



ELSEVIER

Computer-Aided Design 36 (2004) 849–871

COMPUTER-AIDED  
DESIGN

[www.elsevier.com/locate/cad](http://www.elsevier.com/locate/cad)

## Design formalism for collaborative assembly design

Kyoung-Yun Kim<sup>a,1</sup>, Yan Wang<sup>a,1</sup>, Obinna S. Muogboh<sup>b,2</sup>, Bartholomew O. Nnaji<sup>a,\*</sup>

<sup>a</sup>Center for e-Design, University of Pittsburgh, 1048 Benedum Hall, Pittsburgh, PA 15261, USA

<sup>b</sup>Lagos Business School, 2 Ahmed Onibudo Street, Victoria Island, Lagos, Nigeria

Accepted 12 September 2003

### Abstract

Joints in product design are common because of the limitations of component geometric configurations and material properties, and the requirements of inspection, accessibility, repair, and portability. Collaborative product design is emerging as a viable alternative to the traditional design process. The collaborative assembly design (AsD) methodologies are needed for distributed product development. Existing AsD methodologies have limitations in capturing the non-geometric aspects of designer's intent on joining and are not efficient for a collaborative design environment. This paper introduces an AsD formalism and associated AsD tools to capture joining relations and spatial relationship implications. This AsD formalism allows the joining relations to be modeled symbolically for computer interpretation, and the model can be used for inferring mathematical and physical implications. An AsD model generated from the AsD formalism is used to exchange AsD information transparently in a collaborative AsD environment. An assembly relation model and a generic assembly relationship diagram are to capture assembly and joining information concisely and persistently. As a demonstration, the developed AsD formalism and AsD tools are applied on a connector assembly with arc weld and rivet joints.

© 2003 Elsevier Ltd. All rights reserved.

**Keywords:** Assembly design; Design formalism; Service-oriented architecture; e-Design and realization; Collaborative assembly design; Joining process; CAD/CAM; CAE

### 1. Introduction

In today's global market, true competitive advantage can only result from the ability to bring highly customized quality products to the market at lower cost and in less time. Product development has become a very complicated process. Discrete product manufacturers are under pressure from customers to move away from the traditional *make-to-stock* production model to a *build-to-demand* model. Many customers are no longer satisfied with mass-produced goods. They are demanding customization and rapid delivery of innovative products [1,2]. Industries now realize that the best way to reduce life cycle costs is to evolve a more effective product development paradigm using

the Internet and web-based technologies. Yet there remains a gap between these current market demands and current product development paradigms. Today's solid modeling systems for assembly components, while adequate for visualization purposes, cannot support other design activities, such as joining process analysis, manufacturing analysis, and product design intent analysis [3].

Collaborative assembly design (AsD) is emerging as a viable alternative to the traditional AsD process, in which an AsD can be developed via an iterative process between designers, manufacturers, marketing people, and ultimately customers at different locations. This emergence can be linked to the recent outburst of growth in the development of the Internet and associated technologies. There are some research efforts that are investigating the assembly methodologies and protocols necessary for distributed AsD. However, it is not still fully clear how assembly and joint design should be implemented in collaborative design environments. None of the existing research has developed assembly formalism that accommodates joining processes. Thus, there is a strong need to develop an assembly formalism to capture general

\* Corresponding author. Department of Industrial Engineering, University of Pittsburgh, 1048 Benedum Hall, Pittsburgh, PA 15261, USA. Tel.: +1-412-624-9830; fax: +1-412-624-9831.

E-mail addresses: nnaji@engrng.pitt.edu (B.O. Nnaji), kykim@pitt.edu (K.-Y. Kim), yawst4@pitt.edu (Y. Wang), omuogboh@lbs.edu.ng (O.S. Muogboh).

<sup>1</sup> Tel.: +1-412-624-9830; fax: +1-412-624-9831.

<sup>2</sup> Tel.: +234-1-320-0695.

<b>Nomenclature</b>	
FF	form feature
DC	dimensional constraint
MF	mating feature
JF	joint feature
AF	assembly feature
MB	mating bond
MP	mating pair
MC	mating condition
$P_j^i$	member of part class $\mathbf{P}$ , $P_j^i \in \mathbf{P}$
$FF_{jk}$	member of form feature class $\mathbf{FF}$ , $FF_{jk} \in \mathbf{FF}$
$\mathbf{A}_i$	assembly structure class
$J$	member of the assembly operation class $\vartheta$ , $J \in \vartheta$
$R$	member of the relationship class $\mathcal{R}$ , $R \in \mathcal{R}$
$DC_r$	member of dimensional constraint class $\mathbf{DC}$ , $DC_r \in \mathbf{DC}$
$RC_{pq}$	relational constraint between $FF_{jp}$ and $FF_{jq}$ , $RC_{pq} \in \{0, 1, 2\}$
	$RC_{pq} = \begin{cases} 0, & \text{if } FF_{jq} \in FF_{jp} \\ 1, & \text{if } FF_{jp} \in FF_{jq} \\ 2, & \text{otherwise} \end{cases}$
$MF_r$	member of mating feature class, $MF_r \in \mathbf{MF}$ , $MF_r \in FF_{jk}$
$JF_r$	member of joint feature class, $JF_r \in \mathbf{JF}$ , $JF_r \in FF_{jk}$
$:$	$\rightarrow$ belong-to relation
$\Leftrightarrow$	inter-feature association relation
$\otimes$	assembly/joining relation

assembly relationships and joining relationships of an assembly.

In order to achieve high performance of a product in its overall life-cycle, an intelligent AsD system should be able to assist a designer during the product assembly and joint design process. This can be achieved by, predicting expected AsD problems, providing alternative suggestions, and eventually solving assembly and joining problems. An ideal intelligent AsD system should have the capability of employing *spatial relationships* and joining protocols that result in the physical realization of an assembly. Existing designer systems have limitations on capturing the non-geometric aspects of designer intent on an assembly with joints. The result is that the designer in a CAD environment cannot completely specify joining relationships on an AsD. Therefore, the development of an assembly formalism to specify the joining relationships symbolically is a prerequisite for an intelligent assembly modeling system.

In this paper, an assembly formalism, which can provide mathematically solvable implications, is developed and AsD tools are developed in a service-oriented collaborative design environment. By using this AsD formalism, assembly and joining relations are extracted from the assembly and represented symbolically. The capturing of assembly and joining relations to preserve design intent is accomplished by using a spatial relationship kernel, which represents the relationships within mechanical assemblies. The AsD formalism can provide a concurrent environment for designers to assign spatial relationships and other assembly specifications onto the parts' nominal geometry and predict mathematical implications. Also, it leverages an efficient design data sharing mechanism and transparent assembly information flow in a collaborative assembly development processes, which includes joining analysis to predict physical effects from the specified joining processes during the actual AsD stage and not after it.

## 2. Background and literature review

An assembly is a collection of manufactured parts, brought together by assembly operations to perform one or more of several primary functions. Assembly operation is defined as the process or series of acts involved in actual realization of assembly. Joining finalizes the assembly operation and generates joints. Messler [4] divided the primary functions of the assembly into three categories: structural, mechanical, and electrical. Usually, assemblies perform multiple functions, with some function being primary and the others secondary. Thus, the joints in an assembly also perform multiple functions. An example is an automobile welded space frame. The assemblies must be capable of carrying loads, so they must be structurally sound. The loads being carried are another important consideration to the purpose of creating or permitting motion. In this case, the primary function of the joints in the automobile frame is to provide a structural connectivity. Also, it may have a secondary function of allowing certain movement corresponding to vibration of the structure. To achieve desired functions, diverse material properties and multiple parts are often employed. In this instance, joints must be created between those different components and different materials. The joining of different materials to achieve function is often a challenging aspect of joining, such as the joining of transparent and brittle glass with a tough structural metal frame.

To enable material and structural optimization, an appropriate joint design is critical and can provide additional benefits, in terms of damage tolerance by changing properties along a potential crack path, and by disrupting and arresting crack propagation. Local joints should be compatible to the overall structure design. If a deformation effect of a weld joint on a metal frame is propagated onto a windshield area, it can result in a fitting distortion problem between the window and the metal

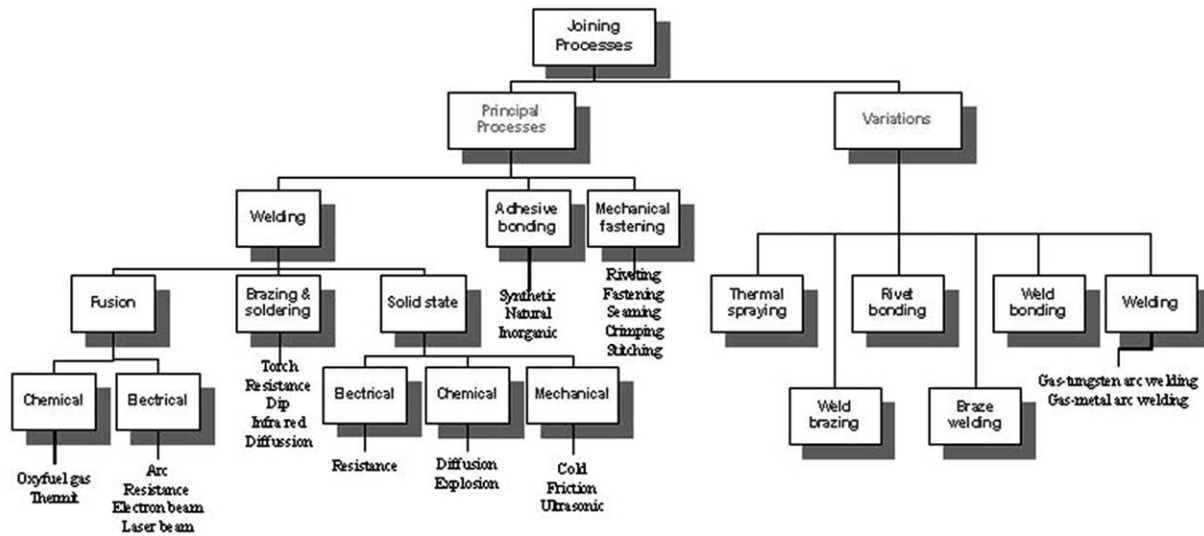


Fig. 1. Joining processes, adapted from Ref. [10].

frame [5]. Fig. 1 shows various joining processes. In this paper, welding and riveting processes are specifically considered as a framework. Some researchers [6–9] developed mathematical models and applications to optimize welding and riveting processes. However, they did not present a methodology to connect their process optimization tools and assembly modeling tools. Appropriate joining processes should be selected considering various constraints, such as material, manufacturing, assembly, and geometric constraints, as well as their physical effects. Through this research, a fundamental AsD formalism has been developed to impose joining information on an assembly. This joining information is essential to determine a proper joint design.

### 2.1. Distributed assembly design

Concurrent engineering offers substantial benefits for new product development, and many companies are taking a strong interest in the collaborative approach. Distributed AsD is emerging as a viable alternative to the traditional AsD process, in which an AsD can be developed via an iterative process between designers, manufacturers, marketing people, and ultimately customers at different locations. This emergence can be linked to the recent outburst of growth in the development of the Internet and associated technologies such as the Java programming language, as well as the rapid advancements made in computing technology that have led to the proliferation of powerful and yet affordable computers. In fact, there are already several systems, both commercial and research, that are investigating the formalisms and protocols necessary for collaborative engineering [11–15]. However, it is not still fully clear how AsD should be implemented in collaborative design environments [16–18].

Several research teams have generated partial solutions to distributed concurrent design and development problem. Wagner et al. [19] performed a feasibility study for how the Internet can be used as an interactive resource during design and manufacturing process. They tested their concept on a simple fixture design. Cheng et al. [20] presented a methodology to implement an Internet and Java-based design support system. Boujut et al. [21] presented a distributed design system for the design of forged parts in attempt to achieve agility in design and manufacturing. Mervyn et al. [22] employed Java RMI (Remote Method Invocation) and XML technologies to realize an interactive fixture design system. However, their works have limitations to evaluate assembly models with respect to manufacturability, assemblability, and ‘joinability’.

### 2.2. Service-oriented collaborative product development

The worldwide availability of technology, capital, information, and labor makes today’s manufacturing enterprises global. Within this distributed economic and technological environment, the problem of how to let engineers collaborate globally during the product development period arises. Information incompleteness, inconsistency, and improcessability are problems that collaborative design groups are facing. Collaborative design tools are needed to improve the collaboration among distributed groups, endorse knowledge sharing, and assist better decision making.

Instead of looking at various engineering tools from the traditional computation viewpoint, Nnaji et al. [23] focused on the engineering implications of those tools from a more abstract level. This approach assures good openness of collaborative design and engineering systems. Their view of design collaboration is service-oriented. With the rapid growth of the number of networked computers, a tremendous amount of resources is available online. The Internet

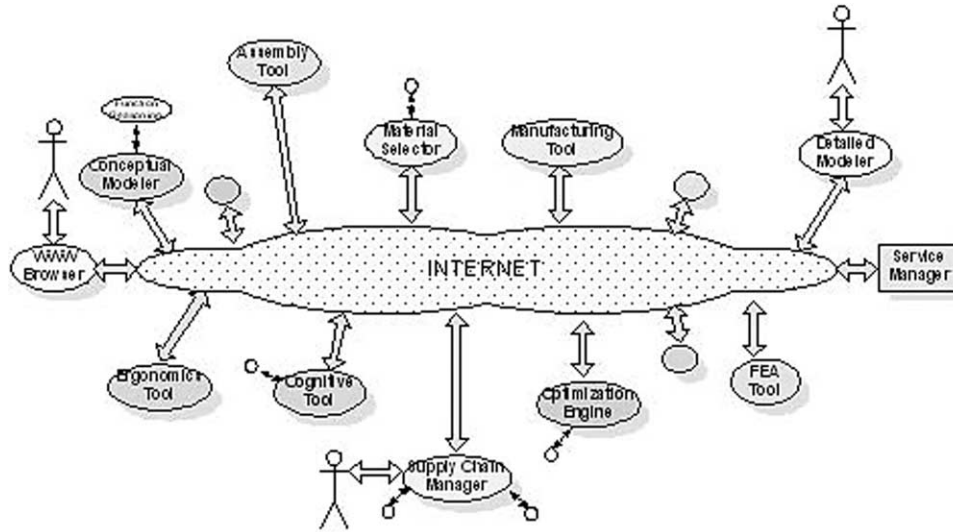


Fig. 2. Peer-to-peer relationships in service-oriented architecture.

is no longer a simple network of computers. From an application perspective, the Internet is a network of potential services. For example, in a three-tier web-based database system consisting of the web browser, server, and database, the web browser provides web document presentation services for human users; the web server provides data processing and retrieval services for the web browser; and the database provides data storage services for the web server. The Internet can be regarded as a complex system of service chains. Computer-aided design and engineering tools can be hooked up to the design platform through the Internet and provide certain services resulting in a distributed product development environment. This incorporates different engineering services and makes them available for automatic transactions in a collaborative AsD environment. This product development environment is called an ‘e-design and realization environment’.

In a service-oriented collaborative architecture, CAD/CAM/CAE tools are connected by the Internet to form

a distributed product development environment, which incorporates different engineering services over the Internet and making them available for transparent transactions in product development. Each design tool for a designer system can be a server that provides certain services requested by clients, either within or external to the designer system [23]. As shown in Fig. 2, servers within the system have a peer-to-peer relationship with each other.

