

ANNALS of Faculty Engineering Hunedoara – International Journal of Engineering

Tome XIII [2015] – Fascicule 4 [November]

ISSN: 1584-2673 [CD-Rom; online]

a free-access multidisciplinary publication of the Faculty of Engineering Hunedoara



¹Alain-Jérôme FOUGERES, ²Egon OSTROSI

AGENT-BASED INTELLIGENT CAD MODELLING

¹ ESTA, Laboratoire IRITES-M3M, 90010 Belfort, FRANCE

² UTBM, Laboratoire IRITES-M3M, 90010 Belfort, FRANCE

Abstract: In Intelligent Computer-Aided Design, the CAD models should be conceived to be able to operate or event to behave intelligently. This paper addresses the intelligent behaviour of CAD models: the basic idea is the transformation of conventional CAD models into intelligent objects. The agent paradigm is used to support this transformation. The proposed agents are elementary geometrical objects which, in one hand, incorporate the functions of observation, decision and action, and in the other, possess their own knowledge. Being aware of context, agents interact to form potential regional transitory communities, called regions. Then agents interact with the other agents in a region to recognize each other and to form specific sub-communities, called intelligent features.

Keywords: Intelligent features, Agent-based system, Computer-aided design modelling, Applications of intelligent computing

1. INTRODUCTION

Features technology made it possible to associate form and knowledge in computer-aided design modelling (CAD modelling). Interestingly, the concept of intelligent CAD emerged with features technology. Indeed, the engineering solutions which are able to: a) recognise each other, b) act as a transitory communities and c) form open, dynamic and distributed systems [1] are considered as "intelligent objects". Features emerged from feature recognition are the first "intelligent objects" in CAD environment. Between 1980 and 2000, many methods based on graph theory [2], expert systems [3], volume based decomposition [4], syntactic method [5], neural networks [6] have been developed. However, the concept of intelligent CAD was not supported by concrete industrial need and its developed was hindered by the insufficient maturity of theoretical fundamentals and technological resources [7].

This paper addresses the intelligent behaviour of CAD models. Recently, agent based technologies have been considered in ordered to increase both knowledge level and intelligence of real and virtual objects. An intelligent agent is a computer system that is capable of flexible autonomous action in order to achieve the goals it has set (designed objectives). Such an agent is always located in an environment: it receives input from environment and acts to change this environment [8]. An agent is a system that enjoys the properties of autonomy, reactivity, pro-activeness, and social ability [9]. Agents are organised entities. The organization assumes that there is a set of entities forming a certain unity and whose various elements are subordinated to each other in an integral unit and a convergent activity. Therefore, an organization requires a certain order between entities possibly heterogeneous, which contributes to the coherence [10].

The contribution of this paper consists in introducing intelligent agents in intelligent CAD modelling. The basic idea of the paper is the transformation of conventional CAD models into intelligent objects. Intelligent agents are used to support this transformation. The proposed agents are elementary geometrical objects which, in one hand, incorporate the functions of observation, decision and actions, and in the other, possess their own knowledge. Being aware of the context, the proposed agents interact to form potential regional transitory communities, called regions. Being aware of their belonging in a region, agents are driven by two processes: division and fusion. Agent division is the process by which a parent agent divides into two or more daughter agents. Agent fusion is a process in which several agents combine to form an agent. The emerged agents interact with the other agents in a region to recognize each other and to form specific sub-communities, called intelligent features. Intelligent features are thus emerged network of agents. In their turn, intelligent features are open, distributed and dynamic objects.

This paper proposes the agent paradigm for intelligent CAD modelling. In the second section, using the linguistic hypothesis of product design, a feature representation is presented. In the third section, agents for feature generation and modelling are formalised and modelled. The forth section presents the application of the method. Finally, in the last section, the conclusion and future developments are proposed.

2. FEATURE REPRESENTATION

Automatic feature recognition must resolve the following problems: Feature Representation and Feature Recognition. The first problem involves choosing or developing a method suitable for representing features so that their representation is unique. The second problem involves developing inference procedures able to perform the most complete recognition possible. So, automatic feature recognition is a complex process.

Topologic and geometric entity graph

A feature is a geometric entity defined by its shape and technological characteristics, typically represented by a set of topologically associated faces. Given two finite sets $D^{tpl} = \{D_1^{tpl}, D_2^{tpl}, \dots, D_m^{tpl}\}$ and $D^{geo} = \{D_1^{geo}, D_2^{geo}, \dots, D_n^{geo}\}$ called the set of topologic domains and the set of geometric domains, and given two finite sets of attributes $A^{tpl} = \{a_1^{tpl}, a_2^{tpl}, \dots, a_m^{tpl}\}$ and $A^{geo} = \{a_1^{geo}, a_2^{geo}, \dots, a_n^{geo}\}$, called the set of geometric attributes and the set of topologic attributes, where each attribute is associated with each domain, and $X = \{X_1, X_2, \dots, X_i, \dots, X_m\}$ a set of features. Then any shape feature X_i can be characterized by a set of faces $F = \{f_1, f_2, \dots, f_i, \dots, f_m\}$ that satisfy a set of topologic and geometric relations. These relations are defined for domains corresponding to the set of topologic and geometric attributes, respectively. Table 1 shows typical cases of those attributes and their respective domains. These relations may be represented by the Topologic and Geometric Entity Graph. Thus, for the two given sets F^* and E :

$$F^* = \{(f_i, e_i^*) \mid f_i \in F\} \tag{1}$$

where: $F = \{f_1, f_2, \dots, f_i, \dots, f_m\}$ is a set of faces; $e_i^* = (a_2^{tpl}, a_3^{geo})$ is an entity associated with each face f_i ;

$$E = \{(f_i, f_j, e_{ij}) \mid f_i, f_j \in F\} \tag{2}$$

where: $e_{ij} = (a_1^{tpl}, a_1^{geo}, a_2^{geo})$ is an entity associated with each pair of faces (f_i, f_j) ;

we call $G = (F^*, E)$, the Topologic and Geometric Entity Graph.

