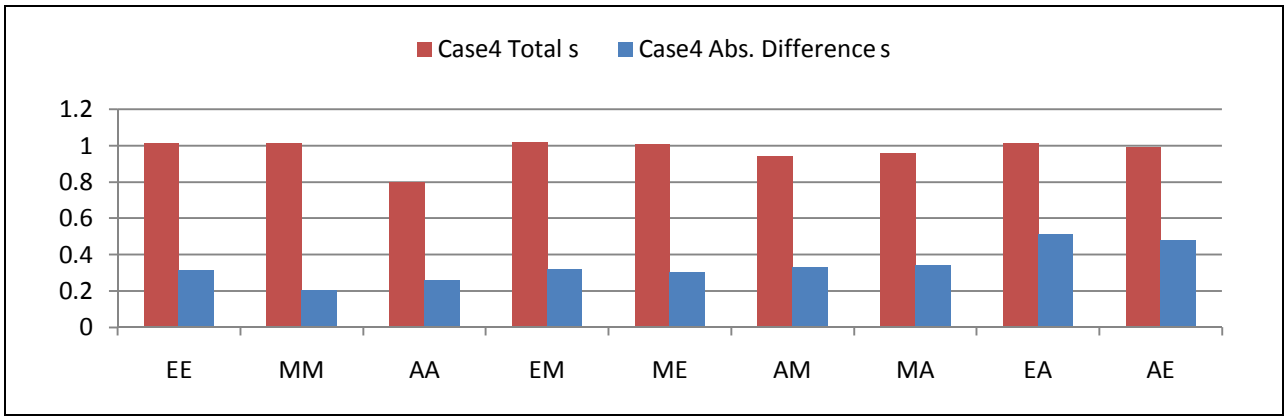

Figure 11. Case 4 satisfaction results

7. Conclusions

In this paper, we define an extended bottom-up design approach, exploring agent-based attitude modeling techniques within the set-based design concept. The conventional bottom-up design approach is usually defined at problem level; however design attitudes that define beliefs, desires and intentions are overlooked at the initial state of the problem, so trade-offs on design preferences remain abstract. In contrast, our extended bottom-up design approach includes design preferences at an earlier state and explores the solution space with design preferences emerging from the desires of various designers.

We perform a CSP simulation for different designer characters. The simulation results show that when design attitudes of heterogeneous designers in distributed design are not evaluated beforehand, the performance of the design process is significantly lower. Regardless of the designer characters, significant dominations usually occur on the same designer. This means that the results are mostly influenced by the process itself. Consequently, individual solutions do not converge in equilibrium, so conflicts are unavoidable. However, when design attitudes are evaluated beforehand, designers can make trade-off intentions on their wellbeing values derived from their beliefs and desires. With this approach, designer domination is relatively less significant and is coherent with designer characters. This shows that the results are only influenced by the design attitudes. Designers can therefore converge

in equilibrium. Consequently, the number of conflicts and the divergence of the solutions that result in the intensity of the conflicts are prevented. However, it should be noted that these process performances are achieved by compromising some of the total wellbeing.

Other conclusions deduced from our approach are about how the satisfaction results emerge from different reciprocal design attitudes. It is shown that reciprocal egoistic attitudes can cause diverging satisfaction results. In contrast, more altruistic reciprocal attitudes can decrease the divergence of individual satisfactions. However, too much altruism can decrease the total satisfaction obtained from the final solution, except when the reciprocating design agent reflects very significant egoistic attitudes.

Our approach is capable of determining and preventing design conflicts, but it does not provide any strategies for resolving existing conflicts. Some cooperative conflict resolution strategies can be defined and integrated into the same platform, but this requires design agents to negotiate, and compromising constraints through relaxing them.

References

Antonsson, E.K., Otto, K.N., 1995. Imprecision in engineering design. Journal of Mechanical Design 117, 25–32.