Service is defined as a process that provides a functional use for a person, an application program, or another service in the system. Services should be specified from the functional aspect of service providers. To make an existing tool available online or to build a brand new tool for such a system, services associated with this tool should be defined. The service transaction among service providers, service consumers, and the service manager within this architecture is shown in Fig. 3. Once a service is registered at a central administrative manager, called the service manager, it is then available within the whole system. This process

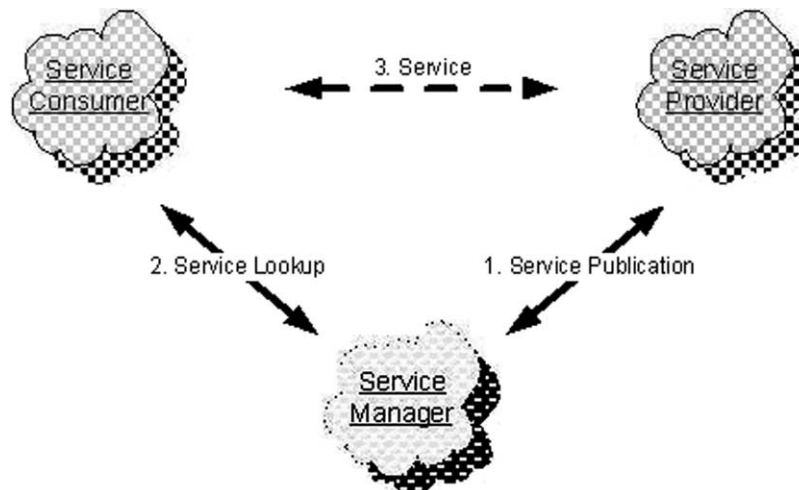


Fig. 3. Service triangular relationship.

is *service publication*. When a service consumer within the system needs a service, it will request a lookup service from the service manager. This process is *service lookup*. If the service is available, the service consumer can request the service from the service provider by the aid of the service manager. Most importantly, this service triangular relationship should be built at run-time. The service consumer (client) does not know the name, the location, or even the way to invoke the service from the service provider (server) during the system and tools development period.

Service providers that provide different services such as AsD component repository, finite element analysis, material, etc. can be developed independently. As shown in Fig. 2, servers that can provide different engineering services are linked by the Internet. Each node in this network can both require and provide certain services, thus, it can be both client and server at different times. The client/server relationship is determined at run-time, so the system is open for the future expansion and extension, in case more services are available.

AsD participants, such as customers, suppliers, assembly designers, production engineers, and other stakeholders, need to exchange AsD information seamlessly in a collaborative environment. There should be common data models and protocols available for them to share the information. Those models and protocols should be widely acceptable and easily implementable, as well as efficient for information transferring. In this paper, AsD models are generated based on the AsD formalism and exchanged among design participants through the service-based architecture.

### 2.3. Existing assembly design formalism

Related to AsD formalisms, Deneux [24] discussed the necessity of assembly feature (AF) in design of complex assembly. Holland and Bronsvort [25] proposed the concepts of a single-part feature model and an AF model for assembly modeling. However, their AF concepts, which considered only assemblies with mechanical fastening, cannot be employed to assembly modeling requiring various other joining processes. Whitney et al. [26,27] proposed a formalism for AsD and focused on only fully constrained assemblies and sub-assemblies. Even though they presented a general methodology for AsD, many spatial relationships in actual mechanical assemblies, e.g. between two cylindrical surfaces, between spherical surface and other surfaces were not addressed. Also, effects on spatial relationship from joining processes were not discussed.

Rémondini et al. [28] proposed assembly operators to deal with the interface between geometric models and analysis models. However, their research focused on geometric aspects of the mechanical analysis model and did not include the formalism to capture the information about joining methods of assembly, which is essential to represent the relationship between mechanical analysis

model and the joining method. Fu et al. [29] used a graph grammar to represent and transform geometric features. However, the method was to represent geometric features of single parts and thus it has limitations to represent joining relationship between geometric features.

None of these researchers has developed an assembly formalism that accommodates downstream AsD activities, such joining analysis and design intent analysis in a collaborative AsD environment. Thus, there is a strong need to develop an assembly formalism to capture designer's intent on AsD and to consider assembly processes and their effects.

### 2.4. Product assembly modeling and spatial relationships

Spatial relationships were first proposed by Ambler and Popplestone [30] in 1975 to describe the relative positions of parts in their final state by specifying feature relationships among them. The spatial relationships include against, coplanar, fits, parax, lin, rot, and fix. In the work of Popplestone et al., the spatial relationships are concentrated on the configuration of a part. Liu and Nnaji [31] focused on the mechanical assembly specifications as well as the configuration of a product, so that spatial relationships can be applied to general assembly and are capable of accepting the design specifications. Design with spatial relationships was defined to infer the assembly positions, as well as to capture designer's intentions. Each spatial relationship, e.g. against, parallel, aligned, incline-offset, include-angle, etc. (Fig. 4), constrains the degrees of freedom (d.o.f.) of motion between the mating entities (face, centerline, center point, etc.).

In spatial relationships, two faces are said to be against if the two faces are touching at some point and normal vectors of those faces are in opposition where they touch. Any combination of two geometric features can possess this property. Two features are aligned if their centerlines are collinear. By selecting the appropriate combination of spatial relationships, the relative mating position with moving d.o.f. of motion can be inferred. The spatial relationships maintain this relative mating position irrespective of the size of the mating entities. Liu and Nnaji [32,33] applied spatial relationships in their work based on a constraint-based product modeling environment for mechanical assemblies, which evolved a framework for a collaborative design advisory system.

RoboWeld-S, an automatic process planner for robotic arc welding of Sheet Metal Weld Assemblies (SMWA) [34,35] was developed based upon spatial relationships for assemblies. Spatial relationships and AF formations relevant to the feature-based modeling of SMWA were presented in the work. However, their methodology cannot explain physical and mathematical effects before and after joining. Also, the AF should be expanded to represent assemblies requiring joining processes.

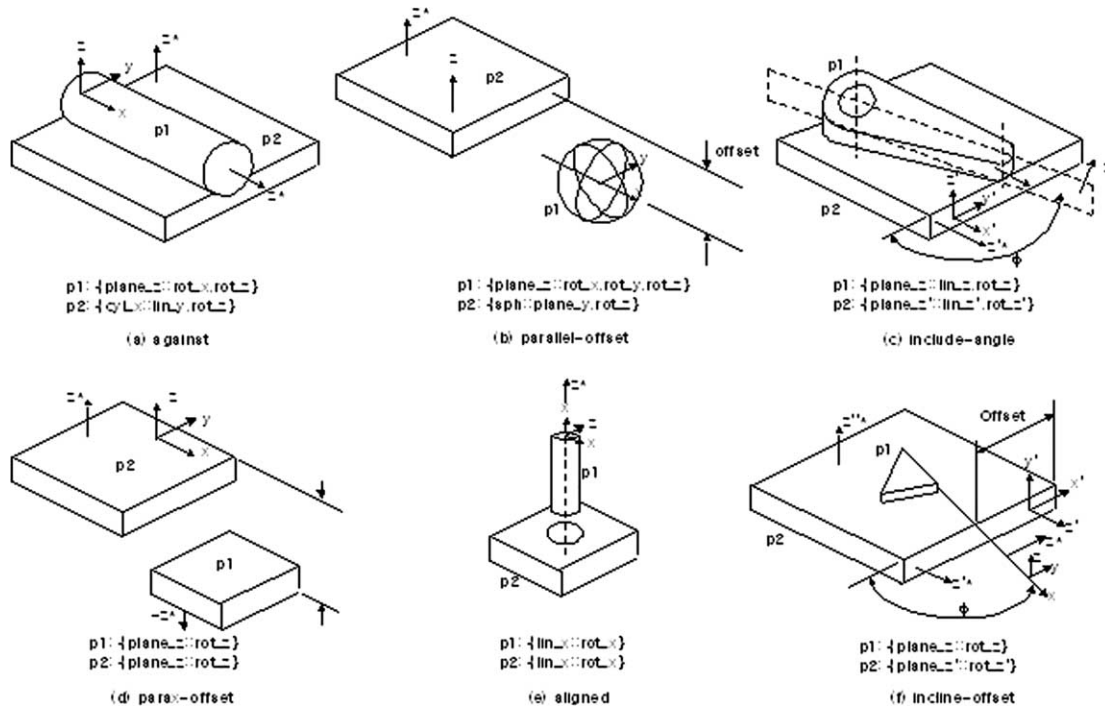


Fig. 4. Six types of spatial relationships.

In this paper, an efficient assembly formalism is developed to represent general assemblies capturing assembly/joining relationship of product assemblies in an AsD model. Design with spatial relationships is the kernel of the assembly component of this assembly formalism.

**3. Assembly design formalism**

An assembly relationship indicates which components are assembled and their joining relationships; the joining relationship denotes how assembly components are joined. For example, two plates, A and B, are assembled by a welding operation. The plates A and B have a joining relationship of welding. To fully describe this assembly, detailed information related to the assembly/joining relationship should be captured in an AsD. The assembly formalism, capturing assembly/joining relationships of a product assembly, comprises of five phases: spatial relationship specification, mating feature extraction, joint feature formation and extraction, AF formation, and assembly engineering relation extraction (Fig. 5). Each of these phases will be described thoroughly in the following sub-sections.

*3.1. Spatial relationships specification*

By interactively assigning spatial relationships, the designer can assemble components together to make final products and infer the d.o.f. remaining on each of the components. The types of d.o.f. were classified as follows

[36];  $lin_n$ : linear translation along  $n$  axis, where,  $n$  contains a fixed point and a vector;  $rot_n$ : rotation about  $n$  axis, where,  $n$  contains a fixed point and a vector;  $cir_n$ : translating along a circle with  $n$  axis, where, the fixed point

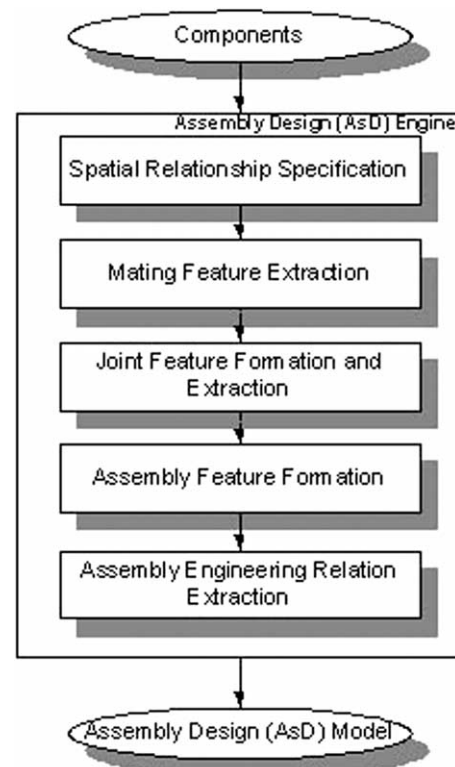


Fig. 5. Procedures of the assembly design formalism.

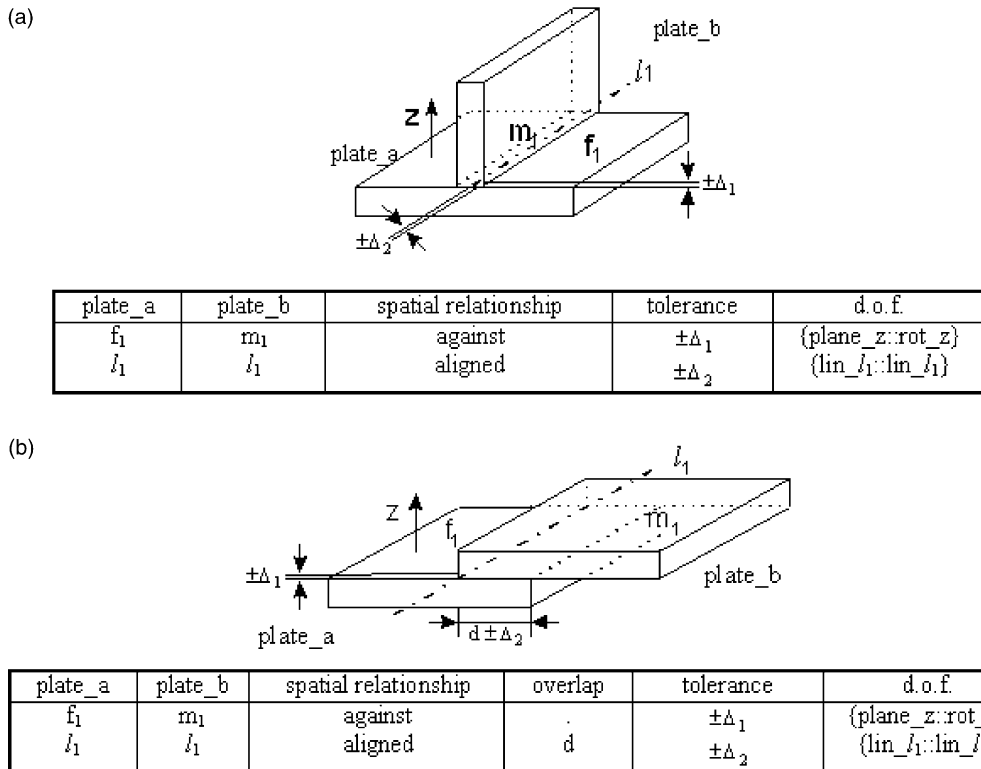


Fig. 6. Examples of assemblies and their spatial relationships. (a) T-joint; (b) lap joint.

of  $n$  is center of the circle and vector of  $n$  is perpendicular to the circle; plane <sub>$n$</sub> , cyl <sub>$n$</sub> , sph: translating along a planar, cylindrical, and spherical surface.

Fig. 6 shows the assembly modeling examples by spatial relationship. Plate <sub>$a$</sub>  and plate <sub>$b$</sub>  represent two plates engaged in spatial relationships. Plate <sub>$b$</sub>  is joined onto a relatively fixed plate <sub>$a$</sub>  and either plate can be a component plate of a sub-assembly.  $f_1$  is a top surface of plate <sub>$a$</sub>  and  $m_1$  is a bottom surface of plate <sub>$b$</sub> .  $l_1$  is an intersecting line. The Tables in Fig. 6(a) and (b) provide the information on the spatial relationships, geometric tolerances ( $\Delta_1$  and  $\Delta_2$ ), and d.o.f. associated with each assembly. In Fig. 6(b) and (d) is a distance being specified for a lap joint. The d.o.f. of a part are expressed as {d.o.f. of moving within the coordinate system of the relative moving part :: d.o.f. of moving within its own coordinate system}, with respect to the other mating parts of the assembly. For a plane <sub>$z_a$</sub>  d.o.f., a body may move on a planar surface along two lin. When a new plane <sub>$z_b$</sub>  is introduced, the remaining d.o.f. is derived by intersecting these two planes. The intersection of surface d.o.f. are available only when a plane is involved, such as, circle <sub>$n$</sub>  is the result of intersecting plane <sub>$z_a$</sub>  with cyl <sub>$n$</sub>  (or sph <sub>$n$</sub> ) together. In the intersection of two rotational d.o.f., say rot <sub>$z_a$</sub>  and rot <sub>$z_b$</sub> , if they share the same rotational axis then rot <sub>$z_a$</sub>  or rot <sub>$z_b$</sub>  remains; if not they cancel each other. Nnaji et al. [36] presented some general reduction rules for d.o.f. In assigning spatial relationships, the mating features are defined and extracted from the parts.