In the graphical representation of the topologic and geometric entity graph, the nodes associated with the label $e_i^* = (a_2^{tpl}, a_3^{geo})$ represent the faces and their topologic and geometric relation, and the edges associated with the label $e_{ij} = (a_1^{tpl}, a_1^{geo}, a_2^{geo})$ represent the topologic and geometric relation between a pair of faces (f_i, f_j) .

Table 1. Domains and Associated Attributes

	Topology		Geometry		
	Relative positions	Type of face	Angle	Type of adjacency	Type of face
	a_1^{tpl}	a_2^{tpl}	a_1^{geo}	a_2^{geo}	a_3^{geo}
Domains	adjacent	base	convex	line	plane
	non-adjacent	side	concave	non-straight line	non-plane
	parallel	frontal	flat	other	

Feature grammar

A feature language describes the generation of feature structures, joint elements and attaching elements. A grammar provides the finite generic description of this language. Thus we will focus on finding a feature grammar, which provides generic and productive description of the feature language. Then, a feature grammar is defined as an 8-plet (3):

$$G_{\text{Feature}} = \left\{ \begin{array}{l} V_{\text{structure}}^T, V_{\text{joint-tie}}^T, V_{\text{structure}}^N, V_{\text{joint-tie}}^N \\ S, \nabla, \Lambda, P \end{array} \right\} \tag{3}$$

where: $V_{\text{structure}}^T = \{a, b, c, \dots\}$ is the terminal vocabulary of structures; $V_{\text{joint-tie}}^T = \{0, 1, 2, \dots, j, \dots, m\}$ $m \in \mathbb{N}$ is the terminal vocabulary of joint-tie elements; $V_{\text{structure}}^N = \{A, B, \dots, S, \dots\}$ is the non-terminal vocabulary of structures; $V_{\text{joint-tie}}^N = \{0, I, II, III, \dots, \nabla, \Lambda\}$ is the non-terminal vocabulary of joint-tie elements; S, ∇, Λ are respectively: the structure, joint, and connection axioms.

$$P : \left\{ \left[\begin{array}{c} \alpha \\ \Gamma_\alpha \\ \Delta_\alpha \end{array} \right] \rightarrow \left[\begin{array}{c} \beta \\ \Gamma_\beta \\ \Delta_\beta \end{array} \right] \right\} \text{ is a set of production rules.}$$

Conditional feature grammar. The Feature Grammar represents the purely syntactic side. It does not always allow expressing the full complexity of structural relations between the primitive elements of a feature. If a syntax rule meets mandatory conditions before being applied, then a conditional feature grammar is defined as follows (4):

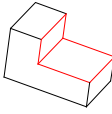
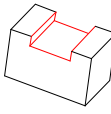
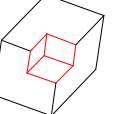
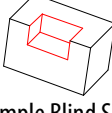
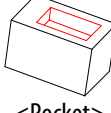
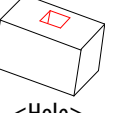
$$G_{Features}^C = \{G_{Features}, A^{geo-tpl}, D^{geo-tpl}, C\} \tag{4}$$

where: $G_{Features}$ is the Feature Grammar; $A_{geo-tpl}$ is the set of geometric and topologic attributes; $D_{geo-tpl}$ is the set of geometric and topologic domains;

$$C = \begin{bmatrix} C_{\alpha \rightarrow \beta} \\ C_{\Gamma_{\alpha} \rightarrow \Gamma_{\beta}} \\ C_{\Delta_{\alpha} \rightarrow \Delta_{\beta}} \end{bmatrix} \text{ are the three levels of semantic conditions.}$$

Application. Given a set of features $X = \{\text{Step, Slot, Blind Slot, Hole, Pocket, Blind Step, Simple Blind Slot, Partial Hole, Hole}\}$, presented in Table 2. A feature can be represented by the selected topologic and geometric entity graph $G = (F^*, E)$, already defined in (1) and (2).

Table 2. Features of the given set X

		
<Step>	<Slot>	<Blind Slot>
		
<Simple Blind Slot>	<Pocket>	<Hole>

A Fuzzy Feature Grammars $G_{Features}^C$ are inferred for the feature classes: $C_1^X = \{\text{Step, Slot, Partial Hole, Hole}\}$ and $C_2^X = \{\text{Blind Step, Simple Blind Slot, Blind Slot, Pocket}\}$ respectively. For the feature class C_1^X , $G_{Features}^C$ is defined as follows (5):

$$G_{Features} = \left\{ \begin{array}{l} V_{structure}^T, V_{junction-connexion}^T, V_{structure}^N, \\ V_{junction-connexion}^N, S, \nabla, \Lambda, P \end{array} \right\} \tag{5}$$

where: $V_{structure}^T = \{a/\mu_a\}$; $V_{junction-connexion}^T = \{0, 1, 2\}$ $m \in N$; $V_{junction-connexion}^N = \{0, I, II, \nabla, \Lambda\}$; $S = \text{Feature}/\mu_{Feature}, \nabla, \Lambda$;