Bazzan, A.L.C., Bordini, R.H., Campbell, J.A., 2002. Evolution of agents with moral sentiments in an iterated prisoner's dilemma exercise, in: Parsons, S., Gmytrasiewicz, P., Wooldridge, M. (Eds.), Game Theory and Decision Theory in Agent-Based Systems, Multiagent Systems, Artificial Societies, and Simulated Organizations. Springer US, pp. 43–64.

Bratman, M.E., Israel, D.J., Pollack, M.E., 1988. Plans and resource-bounded practical reasoning. Computational Intelligence 4, 349–355.

Caballero, A., Botía, J., Gómez-Skarmeta, A., 2011. Using cognitive agents in social simulations. Engineering Applications of Artificial Intelligence 24, 1098–1109.

Castelfranchi, C., Rosis, F.D., Falcone, R., Pizzutilo, S., 1998. Personality traits and social attitudes in multiagent cooperation. Applied Artificial Intelligence 12, 649–675.

Chanron, V., Lewis, K., 2005. A study of convergence in decentralized design processes. Res Eng Design 16, 133–145.

Cohen, P.R., Levesque, H.J., 1990. Intention is choice with commitment. Artificial Intelligence 42, 213–261.

Devendorf, E., Lewis, K., 2011. The impact of process architecture on equilibrium stability in distributed design. Journal of Mechanical Design 133, 101001.

Fathianathan, M., Panchal, J.H., 2009. Modelling an ongoing design process utilizing top-down and bottom-up design strategies. Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture 223, 547–560.

Goyal, M., 2005. Attitude based teams in a hostile dynamic world. Knowledge-Based Systems 18, 245–255.

Granvilliers, L., 2012. Adaptive bisection of numerical CSPs, in: Milano, M. (Ed.), Principles and Practice of Constraint Programming, Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 290–298.

Huang, H.-Z., Gu, Y.-K., Du, X., 2006. An interactive fuzzy multi-objective optimization method for engineering design. Engineering Applications of Artificial Intelligence 19, 451–460.

IBM, 2012. IBM ILOG CPLEX CP Optimizer for Constraint Programs - Features and benefits [WWW Document]. URL http://www-01.ibm.com/software/integration/optimization/cplex-cp-optimizer/about/ (accessed 2.24.13).

Jennings, N.R., Campos, J.R., 1997. Towards a social level characterisation of socially responsible agents. IEE Proceedings - Software Engineering 144, 11.

Karandikar, H., Mistree, F., 1992. Designing a composite material pressure vessel for manufacture: A case study in concurrent engineering. Engineering Optimization 18, 235–262.

Kim, H.M., Michelena, N.F., Papalambros, P.Y., Jiang, T., 2003. Target Cascading in Optimal System Design. Journal of Mechanical Design 125, 474.

Koulinitch, A.S., Sheremetov, L.B., 1998. Coordination and communication issues in multi-agent expert system: concurrent configuration design advisor. Expert Systems with Applications 15, 295–307.

Kwon, O.B., Lee, K.C., 2002. MACE: multi-agents coordination engine to resolve conflicts among functional units in an enterprise. Expert Systems with Applications 23, 9–21.

Lewis, K., Mistree, F., 1998. Collaborative, sequential, and isolated decisions in design. Journal of Mechanical Design 120, 643.

Lin, R., Kraus, S., Wilkenfeld, J., Barry, J., 2008. Negotiating with bounded rational agents in environments with incomplete information using an automated agent. Artificial Intelligence 172, 823–851.

Malak, R.J., Aughenbaugh, J.M., Paredis, C.J.J., 2009. Multi-attribute utility analysis in set-based conceptual design. Computer-Aided Design 41, 214–227.

Meyer, Y., Yvars, P.-A., 2012. Optimization of a passive structure for active vibration isolation: an interval-computation- and constraint-propagation-based approach. Engineering Optimization 44, 1463–1489.