### 3.2. Mating feature extraction

Generally, a feature is a region of interest within a part or an assembly. Features might be considered from the aspect of functionality, manufacturing, inspection, assembly, etc. In other words, features are defined by attaching some sort of attributes according to the user's intention on the design. For assembly modeling, it is necessary to first define and extract the mating features for the operation of product assembly. Usually, two assembly parts do not make contact over their whole surface area; only features of each part are in contact. The mating features are then derived from these features in contact. The definitions of features used in this work are listed below.

1. A *feature* is defined as a set of geometric entities (surfaces, edges, and vertices) together with specifications of the bounding relationship between them and which imply an engineering function on an object [32];
2. A *form feature* is defined as a set of geometric entities (surfaces, edges, and vertices) together with specifications of the bounding relationship between them and which have engineering/functional implications and/or provide assembly aid, such as a center line of a hole, on an object [32,37];
3. A *mating feature* is defined as a set of component geometric entities of form features, which are needed to relatively locate the parts according to their spatial relationships in the whole product assembly.

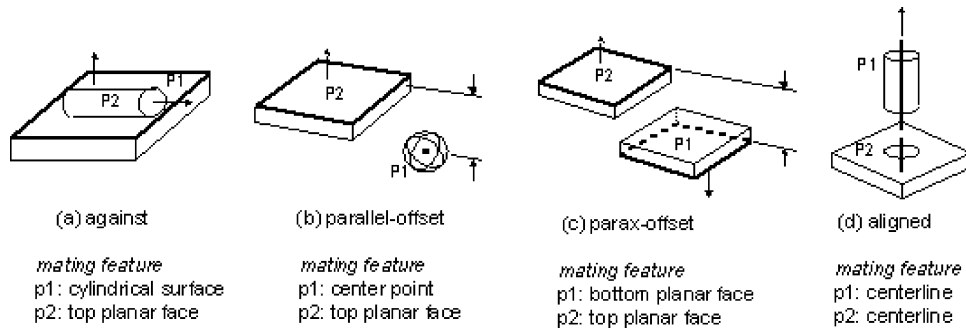


Fig. 7. Spatial relationships and their mating features.

Mating features are very important for representing assembly and joining relationships between assembly components, because actual assembly operations occur at the mating features. Each spatial relationship has specific mating features. For example, when a planar surface of one part is assigned an against spatial relationship with a cylindrical surface of another part, the planar face and the cylindrical surface are the mating features of interest in the assembled pair (Fig. 7(a)). In this fashion, spatial relationships and related mating features provide fundamental elements used to describe assembly. One reason for this is to easily extract mating features universally from parts or predefined features, which are usually diverse, because they are intentoriented. From this point of view, the mating feature of a part could be a planar face, a centerline of a cylinder, an edge of a face boundary, etc. Therefore, the mating features are determined by the types of spatial relationships being specified and the geometric entities being selected (Fig. 7). Mating feature extraction is a preliminary step to capture joining information; however, the mating feature extracted directly from spatial relationship specification is not sufficient to represent joining processes. For example, in Fig. 7(d), the centerline features are not enough to represent the welding operation, since actual weld seams will be around the contact area of the two components. As shown in Fig. 8, the mating feature between  $p_1$  and  $p_2$  should include the cylindrical surfaces.

Unfortunately, detailed joining information cannot be directly captured from spatial relationships and mating features. The next step is a joint feature formation process.

### 3.3. Joint feature formation

Generally, joining processes, such as welding and gluing, happen on the mating entities, such as weld seams. Current mating features of the mating entities have limitations on representing special configurations for joining (e.g. weld seams and grooves). As described above, the mating features of Fig. 7(d) are not enough to represent a joining location (weld seam), a joining method (welding), and groove shapes. In this paper, to enable the description of joining relationships, a new joint feature is defined as:

A set of information including joining methods, groove shapes, joining components and entities, and joining constraints, which is used to represent assembly/joining relations.

The joint feature captures the information of actual joining and it is represented in an AsD model as follows:

$$\{\text{joining method|groove shape|[joining components (joining entities)]|[joining constraint]}\}$$

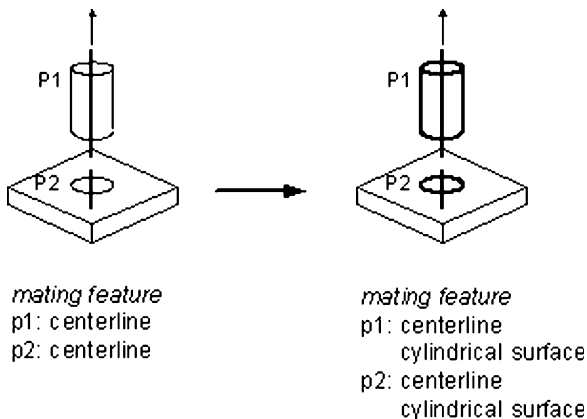


Fig. 8. Mating features expansion of an aligned spatial relationship between a cylinder and a hole.

For example, if a simple t-joint (Fig. 9) is double-fillet welded on two weld seams ( $e_1, e_2$ ) assisted by fixtures located at  $p_1, p_2, \dots, p_8$ , the joint feature can be described as  $\{\text{gas metal arc welding|double fillet|[}fa_1(e_1), fb_1(l_1)\text{], [}fa_1(e_2), fb_1(fb_1)\text{] |[}p_i\text{]}\}$ ,  $i = 1, 2, \dots, 8$ . This joint feature is generated from the input from designer's specifications. The joining entities,  $e_1$  and  $e_2$ , are part of mating features,  $fa_1$  and  $fb_1$ .  $fa_1$  is a bottom surface of plate  $a$  and  $fb_1$  is a top surface of plate  $b$ . The joining entities should be part of the mating features extracted in the previous stage. If a designer specifies a geometric entity, which does not belong to the mating features as a joining entity, then it violates the validity of joining. After joint features are determined, the system is then ready to proceed to the next stage, AF formation.

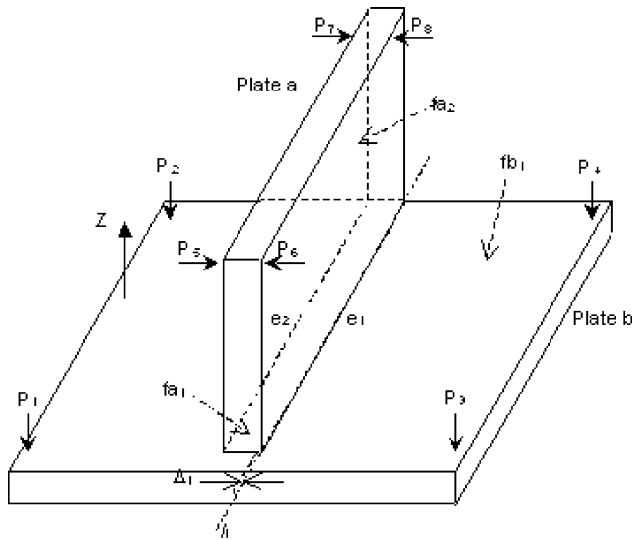


Fig. 9. Example of a welded t-joint.

### 3.4. The assembly feature formation process

The purpose of AF formation is to group the mating features and joint features together and thus integrate the data embedded at the component design stage with new assembly information for subsequent collaborative design processes such as virtual prototyping and simulation, assembly violation detection, process planning, etc. Having designated spatial relationships, mating features, and joint features, the system can then trace back to the component design stage to determine from which form features these mating features originate and what their design specifications are. An assembly feature is defined as:

A group of assembly information including form features and joint features associated with mating relations and assembly/joining relations, such that the association includes a set of spatial relationships between mating features, mating bonds, material, remaining d.o.f., as well as other constraints implied by the original intents on the form features.

It is noted that the mating features provide links between assigning spatial relationships for assembling components and capturing engineering information (e.g. geometry and joining relationships) necessary for the succeeding process, and the AFs act as media to carry and transmit all information to the downstream steps. Different assembly operations imply different information and thus corresponding AFs should be defined for specific assembly operations. For instance, in weld assembly modeling, a weld AF is composed of joint features for welding, mating features, spatial relationships, mating bonds for each spatial relationship, and implied constraints as shown in Table 1. The mating bond is used to capture detailed engineering relations among assembly components and is explained in Section 3.5. Note that the design specifications such as dimensions, positions, and joining methods represent the design constraints at the component level and imply some constraints in mating and joining relations from the viewpoint of assembly and joining. An example of the implied constraints can be found in the case of that the tolerance of pin affects the tolerance between the hole and the pin. After the AF formation steps are completed, assembly engineering relations of an assembly can be easily obtained from the AFs.

### 3.5. Extraction of assembly engineering relations

Assembly engineering relations of the entire assembly can be extracted based on the AFs after specifying the spatial relationships and joining methods between components. A mating bond and a generic assembly relationship diagram (GARD) are used to represent the engineering relationships on the entire structure. The mating bond was originally introduced by Liu and Nnaji [33] for assembly representation. However, the mating bond has a limitation on representing assembly engineering relations, in which joining is considered. In this research, the GARD is introduced to designate assembly engineering relations graphically and the mating bond is extended to pass necessary information to downstream assembly analyses. The mating bond and the GARD provide an efficient design data sharing mechanism in collaborative AsD environment.

Table 1  
Example of assembly feature (for the welded t-joint shown in Fig. 9)

	Plate a		Plate b	
Mating features	$fa_1$ (planar_face)	$l_1$ (face_edge)	$fb_1$ (planar_face)	$l_1$ (face_edge)
Spatial relationships	Against	Aligned	Against	Aligned
Mating bonds	MB1 (against)	MB2 (aligned)	MB1 (against)	MB2 (aligned)
Joint feature		{gas metal arc welding double fillet [ $fa_1(e_1), fb_1(l_1), fa_1(e_2), fb_1(fb_1)$ ]  $[p_i]$ }, $i = 1, 2, \dots, 8$		
Material		Aluminum 6061-T6		
Designed d.o.f.		{plane_z :: rot_z}, {lin_l1 :: lin_l1}		
Implied constraints		Tolerance: $\pm \Delta_1$		

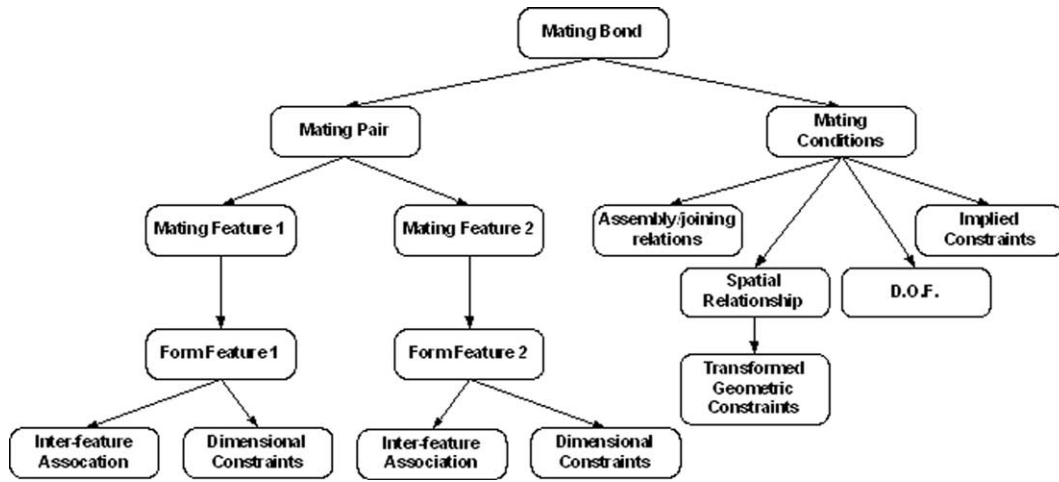


Fig. 10. Mating bonds.

The structure of mating bonds is shown in Fig. 10. There are two dominant groups of information defined in a mating bond: mating pair and mating conditions. The mating pair contains two mating features involved in the joining. The inter-feature association of form features related to assembly is used to record form features, sub-assemblies or assemblies in which the form features associate. From the mating features, the system traces back to its original form features and inherits the implied constraints, e.g. tolerance of the hole. The mating conditions include the assigned spatial relationships, designed d.o.f., and assembly/joining relations. Fig. 11 shows an example of mating bond for aligned spatial relationship specified in a pin assembly.

As mentioned above, assembly engineering relations of an assembly structure can be extracted based on the mating features and joint features after specifying the spatial relationships and joining methods between assembly components. A mating bond is created once two mating features on different components are selected and positioned with each other, and joint features are formatted. AFs are organized by a set of one or more mating bonds.

The concept of assembly engineering relations is shown in Fig. 12. Note that the lines with an arrowhead are interpreted as *belong-to* relations, the lines with solid roundheads as *inter-feature* relations, and the lines with one solid roundhead and one hollow roundhead as *assembly/joining* relations. In other words, each part has been

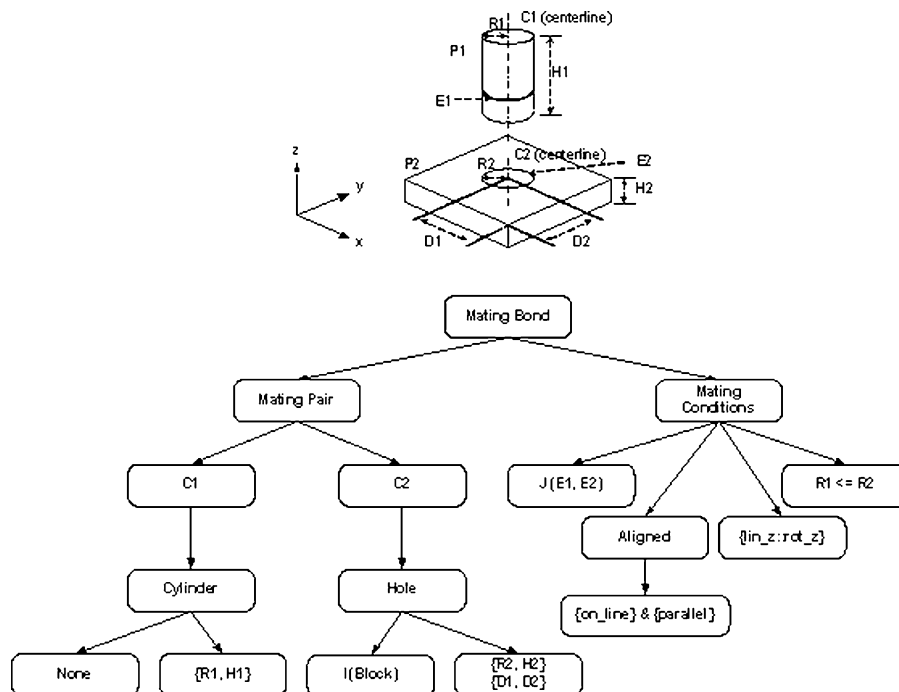


Fig. 11. An example of mating bond for aligned spatial relationship.

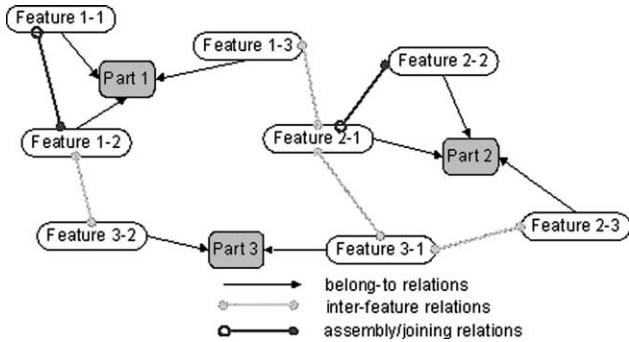


Fig. 12. Assembly engineering relations among assembly components.