$$P = \{P_0, \dots, P_6\}, \text{ where: } P_0: \begin{array}{c} \left[\begin{array}{c} \text{Feature}/\mu_{Feature} \\ \emptyset \\ \emptyset \end{array} \right] \rightarrow \left[\begin{array}{c} \text{Step}/\mu_{Step} \\ \emptyset \\ \emptyset \end{array} \right] \parallel \left[\begin{array}{c} \text{Slot}/\mu_{Slot} \\ \emptyset \\ \emptyset \end{array} \right] \parallel \left[\begin{array}{c} \text{Hole}/\mu_{Hole} \\ \emptyset \\ \emptyset \end{array} \right] \end{array}$$

$$P_1: \begin{array}{c} \left[\begin{array}{c} \text{Hole}/\mu_{Hole} \\ \nabla \\ \nabla \\ \Lambda \\ \Lambda \end{array} \right] \rightarrow \left[\begin{array}{c} a/\mu_a \ E/\mu_E \\ 1 \ \parallel \\ 2 \ \parallel \\ 1 \ \parallel \\ 2 \ \parallel \end{array} \right] \parallel \left[\begin{array}{c} a/\mu_a \ F/\mu_F \\ 1 \ \parallel \\ 2 \ \parallel \\ 1 \ \parallel \\ 2 \ \parallel \end{array} \right] \end{array}$$

$$P_2: \begin{array}{c} \left[\begin{array}{c} \text{Partial Hole}/\mu_{\text{Partial Hole}} \\ \emptyset \\ \Lambda \\ \Lambda \end{array} \right] \rightarrow \left[\begin{array}{c} F/\mu_F \\ \emptyset \\ \parallel \\ \parallel \end{array} \right] \end{array}$$

$$P_3: \begin{array}{c} \left[\begin{array}{c} \text{Slot}/\mu_{Slot} \\ \emptyset \\ \Lambda \\ \Lambda \end{array} \right] \rightarrow \left[\begin{array}{c} E/\mu_E \\ \emptyset \\ \parallel \\ \parallel \end{array} \right] \end{array}$$

$$P_4: \begin{array}{c} \left[\begin{array}{c} \text{Step}/\mu_{Step} \\ \emptyset \\ \Lambda \\ \Lambda \end{array} \right] \rightarrow \left[\begin{array}{c} D/\mu_D \\ \emptyset \\ \parallel \\ \parallel \end{array} \right] \end{array}$$

$$P_5: \begin{array}{c} \left[\begin{array}{c} F/\mu_F \\ \nabla \\ \parallel \\ \parallel \end{array} \right] \rightarrow \left[\begin{array}{c} a/\mu_a \ E/\mu_E \\ 2 \ \parallel \\ 1 \ 0 \\ 0 \ \parallel \end{array} \right] \parallel \left[\begin{array}{c} a/\mu_a \ F/\mu_F \\ 2 \ \parallel \\ 1 \ 0 \\ 0 \ \parallel \end{array} \right] \end{array}$$

$$P_6: \begin{array}{c} \left[\begin{array}{c} E/\mu_E \\ \nabla \\ \parallel \\ \parallel \end{array} \right] \rightarrow \left[\begin{array}{c} a/\mu_a \ D/\mu_D \\ 2 \ \parallel \\ 1 \ 0 \\ 0 \ \parallel \end{array} \right] \end{array}$$

$$P_7: \begin{array}{c} \left[\begin{array}{c} D/\mu_D \\ \emptyset \\ \parallel \\ \parallel \end{array} \right] \rightarrow \left[\begin{array}{c} a/\mu_a \\ \emptyset \\ \parallel \\ 2 \end{array} \right] \end{array}$$

Structures with the same syntax may represent features with different semantics. Thus we can build the Conditional and Feature Grammar. In this case, the first level of production rules will be associated by conditions. For example, for the first level of production rules P_6 , we have the following semantic condition: structures b and A (on the right side of rule $B \rightarrow bA$) are attached if the direction of the main vector A (on the right side) is the same as the direction of the vector $\vec{n}_1 \wedge \vec{n}_2$ of b , where 1 and 2 represent the attaching elements of b .

The previous condition is used in a similar fashion for rules P_1, P_3, P_5 . We will have the following condition for the first level of production rules: the direction of the main vector of A (left side of the rule $A \rightarrow b$) is initialized from $\vec{n}_1 \wedge \vec{n}_2$ of b , where 1 and 2 represent the attaching elements of b . There are no semantic conditions to be satisfied for the other rules.

Feature identification approach

Using the principles discussed above, we have developed a new feature recognition method composed of five main phases: 1) Regioning, 2) Virtual Extension, 3) Structuring, 4) Identification, and 5) Modelling. In this paper, we consider the three interrelated phases of this method (Figure 1), where:

- ≡ the first phase, called Regioning, consists in identifying the potential zones for the birth of features (see the set $\{R\}$ in Figure 1);
- ≡ the second phase, called Virtual Extension, consists in building links and virtual faces (see the set $\{Ve\}$ in Figure 1);
- ≡ the third phase, called Identification, consists in identifying the features in these zones (see the set $\{F\}$ in Figure 1).

3. AGENTS FOR FEATURE MODELLING

A major part of our research [12, 13] focused on modelling agents with strong interactive capabilities (communication, cooperation, etc.), which may be used as basic components for the design of complex systems. A complex system is "made of many components with many interactions" [14]. So design of complex systems includes: 1) distribution and autonomy of system components, and 2) a very accurate modelling of communicative and interactional levels of these components. The agent-based approach provides a level of abstraction suitable for this problem [15].