Montanari, U., 1974. Networks of constraints: Fundamental properties and applications to picture processing. Information Sciences 7, 95–132.

Panchal, J.H., Fernández, M.G., Paredis, C.J.J., Allen, J.K., Mistree, F., 2007. An Interval-based Constraint Satisfaction (IBCS) method for decentralized, collaborative multifunctional design. Concurrent Engineering 15, 309–323.

Papalambros, P.Y., Michelena, N.F., Kikuchi, N., 1997. Distributed cooperative systems design. Proceedings of the 11 th international conference on engineering design 2, 265–270.

Park, H., Michelena, N., Kulkarni, D., Papalambros, P.Y., 2001. Convergence criteria for hierarchical overlapping coordination of linearly constrained convex design problems. Computational Optimization and Applications 18, 273–293.

Parry, G.W., 1996. The characterization of uncertainty in Probabilistic Risk Assessments of complex systems. Reliability Engineering & System Safety 54, 119–126.

Parsons, M.G., Singer, D.J., Sauter, J.A., 1999. A hybrid agent approach for set-based conceptual ship design, in: Proceedings of 10th International Conference on Computer Applications in Shipbuilding. Cambridge, MA.

Pita, M.S., Lima Neto, F.B., 2007. Simulations of egoistic and altruistic behaviors using the vidya multiagent system platform, in: Proceedings of the 2007 GECCO Conference Companion on Genetic and Evolutionary Computation, GECCO '07. ACM, New York, NY, USA, pp. 2927–2932.

Rodriguez, S., Julián, V., Bajo, J., Carrascosa, C., Botti, V., Corchado, J.M., 2011. Agent-based virtual organization architecture. Engineering Applications of Artificial Intelligence 24, 895–910.

Sobek, D.K., Ward, A.C., Liker, J., 1999. Toyota's principles of set-based concurrent engineering. Sloan Management Review 40, 67–83.

Wang, J., Terpenny, J., 2003. Interactive evolutionary solution synthesis in fuzzy set-based preliminary engineering design. Journal of Intelligent Manufacturing 14, 153–167.

Ward, A.C., Liker, J., Sobek, D.K., Cristiano, J.J., 1994. Set-based concurrent engineering and Toyota, in: Proceedings of ASME Design Engineering Technical Conferences. ASME, pp. 79–90.

Wooldridge, M., Jennings, N.R., 1995. Intelligent agents: Theory and practice. Knowledge Engineering Review 10, 115–152.

Xianjia, W., Weibing, L., 2009. Preference and evolution in the iterated prisoner's dilemma. Acta Mathematica Scientia 29, 456–464.

Yannou, B., 2004. Managing uncertainty of product data. An enhancement on constraint programming techniques, in: Tichkiewitch, S., Brissaud, D. (Eds.), Methods and Tools for Co-operative and Integrated Design Methods and Tools for Co-operative and Integrated Design. Kluwer Academic Publishers, Springer, pp. 195–208.

Yannou, B., Harmel, G., 2004. A comparative study of constraint programming techniques over intervals in preliminary design, in: Proceedings of ASME Design Engineering Technical Conferences. ASME, pp. 189–198.

Yannou, B., Harmel, G., 2006. Use of constraint programming for design, in: ElMaraghy, H.A., ElMaraghy, W.H. (Eds.), Advances in Design, Springer Series in Advanced Manufacturing. Springer London, pp. 145–157.

Yvars, P.-A., 2009. A CSP approach for the network of product lifecycle constraints consistency in a collaborative design context. Engineering Applications of Artificial Intelligence 22, 961–970.

Zhao, L., Jin, Y., 2003. Work structure based collaborative engineering design. ASME Conference Proceedings 865–874.

Zheng, Y., Shen, H., Sun, C., 2011. Collaborative design: Improving efficiency by concurrent execution of Boolean tasks. Expert Systems with Applications 38, 1089–1098.