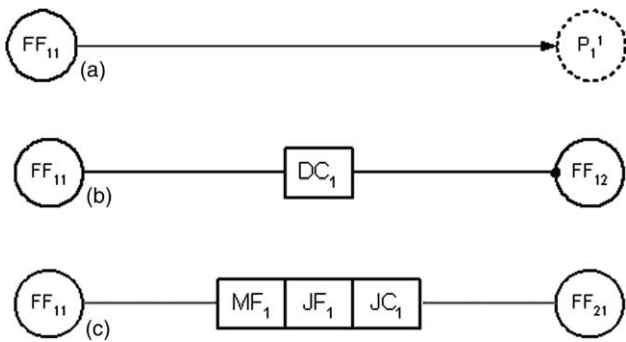


Fig. 13. Relations in GARD. (a) Belong to relations; (b) inter-feature association; (c) assembly/joining relation.

completely designed with its associated features, and functional relationships are built between those part features by attaching some linkages (relations). A GARD is designated to represent graphically this linkage between feature to feature and feature to part. A GARD represents assembly hierarchy and the connectivity of the whole

Table 2  
Symbolic representation of ARM for Fig. 14

Parts	Features and MB	Representation
P <sub>1</sub> <sup>1</sup> & P <sub>2</sub> <sup>1</sup> (pin_a & plate_b)	AF	<ul style="list-style-type: none"> <li>AF<sub>1</sub> = {mating features   mating bonds   joint features   [material]   [designed d.o.f.]   [implied constraints]}</li> <li>= {MF<sub>1</sub>, MF<sub>2</sub>   MB<sub>1</sub>, MB<sub>2</sub>   JF<sub>1</sub>   [Aluminum Alloy 6061 - T6]   {fix}   ±Δ<sub>1</sub>}</li> </ul>
	MF	<ul style="list-style-type: none"> <li>MF<sub>1</sub> = {S/R, [mating components (mating entities)]}</li> <li>= {aligned, [FF<sub>11</sub> (C<sub>1</sub>), FF<sub>22</sub> (C<sub>2</sub>)]}</li> <li>MF<sub>2</sub> = {aligned, [FF<sub>11</sub> (E<sub>1</sub>), FF<sub>22</sub> (E<sub>2</sub>)]}</li> </ul>
	JF	<ul style="list-style-type: none"> <li>JF<sub>1</sub> = {joining method   groove shape   [joining components (joining entities)]   [joining constraint]}</li> <li>= {GMAW   single fillet   [FF<sub>11</sub> (E<sub>1</sub>), FF<sub>22</sub> (E<sub>2</sub>)]   [welding_condition], [fixture_location]}</li> </ul>
	MB	<ul style="list-style-type: none"> <li>MB<sub>1</sub> = {mating pair ( [mating features (form feature (inter-feature association, dimensional constraint))]   mating conditions (assembly/joining relations (form features), S/R (transformed geometric constraints), d.o.f., [implied constraints]) }</li> <li>= {MP<sub>1</sub></li> <li>(MF<sub>1</sub>, [ C<sub>1</sub> (FF<sub>11</sub> ( I (.), {R<sub>1</sub>, H<sub>1</sub>)),</li> <li>C<sub>2</sub> (FF<sub>22</sub> ( I(I<sub>12</sub><sup>2</sup>, RC<sub>12</sub> = 0), {R<sub>2</sub>, H<sub>2</sub>), {D1, D2})))  </li> <li>MC<sub>1</sub></li> <li>(∅<sub>12</sub><sup>(12)</sup>(FF<sub>11</sub>, FF<sub>22</sub>), aligned ({on_line}, {parallel}), {lin_z::rot_z}, [R<sub>1</sub>&lt;=R<sub>2</sub>]))}</li> <li>MB<sub>2</sub></li> <li>= {MP<sub>2</sub></li> <li>(MF<sub>2</sub>, [ E<sub>1</sub> (FF<sub>11</sub> ( I (.), {R<sub>1</sub>, H<sub>1</sub>)),</li> <li>E<sub>2</sub> (FF<sub>22</sub> ( I(I<sub>12</sub><sup>2</sup>, RC<sub>12</sub> = 0), {R<sub>2</sub>, H<sub>2</sub>), {D1, D2})))  </li> <li>MC<sub>2</sub></li> <li>(∅<sub>12</sub><sup>(12)</sup>(FF<sub>11</sub>, FF<sub>22</sub>), aligned ({on_line}, {parallel}), {rot_z}, [R<sub>1</sub>&lt;=R<sub>2</sub>]))}</li> </ul>

product, based on these inter-feature association and assembly/joining relations. Inter-feature associations, assembly/joining relations, and GARDs are explained thoroughly in Section 3.6. Mating bonds capture data structure of assembly engineering relations while GARDs represent these relations diagrammatically.

### 3.6. Assembly relation model and generic assembly relationship diagram

There is a strong need for collaborative AsD systems to communicate and exchange needed design data without transferring whole files from one design collaborator to another. This selective lean information exchange is intended to overcome the bandwidth limitations on Internet/Intranet and to achieve secured relationships among participants. In order to ensure complete transfer of assembly model information during this selective transition, assembly engineering relations should be persistently maintained.

In this research, a new Assembly Relation Model (ARM) including an assembly relationship diagram, GARD is introduced to efficiently capture engineering relations among form features for collaborative design environment. Assembly engineering relations between features as well as between features and parts are defined below.

#### 3.6.1. Definition 1: belong-to relations

A part  $P_j^i$  and a form feature  $FF_{jk}$  are said to have a belong-to relation,

$$B_{jk}^{(i)} : FF_{jk} \rightarrow P_j^i, \quad k = 1, 2, \dots, n,$$

if  $P_j^i \in A_i, j = 1, 2, \dots, m;$  and  $FF_{jk} \in P_j^i.$

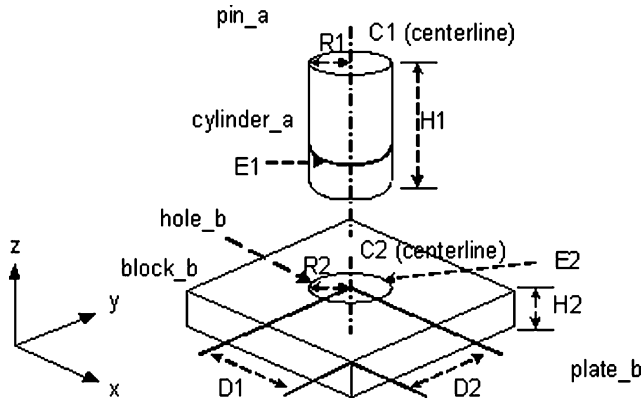


Fig. 14. An assembly with a pin and a plate with hole.

In GARD, a belong-to relation between  $P_j^i$  and  $FF_{jk}$  is represented by an arrow (Fig. 13(a)). Parts are shown in dotted line circle and form features in solid line circle.

3.6.2. Definition 2: inter-feature association relations

A form feature  $FF_{jp}$  and another form feature  $FF_{jq}$  are said to have an inter-feature association relation,

$$I_{pq}^{(j)} : FF_{jp} \Leftrightarrow FF_{jq}, \quad p = 1, 2, \dots, n, \quad q = 1, 2, \dots, l,$$

if  $P_j^i \in \mathbf{A}_i, j = 1, 2, \dots, m; FF_{jp}, FF_{jq} \in \mathbf{FF}; FF_{jp}, FF_{jq} \rightarrow P_j^i; DC_r$  and  $RC_{pq}$  are satisfied, where  $r \in \text{IDI}_{pq}^{(j)}$ ; and  $\text{IDI}_{pq}^{(j)}$  is an index set depending upon this pair,  $FF_{jp}$  and  $FF_{jq}$ .

The inter-feature association relation represents the relations between form features. The relational constraint ( $RC_{pq}$ ) stands for the relationship between two form features in the form feature hierarchy. For example, a block ( $FF_{jq}$ ) can have a blind hole ( $FF_{jp}$ ) at a certain location. The distance between the coordinates of the block and the blind hole is a dimensional constraint. Since the block form feature contains the hole form feature (the block is a parent class of the hole), their relational constraint ( $RC_{pq}$ ) is 0. Fig. 13(b) shows the inter-feature association relation in GARD. The line with square stands for the inter-feature association relation. The square represents a dimensional constraint. Here, the solid dot at the end of line stands for the relational constraint. The circle indicates a form feature associated to the inter-feature association relation. Fig. 13(b) shows a case of ‘ $FF_{11}$  and  $FF_{12}$  have an inter-feature association relation subjected to  $DC_1$  and  $FF_{12}$  contains  $FF_{11}$  ( $FF_{jq} \in FF_{jp}$ )’.

Table 3  
Mathematical representation for Fig. 14

Parts	Assembly engineering relationships
$P_1^1$ and $P_2^1$	$\{FF_{11} \rightarrow P_1^1; FF_{21} \rightarrow P_2^1; FF_{22} \rightarrow P_2^1;$ $FF_{21} \Leftrightarrow FF_{22}   RC_{12} = 0; FF_{11} \otimes FF_{22};$ $\text{IDI}_{12}^{(2)} = \{1\}, \text{JMI}_{12}^{(12)} = \{1, 2\},$ $\text{JJI}_{12}^{(12)} = \{1\}, \text{JCI}_{12}^{(12)} = \{1, 2\}\}$

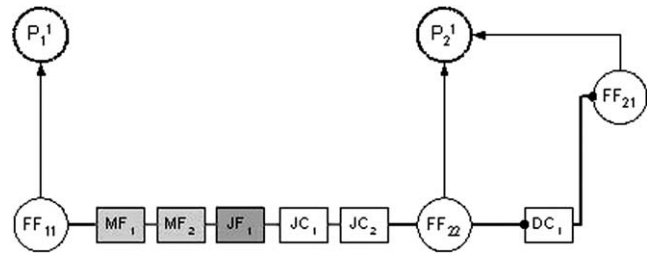


Fig. 15. Pictorial representation for Fig. 14.

3.6.3. Definition 3: assembly/joining relations

A form feature  $FF_{gp}$  and another form feature  $FF_{hq}$  are said to have an assembly/joining relation,

$$\mathcal{D}_{pq}^{(gh)} : FF_{gp} \otimes FF_{hq},$$

if  $P_g^i$  and  $P_h^i \in \mathbf{A}_i, g = 1, 2, \dots, m_1, h = 1, 2, \dots, m_2; FF_{gp} \in P_g^i, p = 1, 2, \dots, l_1; FF_{hq} \in P_h^i, q = 1, 2, \dots, l_2; FF_{gp}, FF_{hq} \in \mathbf{FF}; FF_{gp}, FF_{hq} \in \mathbf{J}; MF_{r1}, JF_{r2} \in \mathbf{J};$  and  $JC_{r3}$  is satisfied, where  $r1 \in \text{JMI}_{pq}^{(gh)}, r2 \in \text{JJI}_{pq}^{(gh)},$  and  $r3 \in \text{JCI}_{pq}^{(gh)}$ ; and  $\text{JMI}_{pq}^{(gh)}, \text{JJI}_{pq}^{(gh)},$  and  $\text{JCI}_{pq}^{(gh)}$  are index sets depending upon this pair,  $FF_{gp}$  and  $FF_{hq}$ .

The assembly/joining relations are represented by lines in GARD. Fig. 13(c) shows an assembly/joining relation between  $FF_{11}$  and  $FF_{21}$ , that is ‘ $FF_{11}$  and  $FF_{21}$  are assembled subjected to  $MF_1, JF_1,$  and  $DC_1$ ’.

3.6.4. Definition 4: generic assembly relationship diagram

Let  $d$  be a GARD of an assembly ( $\mathbf{A}_i$ ) which has a set of parts  $P = \{P_j^i\}$  for  $j = 1, 2, \dots, m$ . Each part  $P_j^i$  has a set of form features  $FF = \{FF_{jk}\}$  for  $k = 1, 2, \dots, n_j$ . There exists a set of belong-to relations,  $R^B = \{B_{jk}^{(i)}\}$ , a set of inter-feature association relations of part  $j, R^I = \{I_{pq}^{(j)}\}$  for  $p = 1, 2, \dots, l_1$  and  $q = 1, 2, \dots, l_2,$  and a set of assembly/joining relations of part  $g$  and part  $h, R^\mathcal{D} = \{\mathcal{D}_{rs}^{(gh)}\}$  for  $g = 1, 2, \dots, m_1,$

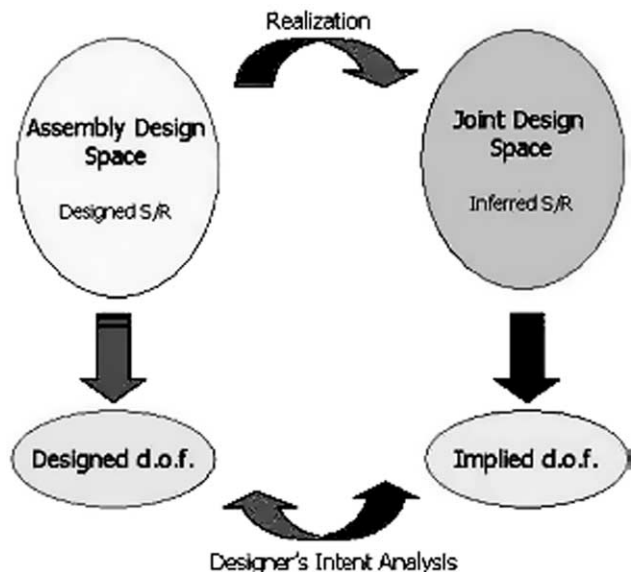


Fig. 16. Spatial relationship implication.

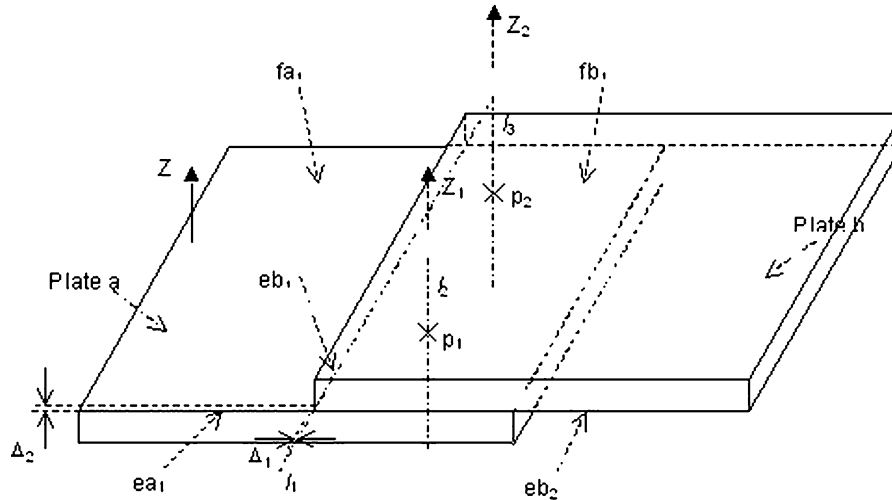


Fig. 17. Lap joint with spatial relationships.