Agent modelling

The distribution is the basis of the modelling agent. Autonomy of an agent is technically implemented by: 1) an independent process, 2) an individual memory (knowledge / data of agent), and 3), an ability to interact with other agents and environment (perception / reception, emission / action). Our generic agent model [13] is inspired by Rasmussen's three-level operator model [16]: 1) reflex-based behaviour, 2) rule-based behaviour, and 3) knowledge-based behaviour with interpretation, decision and plan. Agents are both cognitive and reactive: they have behaviours adapted to the tasks they perform. Reactive task is characterised by the cycle <Observation, Execution>, routine task is characterised by the cycle <Observation, Interpretation, Association state/task, Procedure/rules, Execution>, and finally cognitive task is characterised by the cycle <Observation, Interpretation, Decision of task, Planning, Execution>.

A generic agent $\alpha_i \in A$ is described by the following tuple (6):

$$\alpha_i = \langle \Phi_{\Pi(\alpha_i)}, \Phi_{\Delta(\alpha_i)}, \Phi_{\Gamma(\alpha_i)}, \mathcal{K}_{\alpha_i} \rangle \quad (6)$$

where $\Phi_{\Pi(\alpha_i)}$, $\Phi_{\Delta(\alpha_i)}$ and $\Phi_{\Gamma(\alpha_i)}$ are respectively functions of observation (7), decision (8) and action (9). The set of knowledge $\mathcal{K}_{\alpha_i} = \mathcal{P}_{\alpha_i} \cup \Sigma_{\alpha_i} \cup \Delta_{\alpha_i}$ includes decision rules, values of domain, acquaintances (network of affinities), and dynamic knowledge (observed events, internal states).

$$\Phi_{\Pi(\alpha_i)} : (E_{\alpha_i} \cup I_{\alpha_i}) \times \Sigma_{\alpha_i} \rightarrow \Pi_{\alpha_i} \quad (7)$$

where $E_{\alpha_i}, I_{\alpha_i}, \Sigma_{\alpha_i}, \Pi_{\alpha_i}$ are respectively finite sets of observed events, interactions, states and perceptions of agent α_i .

$$\Phi_{\Delta(\alpha_i)} : \Pi_{\alpha_i} \times \Sigma_{\alpha_i} \rightarrow \Delta_{\alpha_i} \quad (8)$$

where $\Pi_{\alpha_i}, \Sigma_{\alpha_i}, \Delta_{\alpha_i}$ are finite sets of perceptions, states and decisions of agent α_i .

$$\Phi_{\Gamma(\alpha_i)} : \Delta_{\alpha_i} \times \Sigma_{\alpha_i} \rightarrow \Gamma_{\alpha_i} \quad (9)$$

where $\Delta_{\alpha_i}, \Sigma_{\alpha_i}, \Gamma_{\alpha_i}$ are respectively finite sets of decisions, states, actions of agent α_i .

Agent-based system modelling

Agents are grouped and organized in an agent-based system. This kind of system is defined as follows (10):

$$M_{\alpha} = \langle A, I, P, O, \Phi_A \rangle \quad (10)$$

where A, I, P, O, Φ_A are respectively a set of agents, a set of interactions between agents, a set of roles that agents can perform, a set of organizations (or communities) defined for agents of A , and a set of functions of agents' generation ($\varphi_{[1,2,3]} : A^n \rightarrow A^m$).

Interactions can be passive when agents receive messages/signals, or active, when it is the result of voluntary actions from agents. An interaction $\iota_i \in I$ between two agents α_s and α_r is defined by the following tuple (11):

$$\iota_i = \langle \alpha_s, \alpha_r, \gamma_c \rangle \quad (11)$$

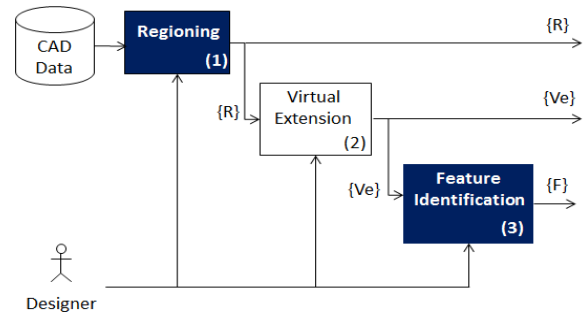


Figure 1. Flowchart of feature identification extracted from the feature recognition method

where α_s is the agent source of the interaction ι_i , α_r is the agent destination, and γ_c is an act of cooperation. For instance, agents can perform the following set of cooperative acts: $A = \{\text{inform, diffuse, ask, reply, confirm}\}$ [17, 18]. These five cooperative acts are sufficient to enable agents to perceive intention of cooperation associated with the proposal contained in a message. A communication act $\lambda_{s,r}$ between two agents ($\lambda_{s,r} \in \Gamma_{\Lambda\alpha_i}$) is defined by (12):

$$\lambda_{s,r} = \langle \lambda, \alpha_s, \alpha_r, \tau, \eta \rangle \tag{12}$$

which can be rewritten $\iota_i = \langle \alpha_s, \alpha_r, \gamma_c \rangle$ and $\gamma_c = \langle \lambda, \tau, \eta \rangle$, where $\lambda \in \Lambda$ is a speech act, α_s is the source agent of communication, α_r is the receiver agent, $\tau \in T$ is a type of message, and $\eta \in H$ is the message, which can be an assertion, a question, a response, etc.

Agents based feature modelling approach

The implementation of the proposed approach performs into the two following phases.

- ≡ Phase 1: Feature agents based systems building. Each face of features is agentified by interactions and cooperation between the agent system and designer(s).
- ≡ Phase 2: Feature recognition. The faces agents interact through messages and activate grammar rules to generate new agents and emerge recognition of regioning, possible virtual extension, and features (networks of face agents).

Feature agent modelling. A feature is an agent network with a goal in a field (manufacturing, maintenance, etc.). Then each feature of the set $X = \{X_1, X_2 \dots X_i \dots X_m\}$ is transformed into an agent α_x , called agents' network, and defined formally as following (13):

$$\alpha_x = \langle A', C \rangle \tag{13}$$

where A' is a set of agents ($A' = \{\alpha_1, \dots, \alpha_n\} \in A$), and C is a set of connections between agents ($c_m = \langle m, \alpha_i, \alpha_j \rangle$); α_i and α_j may be either agent or a sub-network of agents (another feature or sub-feature). For instance, concerning the feature presented in Table 2, $A' = \{\alpha_1, \alpha_2, \alpha_3\}$.

Any shape feature X_i can be characterized by a set of faces $F_{X_i} = \{f_1, f_2 \dots f_j \dots f_m\}$ that satisfy a set of topologic and geometric relations (cf. Table1). Each face f_i of a feature is transformed into an agent α_i (14):

$$\text{Agentification: } F_{X_i} \rightarrow A_{X_i} \tag{14}$$

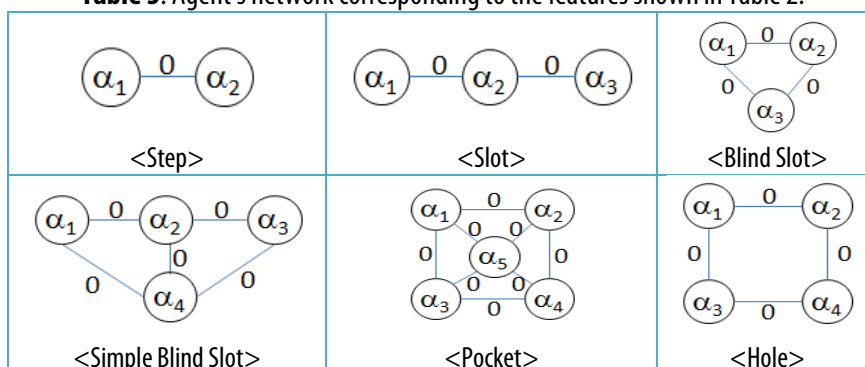
where $F_{X_i} = \{f_1, f_2 \dots f_j \dots f_m\}$ is the set of faces of feature X_i and $A_{X_i} = \{\alpha_1, \alpha_2 \dots \alpha_j \dots \alpha_m\} \in A$ is the set of matching agents. For instance, the following table (Table 3) shows the agentification of the features presented in Table 1.

Feature agent knowledge modelling. Each face agents α_i has knowledge, including:

- ≡ the topologic and geometric relation $e_i^* = (a_2^{tpl}, a_3^{geo})$ of the face f_i (cf. Table 1),
- ≡ a set of triplet $e_{ij} = (a_1^{tpl}, a_1^{geo}, a_2^{geo})$ corresponding to the topologic and geometric relations between pair of faces (f_i, f_j) (cf. Table 1),
- ≡ a set of face agents' networks $\alpha_x = \langle A', C \rangle$, corresponding to the known features, recognized or associated,
- ≡ a set of decision rules Δ_{α_i} and the set of grammar rules $\{P_0, \dots, P_7\}$.

To illustrate the feature agent knowledge modelling, Table 2 describes a simple feature and knowledge of the four face agents $\alpha_1, \alpha_2, \alpha_3, \alpha_4$ representing this feature.

Table 3. Agent's network corresponding to the features shown in Table 2.



Feature agent interaction modelling. Let us consider the feature presented in Table 2. By interaction the three agents $\alpha_1, \alpha_2, \alpha_3$ communicate their geometric and topological characteristics. So they can collectively identify the features they represent together by activating the grammar rules memorized in their knowledge base. For instance, the face agent α_2 recognizes a slot formed by the three agents $\alpha_1, \alpha_2, \alpha_3$.

Feature agent generation. Let us consider the feature presented in Table 2. The face agent α_4 identifies a virtual extension (extension of the face f_4 towards the face f_1) that communicates to the face agent α_1 . The latter is divided into two face agents α_1' and α_1'' by applying the function $\varphi_2 (\varphi_2(\alpha_1) \Rightarrow A' = A + \{\alpha_1', \alpha_1''\})$. Both agents α_1' and α_1'' inherit the knowledge of the face agent α_1 (see Table 4, native knowledge and informed knowledge by designer).