$h = 1, 2, \dots, m_2$ ,  $r = 1, 2, \dots, l_3$ , and  $s = 1, 2, \dots, l_4$ . If  $E_{uv}$  denotes an edge between the nodes  $N_u$  and  $N_v$  of a diagram; then the assembly relationship diagram of  $A_i$  denoted by  $d(A_i)$  is defined by:

$$d(A_i) = (V, E),$$

where

$V = \{V_1, V_2, \dots, V_{m+N}\}$  is the set of nodes in  $d(A_i)$ , where  $N = \sum_j n_j$  and  $E = \{E_{uv}\}$  is the set of edges, where  $u \in \text{FFI}_u, v \in \text{FFI}_v$ ;  $\text{FFI}_u$  and  $\text{FFI}_v$  are index sets depending upon the number of nodes; such that there is a one-to-one correspondence between sets  $V$  and  $P \cup \text{FF}$ ; there is a one-to-one correspondence between sets  $E$  and  $R^B \cup R^I \cup R^D$ .

After AFs are generated, intra-feature and inter-feature relationships are captured in an AsD model. The AsD model contains an ARM connected to a solid model. All geometric entities in an ARM are linked to a related solid model. This ARM goes together with geometric data (solid model) in assembly data transitions, capturing assembly and joining information consistently in a collaborative AsD environment. The ARM can be transformed into three representations (views) (i.e. symbolic, mathematical, and pictorial). The pictorial representation of ARM is generated based upon the GARD. These three views serve as a communication media and help a designer's understanding.

Table 2 shows a symbolic representation of an AsD model (ARM) generated for a simple assembly in Fig. 14. Here,  $P_1^1 = \text{pin}_a$ ;  $P_2^1 = \text{plate}_a$ ;  $\text{FF}_{11} = \text{cylinder}_a$ ;  $\text{FF}_{21} = \text{block}_b$ ;  $\text{FF}_{22} = \text{hole}_b$ . Note that the designed d.o.f. in the AF inferred as {fix}. This d.o.f. is inferred from the specified joining method, that is, gas metal arc welding (GMAW). Spatial relationship implication due to joining is discussed in Section 4. From the AF, two MBs are generated for two aligned spatial relationships. Table 3 shows a mathematical ARM and Fig. 15 shows a pictorial representation of ARM using GARD symbols. In Fig. 15, DC1 (dimensional constraint) of the inter-feature relation is

the location of the hole in the block. JC1 and JC2 are the welding condition and the fixture location, respectively.

#### 4. Spatial relationship implication of joining

Spatial relationships are specified/imposed during the AsD process. As described in the previous sections, each spatial relationship can be interpreted as a constraint imposed on the d.o.f. between relative mating or interacting features. Given a set of spatial relationships, the resultant d.o.f. can be inferred. In other words, any allowable motion for parts has to follow a path along the directions specified by the d.o.f. in order to maintain their spatial relationships.

In AsD, spatial relationships can be assigned to achieve intended d.o.f. These desired spatial relationships are realized and maintained (or enforced) in the physical assembly by joining. Fig. 16 shows how spatial relationships implied by joint design can be used for a designer's intent analysis. Each joining method infers specific spatial relationships and the corresponding d.o.f. are implied by these spatial relationships. The designer's original intent imposed on AsD can be analyzed by comparing the implied d.o.f. and the designed d.o.f. In this manner, all design participants can have an equal understanding of the design intent on an assembly. For example, let us consider a case that a designer wants to permanently join two plates (Fig. 17)

Table 4  
Designed spatial relationships of Fig. 17

Plate a	Plate b	Spatial relationship	Designed d.o.f.
$fa_1$ (top surface)	$fb_1$ (bottom surface)	Against	{plane_z :: rot_z}
$l_1$ (intersecting line)	$eb_1$ (edge)	Aligned	{lin_l1 :: lin_l1}
$ea_1$ (edge)	$eb_2$ (edge)	Aligned	{fixed}

Table 5  
Spatial relationship implication of joining methods

Joining method	Plate <i>a</i>	Plate <i>b</i>	Inferred spatial relationship	Implied d.o.f.
Welding	<i>fa</i> <sub>1</sub> (top surface) <i>l</i> <sub>1</sub> (intersecting line)	<i>fb</i> <sub>1</sub> (bottom surface) <i>eb</i> <sub>1</sub> (edge)	Against Aligned (weld)	{plane_z :: rot_z} {fixed}
One rivet (rivet <i>Q</i> <sub>1</sub> at <i>P</i> <sub>1</sub> )	<i>fa</i> <sub>1</sub> (top surface) <i>l</i> <sub>2</sub> (centerline)	<i>fb</i> <sub>1</sub> (bottom surface) <i>l</i> <sub>2</sub> (centerline)	Against Aligned	{plane_z :: rot_z} {fixed :: rot_z <sub>1</sub> }
Two rivets (rivet <i>Q</i> <sub>1</sub> at <i>P</i> <sub>1</sub> , rivet <i>Q</i> <sub>2</sub> at <i>P</i> <sub>2</sub> )	<i>fa</i> <sub>1</sub> (top surface) <i>l</i> <sub>2</sub> (centerline) <i>l</i> <sub>3</sub> (centerline)	<i>fb</i> <sub>1</sub> (bottom surface) <i>l</i> <sub>2</sub> (centerline) <i>l</i> <sub>3</sub> (centerline)	Against Aligned Aligned	{plane_z :: rot_z} {fixed :: rot_z <sub>1</sub> }

and assigns spatial relationships as in Table 4 to fix those plates. If the designer considers a welded joint and specifies a welding operation as a joining method, then the d.o.f. corresponding to the welding operation can be inferred and used to check whether this welding operation will satisfy the designer’s intent on the assembly. The welding operation causes (1) an *against* spatial relationship between the mating faces, (2) an *aligned* spatial relationship between joining entities on the weld seam, and (3) the two assembly components (two plates) to lose all d.o.f. and become fixed. In this case, the specified joining method (welding) satisfies fully the designed d.o.f.

In other cases, some joining methods may either under-constrain or over-constrain the d.o.f. on an assembly. Consider again the case shown in Fig. 17 with the corresponding designed S/R in Table 4. As shown in Table 5, if a designer wants to join the two plates by applying one cylindrical rivet at *p*<sub>1</sub>, the intended d.o.f. (*fixed*) is under-constrained. In a riveting operation, the end of the rivet shank is deformed after upsetting (Fig. 18). However, after upsetting, the assembly can still have rotational d.o.f., if there is enough tangential (rotational) force applied to

the two plates. Table 6 shows the d.o.f. implication rule when one cylindrical rivet joint is used. When two rivets are used to join the assembly components, the d.o.f. of components are fully constrained based on the reduction rule in Table 7. Note that more than two rivets can increase structural rigidity, even though d.o.f. of the assembly are over-constrained and joining cost and time are increased.

### 5. Demonstration of AsD tools

In this paper, the AsD formalism for a collaborative AsD environment is developed and implemented as a fundamental formalism for AsD tools. An AsD engine generates an AsD model capturing assembly/joining relationships. The AsD model can be used for various AsD activities, such as joining analysis and process planning. A designer can generate an assembly with the AsD engine by specifying spatial relationships, joining methods, weld seam/rivet locations, and joining constraints, such as welding conditions and fixture locations. A GARD tool is used to exchange the AsD model seamlessly in collaborative design environment.

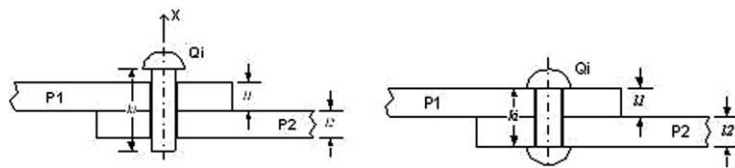


Fig. 18. Riveting operation.

Table 6  
Implied d.o.f. reduction for a rivet joint

Configuration	Condition	Inferred S/R	Implied d.o.f
Two plates	$k_i \leq l_1 + l_2$ $\sum_{i \neq j} j = 0$ where $j = \begin{cases} 1, & \text{for } k_j \geq l_1 + l_2 \\ 0 & \text{otherwise} \end{cases}$  Diameter( <i>Q</i> <sub><i>i</i></sub> ) = diameter(hole_ <i>P</i> <sub>1</sub> ) Diameter( <i>Q</i> <sub><i>i</i></sub> ) = diameter(hole_ <i>P</i> <sub>2</sub> ) Upsetting operation( <i>Q</i> <sub><i>i</i></sub> )	Against(bottom_plane(head_ <i>Q</i> <sub><i>i</i></sub> ), top_plane( <i>P</i> <sub>1</sub> )) Against(bottom_plane( <i>P</i> <sub>1</sub> ), top_plane( <i>P</i> <sub>2</sub> )) Aligned(center_line( <i>Q</i> <sub><i>i</i></sub> ), center_line( <i>P</i> <sub>1</sub> )) Aligned(center_line(hole_ <i>P</i> <sub>1</sub> ), center_line(hole_ <i>P</i> <sub>1</sub> ))	$P_1 : \{fix rot_x\}$ $P_2 : \{fix rot_x\}$

Table 7  
Implied d.o.f. reduction for multiple rivet joints

Configuration	Condition	Inferred S/R	Implied d.o.f
Two plates	$\sum_{i \neq j} j \geq 1$ where $j = \begin{cases} 1, & \text{for } k_j \geq l_1 + l_2 \\ 0, & \text{otherwise} \end{cases}$ $j = 0, 1, \dots, N$ . $N$ is the number of rivets through $P_1$ and $P_2$ Diameter( $Q_j$ ) < diameter(hole $_j$ $_{P_1}$ ) Diameter( $Q_j$ ) < diameter(hole $_j$ $_{P_2}$ ) Upsetting operation( $Q_j$ )	Against(bottom_plane(head_ $Q_j$ ), top_plane( $P_1$ )) Against(bottom_plane( $P_1$ ), top_plane( $P_2$ )) Aligned(center_line $_j$ ( $Q_j$ ), center_line $_j$ ( $P_1$ )) Aligned(center_line $_j$ (hole $_j$ $_{P_1}$ ), center_line $_j$ (hole $_j$ $_{P_2}$ ))	$P_1 : \{\text{fix}\}$ $P_2 : \{\text{fix}\}$

The AsD engine is implemented using Microsoft’s MFC, Spatial’s ACIS, and Tech Soft’s HOOPS. A connector assembly (Fig. 19) is considered as a demonstration of the developed AsD formalism. Fig. 20 shows a graphic user interface of the AsD engine. Fig. 21 shows a data structure of the AsD formalism in terms of UML’s static structure.

The preceding Figures show how a designer generates an assembly with joints and how the AsD engine generates the AsD model. Typical steps are listed as follows:

Step 1:

The designer specifies spatial relationships and corresponding mating features are extracted by the AsD engine (Fig. 22).

Step 2:

The designer determines a joining method and selected corresponding geometric entities on screen. The designer can provide joining conditions. Note that these joining conditions are essential information for succeeding assembly analyses (Fig. 23(a) and (b)).

Step 3:

Spatial relationship implication is inferred and designer’s intent analysis is performed by the AsD engine. If the specified joining method satisfies the desired d.o.f., go to Step 4. Otherwise go to Step 2.

Step 4:

The AsD engine generates additional joint geometry, such as rivets or fasteners based upon joint conditions (Fig. 23(b)). Designer can also determine materials for assembly components in this step.

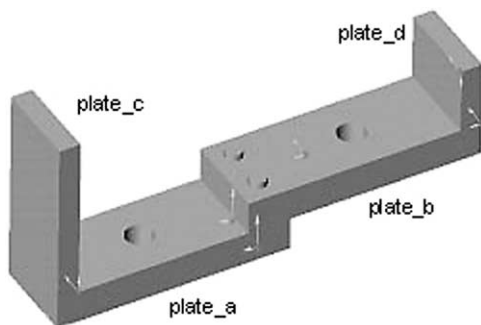


Fig. 19. Connector assembly.

Step 5:

Once the designer provides all information required to form AFs, the AsD engine automatically generates an AsD model including AFs and mating bonds.

Step 6:

For efficient AsD data exchange, the AsD model is generated in a XML format, which is a basic input to generate GARD.

Note that all the information captured in the AsD model supports collaborative AsD. All geometric entities specified in the AsD model data are linked to the solid model. For example, in an ACIS solid model, attribute ID’s are used as a linkage tag. This XML-formatted AsD model data goes together with the geometric data (solid model) in assembly data transitions. It allows assembly and joining information to be persistently captured in a collaborative AsD environment.

While an assembly with joints is formed in the AsD engine, an ARM (AsD model) is generated internally. Table 8 shows the symbolic representation of the ARM in the connector (Fig. 24). Table 9 shows the mathematical representation of the ARM of the connector ( $A_1$ ). In this example, plate\_a and plate\_b are joined by two button rivets. Top\_surface of plate\_a and bottom\_surface of plate\_b have against relationships. The rivets are aligned along centerline of holes at the location that designer specified. plate\_a and plate\_c are joined with GMAW and their mating feature

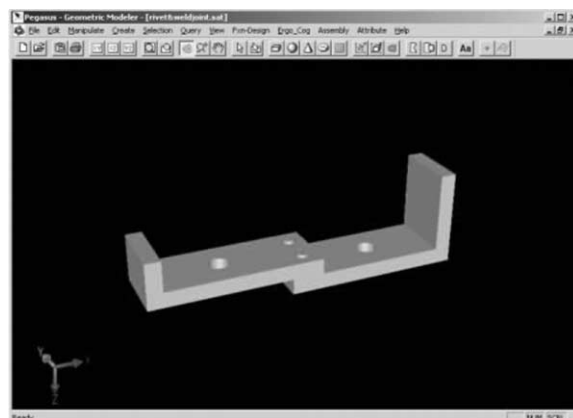


Fig. 20. Graphic user interface of the AsD engine.