Table 4. Native and nurture knowledge of face agents

Graphical representation	Feature agent's knowledge (native and nurture)	
<p>The graphical representation shows a 3D feature with faces f_1, f_2, f_3, f_4 and a slot. Below it is a Topologic and Geometric Entity Graph (T-EG) with nodes f_1, f_2, f_3, f_4 and edges representing relationships like $\langle 0,1,0 \rangle$. At the bottom, a diagram shows the interaction between face agents $\alpha_1, \alpha_2, \alpha_3, \alpha_4$.</p>	α_1 : $\langle P_0..P_7 \rangle, \langle P_8..P_{15} \rangle, \langle f_1, \alpha_1 \rangle$ // native knowledge $\langle e_{1,1}^*, 1, 0 \rangle, \langle e_{1,2}^*, 0, 1, 0 \rangle$ // informed by designer $\langle e_{2,1}^*, 0, 0 \rangle$ // informed by α_2 $\langle \text{slot}, 1, [\alpha_3, \alpha_2, \alpha_1] \rangle$ // informed by α_2	
	α_2 : $\langle P_0..P_7 \rangle, \langle P_8..P_{15} \rangle, \langle f_2, \alpha_2 \rangle$ // native knowledge $\langle e_{2,1}^*, 0, 0 \rangle, \langle e_{2,1,1}^*, 0, 1, 0 \rangle$ // informed by designer $\langle e_{2,3}^*, 0, 1, 0 \rangle$ // informed by designer $\langle e_{3,1}^*, 1, 0 \rangle, \langle e_{3,1}^*, 1, 0 \rangle$ // informed by α_1 and α_3 $\langle \text{slot}, 1, [\alpha_3, \alpha_2, \alpha_1] \rangle$ // identified feature	
	α_3 : $\langle P_0..P_7 \rangle, \langle P_8..P_{15} \rangle, \langle f_3, \alpha_3 \rangle$ // native knowledge $\langle e_{3,1}^*, 1, 0 \rangle, \langle e_{3,2}^*, 0, 1, 0 \rangle$ // informed by designer $\langle e_{3,4}^*, 3, 1, 0 \rangle$ // informed by designer $\langle e_{2,1}^*, 0, 0 \rangle, \langle e_{4,1}^*, 0, 0 \rangle$ // informed by α_2 and α_4 $\langle \text{slot}, 1, [\alpha_3, \alpha_2, \alpha_1] \rangle$ // informed by α_2	
	α_4 : $\langle P_0..P_7 \rangle, \langle P_8..P_{15} \rangle, \langle f_4, \alpha_4 \rangle$ // native knowledge $\langle e_{4,1}^*, 0, 0 \rangle, \langle e_{4,3}^*, 3, 1, 0 \rangle$ // informed by designer $\langle e_{3,1}^*, 1, 0 \rangle$ // informed by α_3	

4. APPLICATION: REGIONING AND FEATURE IDENTIFICATION.

In this section we propose to apply the method described in Figure 1 on two simple features shown in figure 2 (Feature A and Feature B). We begin by describing the agentification of these features, and then we will detail more specifically the two phases Regioning and Features Identification.

Agentification

Each face of the feature A is transformed in a face agent (Figure 3): $F_{\text{FeatureA}} = \{f_1, f_2 \dots f_{14}\} \rightarrow A_{\text{FeatureA}} = \{\alpha_1, \alpha_2 \dots \alpha_{14}\}$.

The knowledge of each face agents α_i are:

- \equiv the set of face agents $A_{\text{FeatureA}} - \alpha_i$,
- \equiv the geometric relations: $e_i^* = (a_2^{\text{tpl}}, a_3^{\text{geo}})$ and $\{e_{ij} = (a_1^{\text{tpl}}, a_1^{\text{geo}}, a_2^{\text{geo}})\}$ with $\alpha_j \in A_{\text{FeatureA}} - \alpha_i$.
- \equiv the set of decision rules Δ_{α_i} and the set of grammar rules $\{P_0, \dots, P_7\}$.

Regioning

A region defines a potential area of a part where either canonical features or features in interaction may be recognized. During the interaction, features may lose their concavity. As a result, some faces of features in interaction are not identified during the

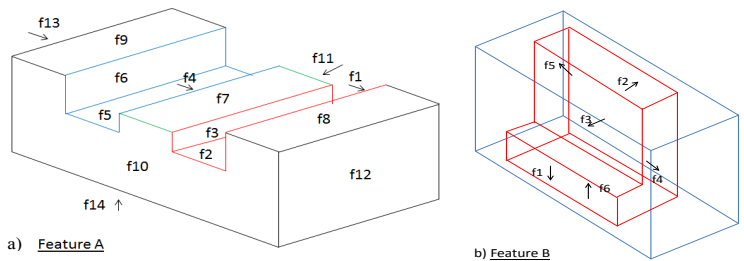


Figure 2. Two simple examples of features: the Feature A and the Feature B

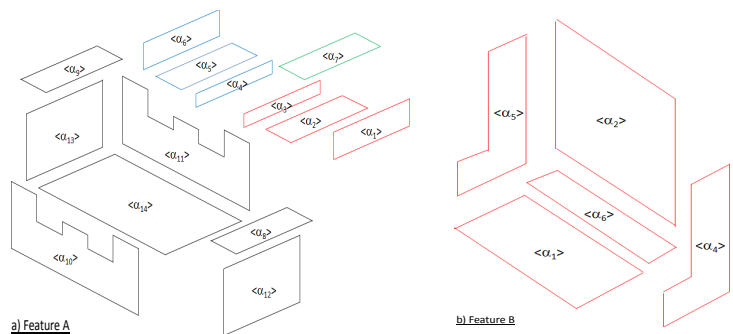


Figure 3. Agentification of a) the Feature A, and b) the Feature B

recognition phase. In this case, the potential region for feature recognition is expanded with concave border faces (local expansion principle). For example, the part shown below (Figure 2.a) contains two regions. The first region comprises the concave faces 1, 2, 3, and the convex face 7, which may be transformed to concave by the virtual extension towards face 1. The second region comprises the concave faces 4, 5, 6, and the convex face 7. In this case, the convex face 7 can be transformed to concave by virtual extension towards face 6. These two regions share face 7. As a result, a macro-region is defined by $\langle \text{macro-region}_1 \rangle \rightarrow \langle \text{région}_{11} \rangle \langle \text{région}_{12} \rangle$ where $\langle \text{région}_{11} \rangle$ and $\langle \text{région}_{12} \rangle$ are the first and second regions, respectively, of the first macro-region $\langle \text{macro-region}_1 \rangle$.