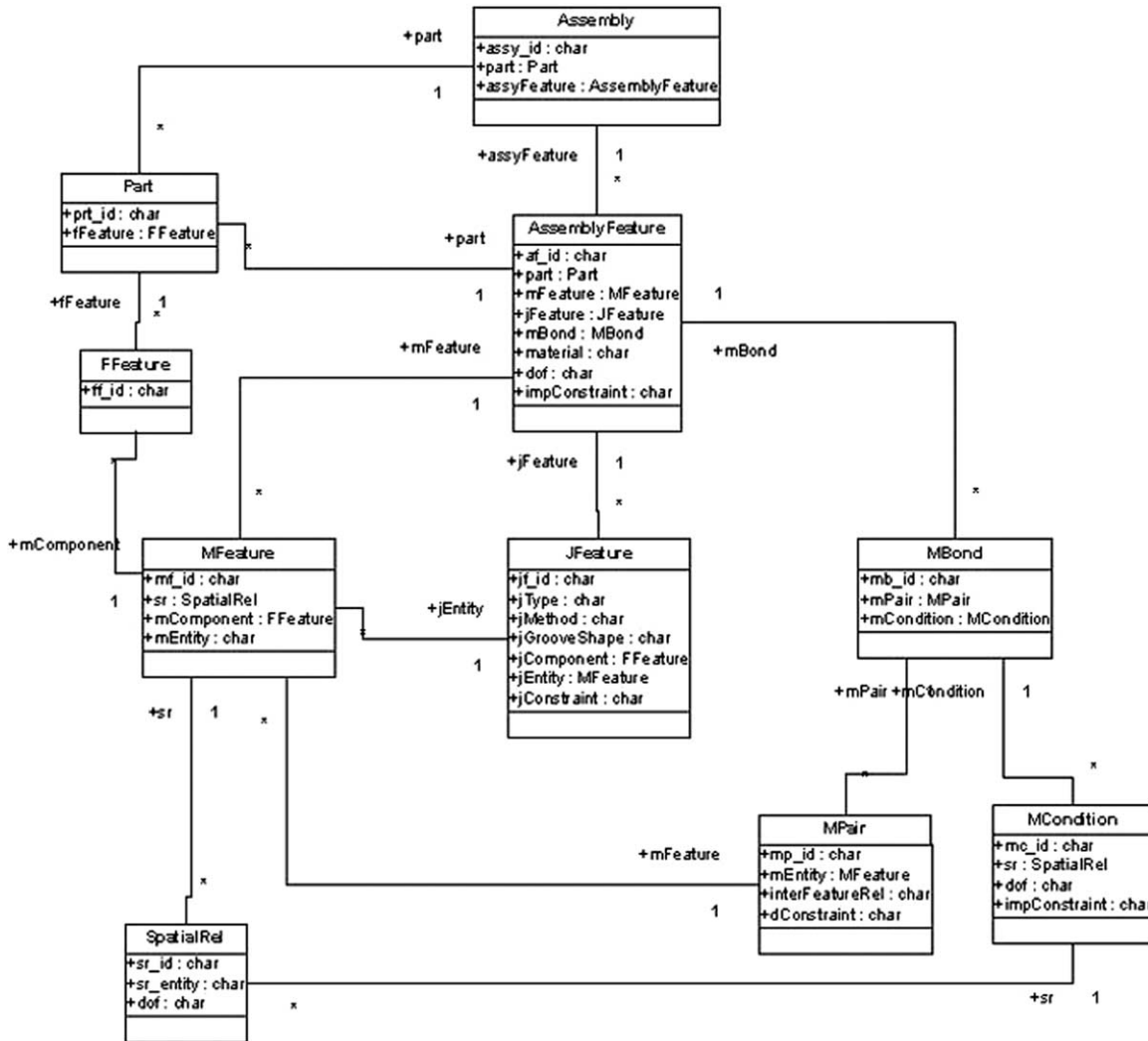


Fig. 21. Data structure of the assembly design formalism.

entities are top\_surface of plate\_a and bottom\_surface of plate\_c. Similarly, plate\_b and plate\_d are joined by using GMAW. In this demonstration, following notations are used:

- $e_i^{jk}$  = *i*th edge of  $FF_{jk}$
- $P_1^1$  = plate\_a;  $P_2^1$  = plate\_b;  $P_3^1$  = plate\_c;  $P_4^1$  = plate\_d;
- $FF_{11}$  = block(length, width, height) = block( $L_{11}$ ,  $W_{11}$ ,  $H_{11}$ ) = block(110, 40, 10);
- $FF_{21}$  = block( $L_{21}$ ,  $W_{21}$ ,  $H_{21}$ ) = block(110, 40, 10);
- $FF_{31}$  = block( $L_{31}$ ,  $W_{31}$ ,  $H_{31}$ ) = block(50, 40, 10);
- $FF_{41}$  = block( $L_{41}$ ,  $W_{41}$ ,  $H_{41}$ ) = block(20, 40, 10);
- $FF_{12}$  = hole(diameter, depth) = hole( $DM_{12}$ ,  $DT_{12}$ ) = hole(12.81, 10);
- $FF_{22}$  = hole( $DM_{22}$ ,  $DT_{22}$ ) = hole(12.81, 10);
- JC<sub>1</sub> (of  $FF_{11}$  and  $FF_{21}$ ) = {location of rivets|tolerance} = { $P11(100, 10, .), P11(100, 40, .) \pm \Delta 1$ };

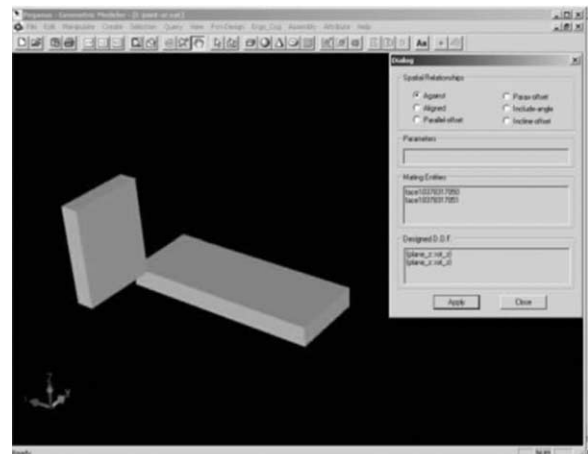


Fig. 22. Spatial relationship specification and mating feature extraction.

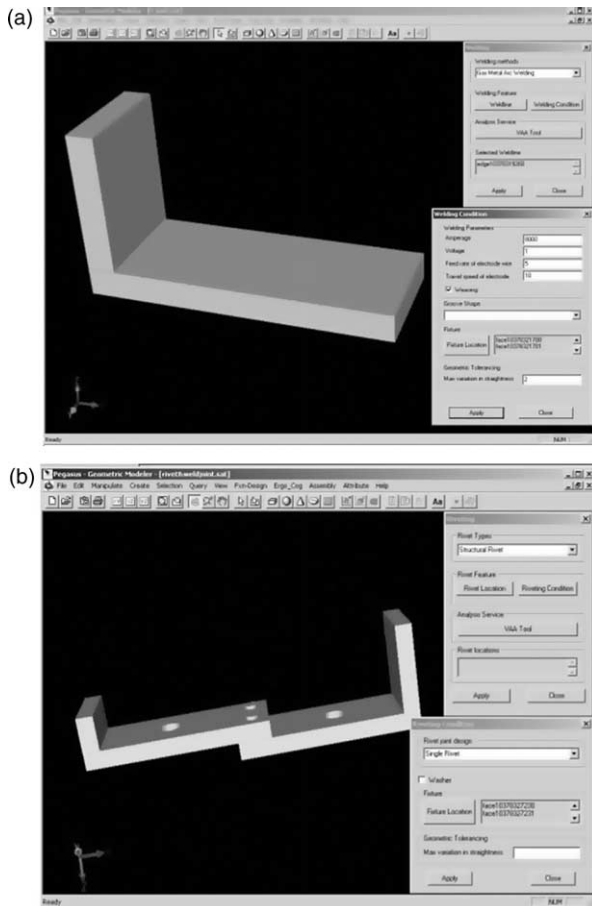


Fig. 23. Joining method specification and joint feature formation. (a) Welding; (b) riveting.

$JC_2$  (of  $FF_{11}$  and  $FF_{21}$ ),  $JC_4$  (of  $FF_{11}$  and  $FF_{31}$ ), and  $JC_7$  (of  $FF_{21}$  and  $FF_{41}$ ) = fixture location;  
 $JC_3$  (of  $FF_{11}$  and  $FF_{31}$ ) and  $JC_6$  (of  $FF_{21}$  and  $FF_{41}$ ) = welding condition;  
 $JC_5$  (of  $FF_{11}$  and  $FF_{31}$ ) = {datum planes|tolerance } = {max\_displacement|  $\pm \Delta_2$ };  
 $JC_8$  (of  $FF_{21}$  and  $FF_{41}$ ) = {max\_displacement|  $\pm \Delta_3$  };  
 $DC_1$  (of  $FF_{11}$  and  $FF_{12}$ ) = {location of hole|tolerance } = { $P11(50, 20, \cdot)$ |  $\pm \Delta_4$ };  
 $DC_2$  (of  $FF_{21}$  and  $FF_{22}$ ) = { $P_2^1(50, 20, \cdot)$ |  $\pm \Delta_5$ }.

Datum planes in  $JC_5$  and  $JC_8$  are reference planes that represent the designated tolerance limits of welding deformation. The datum planes are offset from the structure by the tolerance limits allowed.

Based upon the AsD model, the corresponding GARD can be generated with the aid of a developed GARD tool to help the designer to easily understand the assembly and joining relationships in an assembly. The GARD tool is implemented using Microsoft's Visio and Visual Basic for Applications. Fig. 25(a) shows an initial window of the GARD tool. The designer can open the AsD data with this tool and generate a GARD corresponding to the AsD model. Fig. 25(b) shows a GARD for the connector assembly.

By clicking each diagram's entities, relevant assembly/joining information and design model can be retrieved locally or remotely through the service-oriented architecture. This GARD tool provides user-friendly communication media in a collaborative AsD.

## 6. AsD formalism and AsD tools in a service-oriented collaborative assembly design

This section explains how the developed AsD formalism and the AsD tools can be integrated into a service-oriented collaborative AsD environment. A typical scenario is when a system integrator, such as an auto manufacturer, out-sources the design and manufacturing of sub-systems, such as car frames from different vendors. If the auto manufacturer wants to design a complete car frame with sub-frames designed by vendors A and B, the vendors provide an XML-formatted AsD model, which are simple ASCII files, of the sub-frames to the auto-manufacturer instead of sending the entire CAD model. The system integrator can easily generate GARDs from the AsD model. The GARDs are linked to the corresponding design models of sub-system components. These design models in certain proprietary CAD formats are translated into CAD kernel formats, such as SAT or ACIS, as soon as the vendors send the AsD model to the system integrator. The design models in the CAD kernel format can be provided to the system integrator when the system integrator indicates a request for viewing a specific component from the GARD tool. The auto-manufacturer therefore can decide the assembly components to be joined. The determined assembly components are loaded into the AsD engine and the system integrator can specify a joining method between assembly components. The new AsD model and its corresponding GARD are generated automatically based upon the AsD formalism.

During collaborative AsD, AsD participants typically use different CAD systems. To generate a complete assembly, each design model needs to be translated into a single CAD format, which can be accomplished by using specialized translators. However, this often causes problems in the numerical accuracy of geometric model, since different CAD systems employ different methods of CAD model generation. The first solution of this is to provide a modeler that has the ability to process models created in different CAD formats. The second solution is to use solid modeling kernels. Typically, a CAD system is built on a solid modeling kernel. Based upon the solid modeling kernel, suitable interfaces and high-level operations, such as feature based modeling and editing, of the CAD system are implemented. It is to be noted that while there are a large number of proprietary CAD formats, there are relatively few solid modeling kernels that are available, such as ACIS and Parasolid. This AsD engine utilizes the second method and it is implemented with ACIS kernel.

Table 8  
Symbolic representation for the connector in Fig. 24

Parts	Features and MB	Representation
P <sub>1</sub> <sup>1</sup> & P <sub>2</sub> <sup>1</sup> (plate_a & plate_b)	AF	<ul style="list-style-type: none"> <li>AF<sub>1</sub> = {mating features   mating bonds   joint features   [material]   [designed d.o.f.]   [implied constraints]} = { MF<sub>1</sub>, MF<sub>2</sub>, MF<sub>3</sub>   MB<sub>1</sub>, MB<sub>2</sub>, MB<sub>3</sub>   JF<sub>1</sub>   [Aluminum Alloy 6061 - T6, Steel]   {fix}   [tolerance]} </li> </ul>
	MF	<ul style="list-style-type: none"> <li>MF<sub>1</sub> = {S/R, [mating components (mating entities)]} = {against, [FF<sub>11</sub> (top_surface), FF<sub>21</sub> (bottom_surface)]}</li> <li>MF<sub>2</sub> = {aligned, [FF<sub>11</sub> (l<sub>1</sub>), FF<sub>21</sub> (e<sub>2</sub><sup>21</sup>)]}</li> <li>MF<sub>3</sub> = {aligned, [FF<sub>11</sub> (e<sub>1</sub><sup>11</sup>), FF<sub>21</sub> (e<sub>1</sub><sup>21</sup>)]}</li> </ul>
	JF	<ul style="list-style-type: none"> <li>JF<sub>1</sub> = {joining method   groove shape   [joining components (joining entities)]   [joining constraint]} = {Button_Rivet   .   [FF<sub>11</sub> (top_surface), FF<sub>21</sub> (bottom_surface)]   [diameter_of_rivet, 5.66], [location_of_rivet, FF<sub>11</sub> (100, 10, .), FF<sub>11</sub> (100, 40, .)], [fixture_location]}</li> </ul>
	MB	<ul style="list-style-type: none"> <li>MB<sub>1</sub> = {mating pair ( [mating features (form feature (inter-feature association, dimensional constraint))]   mating conditions (assembly/joining relations (form features), S/R (transformed geometric constraints), d.o.f., [implied constraints]) } = {MP<sub>1</sub> (MF<sub>1</sub>, [ top_surface (FF<sub>11</sub> ( l (.), {L<sub>11</sub>, W<sub>11</sub>, H<sub>11</sub>)}), bottom_surface (FF<sub>21</sub> ( l (.), {L<sub>21</sub>, W<sub>21</sub>, H<sub>21</sub>)}))   MC<sub>1</sub> (∂<sub>11</sub><sup>(12)</sup>(FF<sub>11</sub>, FF<sub>21</sub>), against ({on_surface}), {plane_z::rot_z}, [ . ])) } MB<sub>2</sub> = {MP<sub>2</sub> (MF<sub>2</sub>, [ l<sub>1</sub> (FF<sub>11</sub> ( l (.), {L<sub>11</sub>, W<sub>11</sub>, H<sub>11</sub>)}), e<sub>2</sub><sup>21</sup> (FF<sub>21</sub> ( l (.), {L<sub>21</sub>, W<sub>21</sub>, H<sub>21</sub>)}))   MC<sub>2</sub> (∂<sub>11</sub><sup>(12)</sup>( FF<sub>11</sub>, FF<sub>21</sub>), aligned ({on_line} &amp; {parallel}), {lin_l<sub>1</sub>::lin_l<sub>1</sub>}, [ . ])) } MB<sub>3</sub> = {MP<sub>3</sub> (MF<sub>3</sub>, [e<sub>1</sub><sup>11</sup> (FF<sub>11</sub> ( l (.), {L<sub>11</sub>, W<sub>11</sub>, H<sub>11</sub>)}), e<sub>1</sub><sup>21</sup> (FF<sub>21</sub> ( l (.), {L<sub>21</sub>, W<sub>21</sub>, H<sub>21</sub>)}))   MC<sub>3</sub> (∂<sub>11</sub><sup>(12)</sup>( FF<sub>11</sub>, FF<sub>21</sub>), aligned ({on_line} &amp; {parallel}), {lin_e<sub>1</sub><sup>11</sup>::lin_e<sub>1</sub><sup>11</sup>}, [ . ])) }</li> </ul>
P <sub>1</sub> <sup>1</sup> & P <sub>3</sub> <sup>1</sup> (plate_a & plate_c)	AF	<ul style="list-style-type: none"> <li>AF<sub>2</sub> = {mating features   mating bonds   joint features   [material]   [designed d.o.f.]   [implied constraints]} = {MF<sub>4</sub>, MF<sub>5</sub>, MF<sub>6</sub>   MB<sub>4</sub>, MB<sub>5</sub>, MB<sub>6</sub>   JF<sub>2</sub>   [Aluminum Alloy 6061 - T6]   {fix}   [tolerance]} </li> </ul>
	MF	<ul style="list-style-type: none"> <li>MF<sub>4</sub> = {S/R, [mating components (mating entities)]} = {against, [FF<sub>11</sub> (top_surface), FF<sub>31</sub> (bottom_surface)]}</li> <li>MF<sub>5</sub> = {aligned, [FF<sub>11</sub> (l<sub>2</sub>), FF<sub>31</sub> (e<sub>2</sub><sup>31</sup>)]}</li> <li>MF<sub>6</sub> = {aligned, [FF<sub>11</sub> (e<sub>1</sub><sup>12</sup>), FF<sub>31</sub> (e<sub>1</sub><sup>31</sup>)]}</li> </ul>
	JF	<ul style="list-style-type: none"> <li>JF<sub>2</sub> = {joining method   groove shape   [joining components (joining entities)]   [joining constraint]} = {GMAW   single fillet   [FF<sub>11</sub> (l<sub>2</sub>), FF<sub>31</sub> (e<sub>2</sub><sup>31</sup>)]   [welding_condition], [fixture_location]}</li> </ul>
	MB	<ul style="list-style-type: none"> <li>MB<sub>4</sub> = {mating pair ( [mating features (form feature (inter-feature association, dimensional constraint))]   mating conditions (assembly/joining relations (form features), S/R (transformed geometric constraints), d.o.f., [implied constraints]) } = {MP<sub>4</sub> (MF<sub>4</sub>, [ top_surface (FF<sub>11</sub> ( l (.), {L<sub>11</sub>, W<sub>11</sub>, H<sub>11</sub>)}), bottom_surface (FF<sub>31</sub> ( l (.), {L<sub>31</sub>, W<sub>31</sub>, H<sub>31</sub>)}))   MC<sub>4</sub> (∂<sub>11</sub><sup>(13)</sup>( FF<sub>11</sub>, FF<sub>31</sub>), against ({on_surface}), {plane_z::rot_z}, [ . ])) } MB<sub>5</sub> = {MP<sub>5</sub> (MF<sub>5</sub>, [ l<sub>2</sub> (FF<sub>11</sub> ( l (.), {L<sub>11</sub>, W<sub>11</sub>, H<sub>11</sub>)}), e<sub>2</sub><sup>31</sup> (FF<sub>31</sub> ( l (.), {L<sub>31</sub>, W<sub>31</sub>, H<sub>31</sub>)}))   MC<sub>5</sub> (∂<sub>11</sub><sup>(13)</sup>( FF<sub>11</sub>, FF<sub>31</sub>), aligned ({on_line} &amp; {parallel}), {lin_l<sub>2</sub>::lin_l<sub>2</sub>}, [ . ])) }</li> </ul>

Table 8 (continued)

		<ul style="list-style-type: none"> <li>MB<sub>6</sub> = {MP<sub>6</sub> (MF<sub>6</sub>, [e<sub>1</sub><sup>12</sup> (FF<sub>11</sub> ( l (.), {L<sub>11</sub>, W<sub>11</sub>, H<sub>11</sub>))), e<sub>1</sub><sup>31</sup> (FF<sub>31</sub> ( l (.), {L<sub>31</sub>, W<sub>31</sub>, H<sub>31</sub>)))] )   MC<sub>6</sub> (∂<sub>17</sub><sup>(13)</sup>( FF<sub>11</sub>, FF<sub>31</sub>), aligned ({on_line} &amp; {parallel}), {lin_e<sub>1</sub><sup>12</sup>::lin_e<sub>1</sub><sup>12</sup>, [ . ]))</li> </ul>
P <sub>2</sub> <sup>1</sup> & P <sub>4</sub> <sup>1</sup> (plate_b & plate_d)	AF	<ul style="list-style-type: none"> <li>AF<sub>3</sub> = {mating features   mating bonds   joint features   [material]   [designed d.o.f.]   [implied constraints]} = {MF<sub>7</sub>, MF<sub>8</sub>, MF<sub>9</sub>   MB<sub>7</sub>, MB<sub>8</sub>, MB<sub>9</sub>   JF<sub>3</sub>   [Aluminum Alloy 6061 - T6]   {fix}   [tolerance]}</li> </ul>
	MF	<ul style="list-style-type: none"> <li>MF<sub>7</sub> = {S/R, [mating components (mating entities)] = {against, [FF<sub>21</sub> (top_surface), FF<sub>41</sub> (bottom_surface)]}</li> <li>MF<sub>8</sub> = {aligned, [FF<sub>21</sub> (l<sub>4</sub>), FF<sub>41</sub> (e<sub>2</sub><sup>41</sup>)]}</li> <li>MF<sub>9</sub> = {aligned, [FF<sub>21</sub> (e<sub>3</sub><sup>21</sup>), FF<sub>41</sub> (e<sub>1</sub><sup>41</sup>)]}</li> </ul>
	JF	<ul style="list-style-type: none"> <li>JF<sub>3</sub> = {joining method   groove shape   [joining components (joining entities)]   [joining constraint]} = {GMAW   single fillet   [FF<sub>21</sub> (l<sub>4</sub>), FF<sub>41</sub> (e<sub>2</sub><sup>41</sup>)]   [welding_condition], [fixture location]}</li> </ul>
	MB	<ul style="list-style-type: none"> <li>MB<sub>7</sub> = {mating pair ( [mating features (form feature (inter-feature association, dimensional constraint))]   mating conditions (assembly/joining relations (form features), S/R (transformed geometric constraints), d.o.f., [implied constraints]) } = {MP<sub>7</sub> (MF<sub>7</sub>, [ top_surface (FF<sub>21</sub> ( l (.), {L<sub>21</sub>, W<sub>21</sub>, H<sub>21</sub>))), bottom_surface (FF<sub>41</sub> ( l (.), {L<sub>41</sub>, W<sub>41</sub>, H<sub>41</sub>)))] )   MC<sub>7</sub> (∂<sub>17</sub><sup>(24)</sup>( FF<sub>21</sub>, FF<sub>41</sub>), against ({on_surface}), {plane_z::rot_z}, [ . ]))</li> <li>MB<sub>8</sub> = {MP<sub>8</sub> (MF<sub>8</sub>, [ l<sub>4</sub> (FF<sub>21</sub> ( l (.), {L<sub>21</sub>, W<sub>21</sub>, H<sub>21</sub>))), e<sub>2</sub><sup>41</sup> (FF<sub>41</sub> ( l (.), {L<sub>41</sub>, W<sub>41</sub>, H<sub>41</sub>)))] )   MC<sub>8</sub> (∂<sub>17</sub><sup>(24)</sup>( FF<sub>21</sub>, FF<sub>41</sub>), aligned ({on_line} &amp; {parallel}), {lin_l<sub>4</sub>::lin_l<sub>4</sub>, [ . ]))</li> <li>MB<sub>9</sub> = {MP<sub>9</sub> (MF<sub>9</sub>, [e<sub>3</sub><sup>21</sup> (FF<sub>21</sub> ( l (.), {L<sub>21</sub>, W<sub>21</sub>, H<sub>21</sub>))), e<sub>1</sub><sup>41</sup> (FF<sub>41</sub> ( l (.), {L<sub>41</sub>, W<sub>41</sub>, H<sub>41</sub>)))] )   MC<sub>9</sub> (∂<sub>17</sub><sup>(24)</sup>( FF<sub>21</sub>, FF<sub>41</sub>), aligned ({on_line} &amp; {parallel}), {lin_e<sub>3</sub><sup>21</sup>::lin_e<sub>3</sub><sup>21</sup>, [ . ]))</li> </ul>

Fig. 26 shows how AsD collaborators (e-designers) can share AsD models interacting with different CAD systems in a service-oriented collaborative AsD environment. Consider a system integrator who wants to assemble two components designed by vendor 1 and vendor 2. Through the service-oriented architecture, AsD models of assembly components can be provided remotely to the system integrator and the system integrator can generate an

assembly without receiving entire CAD model. In case the vendors' CAD systems provide different CAD kernels, a multi-kernel agent is responsible for consistent model translation. Detailed processes are described below. The numbers in Fig. 26 stand for the index of each process.

Table 9  
Mathematical representation for the connector in Fig. 24

Parts	Assembly relationships
P <sub>1</sub> <sup>1</sup> and P <sub>2</sub> <sup>1</sup>	{FF <sub>11</sub> : →P <sub>1</sub> <sup>1</sup> ; FF <sub>12</sub> : →P <sub>1</sub> <sup>1</sup> ; FF <sub>21</sub> : →P <sub>2</sub> <sup>1</sup> ; FF <sub>22</sub> : →P <sub>2</sub> <sup>1</sup> ; FF <sub>11</sub> ⇔ FF <sub>12</sub>   RC <sub>12</sub> = 0; FF <sub>21</sub> ⇔ FF <sub>22</sub>   RC <sub>12</sub> = 0; FF <sub>11</sub> ⊗ FF <sub>21</sub> ; IDI <sub>12</sub> <sup>(1)</sup> = {1}, IDI <sub>12</sub> <sup>(2)</sup> = {2}, JMI <sub>11</sub> <sup>(12)</sup> = {1, 2, 3}, JJI <sub>11</sub> <sup>(12)</sup> = {1}, JCI <sub>11</sub> <sup>(12)</sup> = {1, 2}}
P <sub>1</sub> <sup>1</sup> and P <sub>3</sub> <sup>1</sup>	{FF <sub>11</sub> : →P <sub>1</sub> <sup>1</sup> ; FF <sub>12</sub> : →P <sub>1</sub> <sup>1</sup> ; FF <sub>31</sub> : →P <sub>3</sub> <sup>1</sup> ; FF <sub>11</sub> ⇔ FF <sub>12</sub>   RC <sub>12</sub> = 0; FF <sub>11</sub> ⊗ FF <sub>31</sub> ; IDI <sub>12</sub> <sup>(1)</sup> = {4}, JMI <sub>11</sub> <sup>(13)</sup> = {4, 5, 6}, JJI <sub>11</sub> <sup>(13)</sup> = {2}, JCI <sub>11</sub> <sup>(13)</sup> = {3, 4, 5}}
P <sub>2</sub> <sup>1</sup> and P <sub>4</sub> <sup>1</sup>	{FF <sub>21</sub> : →P <sub>2</sub> <sup>1</sup> ; FF <sub>22</sub> : →P <sub>2</sub> <sup>1</sup> ; FF <sub>41</sub> : →P <sub>4</sub> <sup>1</sup> ; FF <sub>21</sub> ⇔ FF <sub>22</sub>   RC <sub>12</sub> = 0; FF <sub>21</sub> ⊗ FF <sub>41</sub> ; IDI <sub>12</sub> <sup>(2)</sup> = {5}, JMI <sub>11</sub> <sup>(24)</sup> = {7, 8, 9}, JJI <sub>11</sub> <sup>(24)</sup> = {3}, JCI <sub>11</sub> <sup>(24)</sup> = {6, 7, 8}}

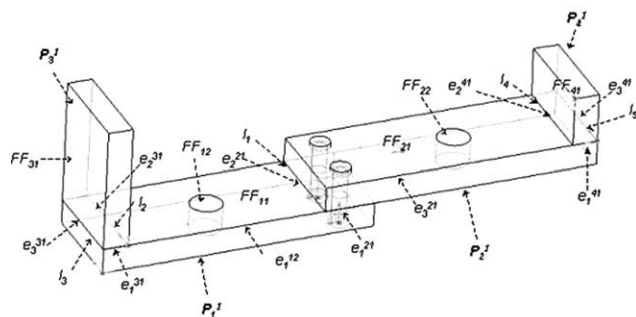


Fig. 24. Connector assembly with two welded joints and one pin joint.

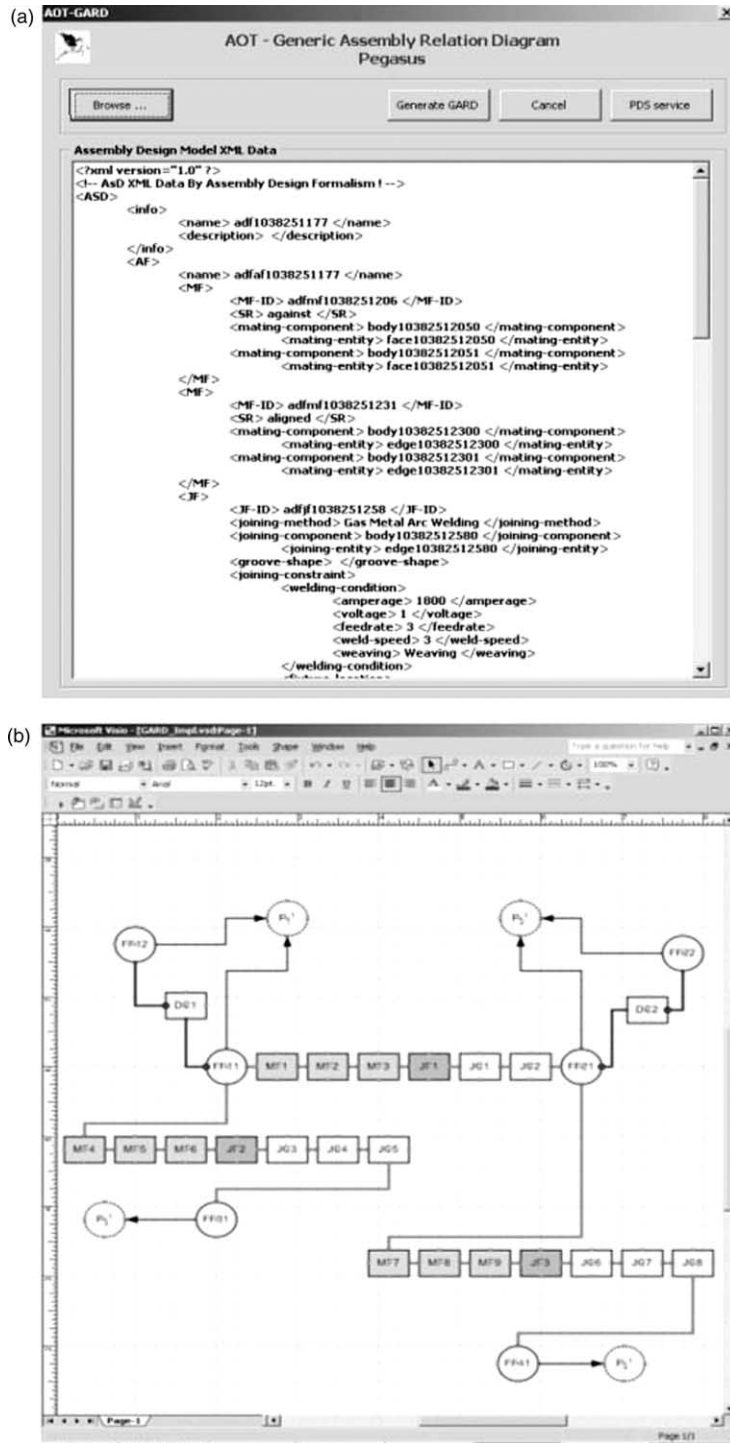


Fig. 25. GARD tool. (a) Initial interface; (b) GARD for the connector assembly.

Step 1:

A system integrator requests AsD models of sub-assemblies/components interested ①.

Step 2:

Vendors provide requested AsD models in XML format to system integrator, while the corresponding CAD models are translated to the CAD kernel model and stored in the local

database of each vendors ①. If the vendor doesn't have the capability to translate the CAD model to the kernel model, a third-party multi-kernel agent can be employed ②.