Let us consider the Feature A presented in Table 2. The following table presents (Table 4): 1) the different rules that achieve a regioning and 2) the implementation of these rules in the agent world. In this case a network of agents face is gradually formed by the creation of region agents ($\alpha_{r11}, \alpha_{r12}$), macro-region agents (α_{mr1}), and feature agent (α_{p1}).

Table 4. Rules of regioning used by face agents

Rules			Face agents network
$\langle \text{feature} \rangle$	$\rightarrow \langle \text{macro-region}_1 \rangle$	(15)	
$\langle \text{macro-region}_1 \rangle$	$\rightarrow \langle \text{region}_{11} \rangle \langle \text{region}_{12} \rangle$	(16)	
$\langle \text{region}_{11} \rangle$	$\rightarrow \langle \text{primary-faces}_{11} \rangle$ $\langle \text{secondary-faces}_{11} \rangle$	(17)	
$\langle \text{region}_{12} \rangle$	$\rightarrow \langle \text{primary-faces}_{12} \rangle$ $\langle \text{secondary-faces}_{12} \rangle$		
$\langle \text{primary-faces}_{11} \rangle$	$\rightarrow \langle f_1 \rangle \langle f_2 \rangle \langle f_3 \rangle$	(18)	
$\langle \text{secondary-faces}_{11} \rangle$	$\rightarrow \langle f_7 \rangle$	(19)	
$\langle \text{secondary-faces}_{12} \rangle$	$\rightarrow \langle f_7 \rangle$		

The procedure of regioning is illustrated in the following figure (Figure 4). The 14 faces of Feature A are then agentified, then : a) neighboring agents representing concave faces gather (α_1 - α_2 - α_3 and α_4 - α_5 - α_6); b) the face agent α_7 representing the convex face f_7 gathers with its neighbors; c) the seven face agents (α_1 - α_2 - α_3 - α_7 - α_4 - α_5 - α_6) represent two regions after regrouping with their neighbors $\alpha_8, \alpha_9, \alpha_{10}, \alpha_{11}$; and d) all face agents gathers with the last three neighboring agents ($\alpha_{12}, \alpha_{13}, \alpha_{14}$) to represent a feature.

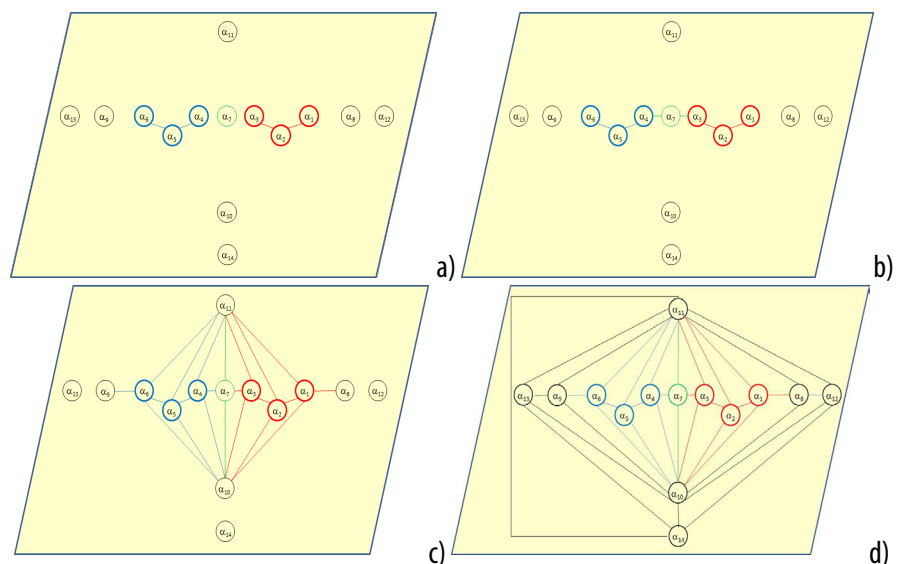


Figure 4. First steps in the process of agent-based regioning of Feature A

Features identification

Let us consider the feature presented in Table 2. P_0, \dots, P_7 (5) are rules that agents can apply when conditions are satisfied, to identify an element of the feature classes $C_1^X = \{ \text{Step, Slot, Partial Hole, Hole} \}$. When a rule is triggered: 1) a connexion between two agents is made, and 2) an agent is created by the function ‘generate (agent)’ of the agent which coordinate the connexion. For instance, $\langle \text{Feature_2} \rangle$ agent and $\langle \text{Feature_2} \rangle$ agent are generated by application of rule P_0 , after the connexion of $\langle \text{Slot} \rangle$ agents (Figure 5).

5. CONCLUSIONS

Intelligent behaviour of models can be particularly useful in of Computer-Aided Design. The CAD models should be conceived to be able to operate or event to behave intelligently. For introducing the intelligent behaviour, a formal method of the transformation of conventional CAD models into intelligent objects is proposed. The proposed method involves on the one hand transforming a geometric model for the part into a feature-based model adapted to the desired engineering view and the using intelligent agents to support this transformation on the other hand.