Step 3:

The system integrator reviews sub-assemblies/components with the aid of the GARD tool and a product viewer ③, ④, and ⑥. According to the system integrators'

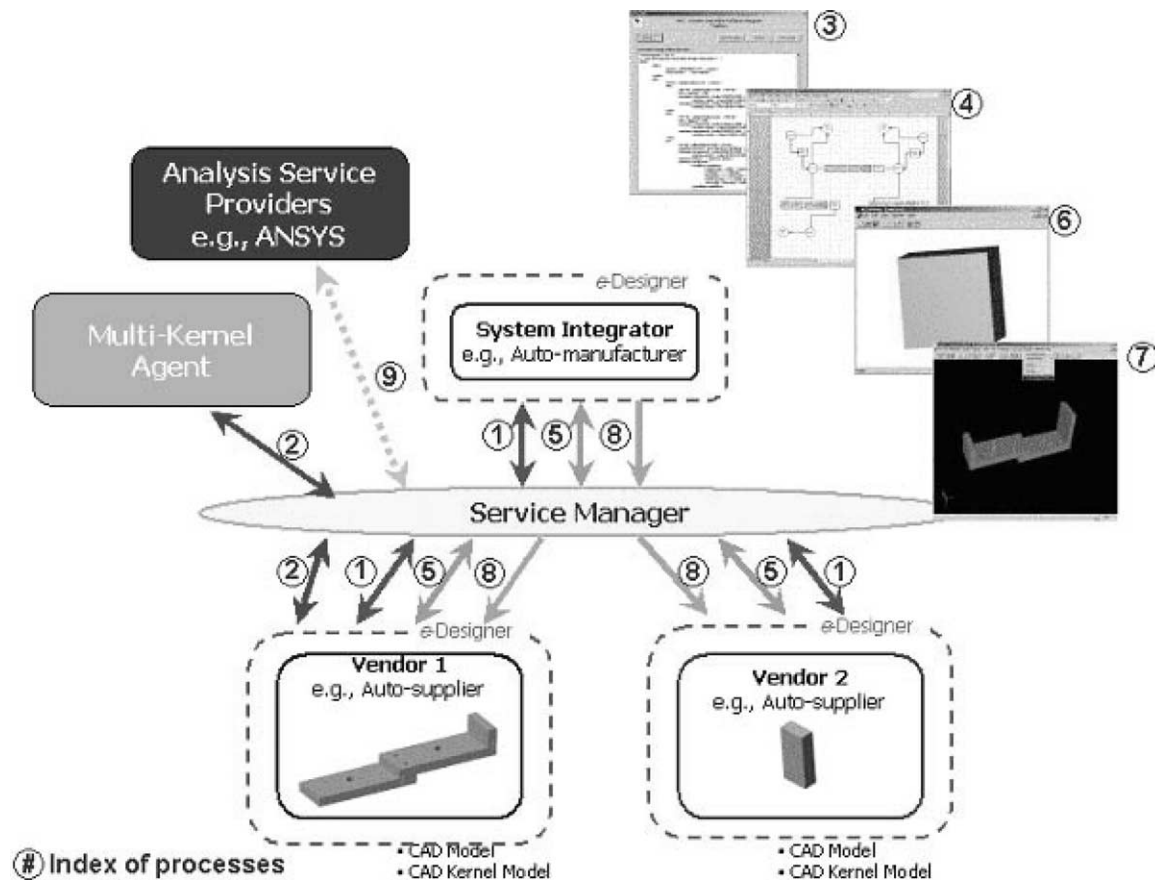


Fig. 26. AsD tools in a service-oriented collaborative assembly design.

needs, the GARD tool can selectively retrieve necessary parts in kernel format from the vendors' database (5).

Step 4:

After determining which sub-assemblies/components are to be joined, the system integrator can load kernel models of selected individual parts into the AsD engine and specify joining methods between the parts (7).

Step 5:

An AsD model for the new assembly is generated based upon the AsD Formalism (7). The new AsD model can be sent to the vendors to share assembly information (8).

Step 6:

When the system integrator needs to know additional AsD information, such as the physical effects of joining, the system integrator can request relevant service using the AsD models (9).

## 7. Conclusions

This research introduces an AsD formalism to specify the assembly and joining relations symbolically to support collaborative AsD. By using the AsD formalism, assembly and joining relations are extracted from the assembly and the relation models have mathematically solvable

implications. A spatial relationship kernel preserves design intent on the assembly. The spatial relationship implication is inferred to validate the specified joining method that satisfies the designer's intent. AsD tools are developed to implement the AsD formalism, which leverage an efficient assembly data sharing mechanism and transparent assembly information flow for a collaborative AsD environment. An AsD model generated by the AsD engine contains an ARM linked to a solid model. The AsD model supplements geometric and topological information of nominal geometry with assembly/joining information, which is essential for various AsD activities, such as joining analysis, process planning, and integrated simulation. ARM has three views (i.e. symbolic, mathematical, and pictorial views). The GARD tool interprets the symbolic representation and generates relevant pictorial representations in the format of GARD. GARD serves as a media to exchange AsD and joining information concisely and persistently in a collaborative design environment. The contributions of this paper on collaborative AsD are summarized as follows:

1. The developed AsD formalism specifies the assembly and joining relations symbolically to support collaborative AsD. By using the AsD formalism, assembly

- and joining relations are extracted from the generated AsD model and ARM has mathematically solvable implications. Furthermore, the XML-based AsD model provides a scalable collaborative AsD environment.
- The ARM has three views (i.e. symbolic, mathematical, and pictorial). The pictorial view, GARD, serves as a media to exchange AsD and joining information concisely, selectively, persistently, and in a user-friendly manner in a collaborative design environment.
  - AsD tools, including the AsD engine and GARD tool, are developed to implement the AsD formalism, which leverage an efficient assembly data sharing mechanism and transparent assembly information flow for a collaborative AsD environment through the Internet. The GARD tool interprets the symbolic representation of ARM and generates relevant pictorial representations in the format of GARD. The lean information sharing framework has been developed based on a collaborative product development architecture.
  - The AsD model supplements geometric and topological information of nominal geometry with assembly/joining information, which is essential for various integrated AsD activities, such as joining analysis, virtual prototyping and simulation, and assembly violation detection, in a collaborative design environment.
  - A spatial relationship kernel preserves design intent on the assembly. The spatial relationship implication is inferred to validate the specified joining method that satisfies the designer's intent. This designer's intent is captured explicitly and persistently during collaborative design transactions. In this manner, all design participants can have an equal understanding of the design intent on an assembly.

## Acknowledgements

This research is supported in part by the Office of Naval Research, US Department of Defense (Grant No N00014-02-1-0649).

## References

- FIPER. National Institute of Standard Technology Annual Review on FIPER. General Electric Aircraft Engines, Springdale, OH; December 12–13, 2001.
- Pegasus. NSF IUCRC for e-Design. Strategic Planning Meeting, Pittsburgh, PA, USA; December 9–10, 2003. [www.e-designcenter.info](http://www.e-designcenter.info)
- Sriraman V. Assembly modeling. *Eng Des Graph J* 1999;63(1):9–19.
- Messler RW. Joining of advanced materials. London: Butterworth-Heinemann; 1993.
- Nnaji BO, Gupta D, Kim KY. Welding distortion minimization for an aluminum alloy extruded beam structure using a 2D model. *ASME J Manufact Sci Engng*, accepted, June 2003.
- Moon HS, Na SJ. Optimum design based on mathematical model and neural network to predict weld parameters for fillet joints. *J Manufact Syst* 1997;16(1):13–23.
- Tarnq YS, Wu JL, Yeh SS, Juang SC. Intelligent modeling and optimization of the gas tungsten arc welding process. *J Intell Manufact* 1999;10:73–9.
- Xiong Y, Bedair OK. Analytical and finite element modeling of riveted lap joints in aircraft structure. *AIAA J* 1999;37(1):93–9.
- Menzemer CC, Fei L, Srivatsan TS. Design criteria for bolted connection elements in aluminum alloy 6061. *J Mech Des* 1999;121(3):348–58.
- Kalpakjian S. Manufacturing engineering and technology, 3rd ed. Reading, MA: Addison-Wesley Co.; 1995.
- Cutkosky MR, Tenenbaum JM, Glicksman J. Madefast: collaborative engineering over the Internet. *Commun ACM* 1996;39(9):78–87.
- Kim CY, Kim NK, Kim YH, Kang SH, O'Grady P. Iowa Internet laboratory Technical report TR98-02. Distributed Concurrent Engineering: Internet-Based Interactive 3D Dynamic Browsing and Markup of STEP Data; March 4, 1998.
- Mueller A. Technical White Paper: Shared Engineering with OneSpace from CoCreate; January 25, 1999.
- Parametric Technology Corporation [Online] PTC Windchill Pro/INTERLINK Datasheet. [http://www.ptc.com/products/windchill/engineering/ds\\_prointralink.htm](http://www.ptc.com/products/windchill/engineering/ds_prointralink.htm), (2/15/00).
- Gadh R. Collaborative Product Modeling on the Internet [Online] <http://smarcad.me.wisc.edu/groups/internet/>
- Rojas EM, Songer AD. Web-centric systems: a new paradigm for collaborative engineering. *J Manage Engng* 1999;15(1):39–45.
- Krishnamurthy L, Law KH. A data management model for collaborative design in a CAD environment. *Engng Comput* 1997;13:65–86.
- Florida-James B, Rossiter N, Chao KM. An agent system for collaborative version control in engineering. *Int Manufact Syst* 2000;11(4):258–66.
- Wagner R, Castanotto G, Goldberg K. FixtureNet: interactive computer-aided design via the World Wide Web. *Int J Hum-Comput Stud* 1997;46:773–88.
- Cheng K, Pan PY, Harrison DK. Web-based design and manufacturing support systems: implementation perspectives. *Int J Comput Integr Manufact* 2001;14(1):14–27.
- Boujut JF, Tichkiewitch S, Blanco E. Integration of downstream actors in the design process using a dedicated expert CAD tool for forged parts. *Concurrent Engng Res Appl* 1997;5(4).
- Mervyn F, Kumar SA, Bok SH, Nee AYC. Development of an Internet-enabled interactive fixture design system. *Comput-Aided Des* 2003;35:945–57.
- Nnaji BO, Wang Y, Kim KY, Muogboh OS. PEGASUS: a service-oriented product design and realization engineering system over the Internet. *IIE Trans* 2003.
- Deneux D. Introduction to assembly features: an illustrated synthesis methodology. *J Intell Manufact* 1999;10:29–39.
- van Holland W, Bronsvort WF. Assembly features in modeling and planning. *Robot Comput Integr Manufact* 2000;16:277–94.
- Whitney DE, Mantripragada R, Adams JE, Rhee SJ. Toward a theory for design of kinematically constrained mechanical assemblies. *Int J Robot Res* 1999;18(12):1235–48.
- Whitney DE. A design procedure applicable to different classes of assemblies. Proceedings of the ASME Design Technical Conferences: DETC2001/CIE-21304, Pittsburgh, PA; September 9–12, 2001.
- Rémondini L, Léon JC, Trompette P. High-level operations dedicated to the integration of mechanical analysis within a design process. *Engng Comput* 1998;14:81–92.
- Fu Z, De Pennington A, Saia S. A graph grammar approach to feature representation and transformation. *Int J Comput Integr Manufact* 1993;6(1 and 2):137–51.

- [30] Ambler AP, Popplestone RJ. Inferring the positions of bodies from specified spatial relationships. *Artif Intell* 1975;6(2).
- [31] Liu HC, Nnaji BO. Design with spatial relationships. *J Manufact Syst* 1991;10(6).
- [32] Liu TL, Nnaji BO. A framework of design advisory system for mechanical assemblies. Submitted for publication.
- [33] Liu TL, Nnaji BO. Realization and management of product design constraints in CAD modeling. Submitted for publication.
- [34] Kim DW, Liu TL, Nnaji BO, Kim KY. Sheet metal weld assemblies modeling with spatial relationships. Submitted for publication.
- [35] Kim KY, Kim DW, Nnaji BO. Robot arc welding task sequencing using genetic algorithms. *IIE Trans* 2002;24(10):865–80.
- [36] Nnaji BO, Liu HC, Rembold U. A product modeler for discrete components. *Int J Prod Res* 1993;31(9):2017–44.
- [37] Shah JJ, Rogers MT. Functional requirements and conceptual design of the feature-based modeling system. *Comput-Aided Engng J* 1988; 5:9–18.



**Kyoung-Yun Kim** is currently a research specialist at the United States National Science Foundation (NSF) Industry/University Cooperative Research Center (IUCRC) for e-Design, University of Pittsburgh. He is a member of SME, IEEE, and AWS. He has published technical articles in leading academic journals including *International Journal of Production Research* and *IIE Transactions*. He has a BS and MS in Industrial Engineering from Chonbuk National University, South Korea; and a PhD in Industrial Engineering

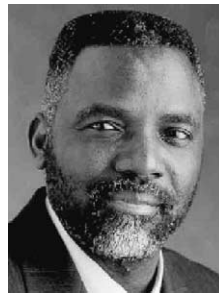
from University of Pittsburgh, USA. He worked as a researcher at the Automation and Robotics Laboratory, University of Pittsburgh from 1998 to 2003. His research interests include design and manufacturing engineering, Internet-based collaborative design, and assembly modeling.



**Yan Wang** is a Research Specialist at NSF IUCRC Center for e-Design, University of Pittsburgh. He received his PhD degree in Industrial Engineering from University of Pittsburgh in 2003. He had MS degree from Chinese Academy of Sciences in 1998 and BS degree from Tsinghua University (China) in 1996, both in Electrical Engineering. His current research interests include collaborative design, geometric data modeling, and constraint-driven systems.



**Obinna Muogboh** joined the Lagos Business School (LBS), Pan African University faculty in February 2003. Before joining LBS, he worked as a researcher at the Automation and Robotics Laboratory, and then the Center for e-Design, University of Pittsburgh. He received his BE in Electronic Engineering from University of Nigeria; and his MSc and PhD in Industrial Engineering from University of Pittsburgh, USA. His teaching and research interests include operations management, simulation, data analysis, modeling, product design, and manufacturing. He also conducts research and consults in the area of lean enterprise and quality management.



**Bartholomew O. Nnaji** has a BS in Physics from St John's University with distinction; an MS and PhD in Industrial and Systems Engineering from Virginia Tech, and obtained a certificate of Postdoctoral studies in Artificial Intelligence and Robotics from MIT. He was Professor of Department of Mechanical and Industrial Engineering at the University of Massachusetts at Amherst till 1996. He was the ALCOA Foundation Professor in Manufacturing Engineering at the University of Pittsburgh till 2003 when he became the William Kepler

Whiteford Professor and the Director of the United States National Science Foundation (NSF) Industry/University Cooperative Research Center (IUCRC) for e-Design, University of Pittsburgh. Professor Nnaji has received three honorary doctorates from international universities. He received 1988 Outstanding Young Manufacturing Engineer Award by the Society of Manufacturing Engineering; 1992 Outstanding Young Industrial Engineer Award; He is a Fellow of Nigerian Academy of Sciences; Fellow of the Institute of Industrial Engineers, and Fellow of the Society of Manufacturing Engineers. 2002, he received the Baker Distinguished Research Award from the Institute of Industrial Engineers. He was honored with the US Secretary of State's Distinguished Public Service Award in 1995; Distinguished Scientist Award by the World Bank-IMF in 1998; and the Nigerian President national honor-Officer of the Order of Niger (OON) in 2000. Professor Nnaji has served as principal or co-principal investigator on over \$35 million research. He has published five books and over 100 technical articles. One of his books, *Computer Integrated Manufacturing and Engineering*, won the 1994 world best text book prize for Manufacturing Engineering. Professor Nnaji is the founding Editor-in-Chief of the *International Journal of Design and Manufacturing* and also served as the Editor for the Design Department of *Institute of Industrial Engineers Transactions on Design and Manufacturing*. Professor Nnaji served as Nigeria's Federal Minister of Science and Technology in 1993.