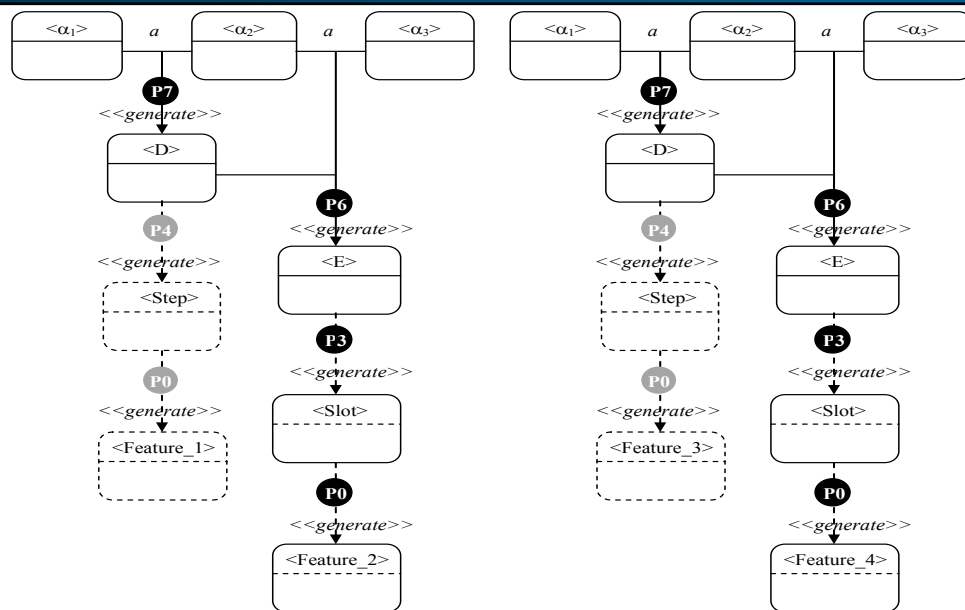


Figure 5. Identification of two $\langle \text{Slot} \rangle$ features

Grammars for feature generation and feature recognition method are used for intelligent agent modelling. From our observation, the result of intelligent behavior of the models shows that recognized features are identified dynamically. Among the advantages of the proposed approach, the following are noteworthy. The proposed approach can be used to analyze the CAD-models as well as to evaluate the choice of recognised features according to multiple view requirements. Furthermore, the approach can promote the capitalization and sharing of the know-how of the designers. Future work will include assessing the importance of the roles of virtual extensions.

References

- [1.] T.S. Lopez, D.C. Ranasinghe, B. Patkai, and D. McFarlane, "Taxonomy, technology and applications of smart objects", *Information Systems Front*, vol. 13, n° 2, pp. 281-300, 2011.
- [2.] M. Marefat and Kashyap, "Geometric reasoning for recognition of three dimensional object features", *Trans. of 8th Army Conf. on Applied Mathematics and Computing*, pp. 705-731, 1990.
- [3.] M.R. Henderson and D.C. Anderson, "Computer recognition and extraction of form features: a CAD/CAM link", *Computers in Industry*, vol. 5, pp. 329-339, 1984.
- [4.] Y.S. Kim, "Recognition of form Features Using Convex Decomposition", *CAD*, vol. 24, n° 9, 1992.
- [5.] E. Ostrosi and M. Ferney "Feature modeling using a grammar representation approach", *International Journal Artificial Intelligence for Engineering Design, Analysis and Manufacturing (AI EDAM)*, vol. 19, no 4, pp. 245-259, 2005.
- [6.] M.R. Henderson, "Manufacturing feature identification", *Artificial Neural Networks for Intelligent Manufacturing*, London: Chapman & Hall, pp. 229-264, 1994.
- [7.] I. Horvath and R.W. Vroom, "Ubiquitous computer aided design: A broken promise or a Sleeping Beauty?", *Journal of Computer-Aided Design*, vol. 59, pp. 161-175, 2015.
- [8.] N.R. Jennings, "On agent-based software engineering", *Artificial Intelligence*, vol. 117, pp. 277-296, 2000.
- [9.] M. Wooldridge, "Agent-based Software Engineering", *IEE Proceedings on Software Engineering*, vol. 144, n° 1, pp. 26-37, 1997.
- [10.] J. Ferber, T. Stratulat, and J. Tranier, "Towards an integral approach of organizations in multi-agent systems: the MASQ approach", in *Multi-agent Systems: Semantics and Dynamics of Organizational Models*, Virginia Dignum (Ed), IGI, 2009.
- [11.] W. Bronsvort and F.W. Jansen, "Feature modelling and conversion - key concepts to concurrent engineering", *Computers an Industry*, vol. 21, pp. 61-68, 1993.
- [12.] A.-J. Fougères, "Agents to cooperate in distributed design, *IEEE Int. Conf. on Systems, Man, and Cybernetics*, The Hague, Netherlands, pp. 2629-2634, 2004
- [13.] A.-J. Fougères, "Modelling and simulation of complex systems: an approach based on multi-level agents", *Int. J. of Computer Science Issues*, vol. 8, n° 6, pp. 8-17, 2011.
- [14.] H.A. Simon, *The sciences of artificial*, MIT press Cambridge, Massachusetts, London, England, Third edition, 1969.
- [15.] M. Wooldridge, *An Introduction to Multiagent Systems*, John Wiley and Sons Ltd, 2002.
- [16.] J. Rasmussen, "Skills, rules, and knowledge; signals, signs, and symbols, and other distinctions in human performance models", *IEEE Trans. on Systems, Man, and Cybernetics*, vol. 13, pp. 257-266, 1983.
- [17.] E. Ostrosi, A.-J. Fougères, and M. Ferney, "Fuzzy Agents for Product Configuration in Collaborative and Distributed Design Process", *Applied Soft Computing*, vol. 8, n° 12, pp. 2091-2105, 2012.
- [18.] D. Choulier, A.-J. Fougères, and E. Ostrosi, "Developing multiagent systems for design activity analysis", *Journal of Computer-Aided Design*, vol. 59, pp. 201-213, 2015.