Emerging Design Methods and Tools in Collaborative Product Development

Edward Red  
*Brigham Young University, ered@byu.edu*

David French  
*Brigham Young University, davidfrench11@gmail.com*

Gregory Jensen  
*Brigham Young University, cjensen@byu.edu*

Sheli Sillito Walker  
*Brigham Young University, sheli.sillito@byu.edu*

Peter Madsen  
*Brigham Young University, petermadsen@byu.edu*

Follow this and additional works at: [http://lib.dr.iastate.edu/edesign_pubs](http://lib.dr.iastate.edu/edesign_pubs)

Part of the [Computer-Aided Engineering and Design Commons](https://lib.dr.iastate.edu/edesign_pubs) and the [Organizational Behavior and Theory Commons](https://lib.dr.iastate.edu/edesign_pubs)

**Recommended Citation**

[http://lib.dr.iastate.edu/edesign_pubs/8](http://lib.dr.iastate.edu/edesign_pubs/8)

This Article is brought to you for free and open access by the Center for e-Design at Iowa State University Digital Repository. It has been accepted for inclusion in Center for e-Design Publications by an authorized administrator of Iowa State University Digital Repository. For more information, please contact digirep@iastate.edu.
Emerging Design Methods and Tools in Collaborative Product Development

Abstract
Product development uses the engineering design process to conceptualize and design new products, while relying on computer-aided application tools like CAD/CAE/CAM that are unfortunately designed for single users. In the absence of multiuser engineering applications, this paper uses surveys and facility visits to show an increased reliance on social communication tools for closing design collaboration feedback loops. Product development requires collaboration among myriad personnel and organizations, each having unique complementary experiences and capabilities. Collaborative design has a primary goal: reduce time-to-market and competitive costs for new products, while retaining quality of product performance and minimizing environmental impact. The focus of this paper is to compare contemporary methods and tools used in collaborative product design at notable corporations to emerging multiuser computer-aided applications. This comparison will define a future where design mistakes and time-to-market are reduced, collaboration is not only truly concurrent, but simultaneously concurrent, and where design rationale is more easily captured and shared for later review and for educational training.

Keywords
simultaneously concurrent, multiuser, decomposition, collaborative tools, communication tools, technical tools

Disciplines
Computer-Aided Engineering and Design | Organizational Behavior and Theory

Comments
This article is from Journal of Computing and Information Science in Engineering 13 (2013): 031001, doi:10.1115/1.4023917. Posted with permission.
Emerging Design Methods and Tools in Collaborative Product Development

Product development uses the engineering design process to conceptualize and design new products, while relying on computer-aided application tools like CAD/CAE/CAM that are unfortunately designed for single users. In the absence of multiuser engineering applications, this paper uses surveys and facility visits to show an increased reliance on social communication tools for closing design collaboration feedback loops. Product development requires collaboration among myriad personnel and organizations, each having unique complementary experiences and capabilities. Collaborative design has a primary goal: reduce time-to-market and competitive costs for new products, while retaining quality of product performance and minimizing environmental impact. The focus of this paper is to compare contemporary methods and tools used in collaborative product design at notable corporations to emerging multiuser computer-aided applications. This comparison will define a future where design mistakes and time-to-market are reduced, collaboration is not only truly concurrent, but simultaneously concurrent, and where design rationale is more easily captured and shared for later review and for educational training. [DOI: 10.1115/1.4023917]

Keywords: simultaneously concurrent, multiuser, decomposition, collaborative tools, communication tools, technical tools

1 Introduction

A new NSF Center Site under the acronym ν-CAx (“Nu = new” computer-aided applications) was organized at Brigham Young University in 2010, as part of NSF’s Center for e-Design coalition. Sponsored primarily by its industry members, ν-CAx research investigates simultaneous concurrency in computer-aided applications (CAX) like CAD, CAE, and CAM (respectively, design, engineering analysis, and manufacturing).

Simultaneous concurrency is a collaborative collective where a diverse group of users simultaneously edit a complex product model. This is quite different from the serial access mode of today which restricts single user edit access to a product lifecycle management (PLM) model file.

We define collaboration as “any team work or focused communication that people have with others in their work place.” But, there are other similar definitions (1) “Collaboration is a process of participation through which people, groups and organizations work together to achieve desired results.” [1] (2) Reed’s law paraphrased: groups of people provided with effective group-forming services dramatically increase the value of their work together [2].

The simultaneous paradigm allows design teams to modify a design model either synchronously and/or asynchronously. Synchronous modification requires team coordination planning, whereas asynchronous modification uses model decomposition for more flexible, even random, team member access, including simultaneous access.

A modern example of simultaneous group access is Google Docs [3]. Editing control can be somewhat chaotic without pre-planned coordination when several users edit the same document characters. Nevertheless, the cloud serving architecture permits an enabled team to simultaneously and openly edit a document from anywhere. This proves effective if users are assigned to edit different sections of the document using formal or ad hoc decomposition methods to avoid collisions.

In contrast to Google Docs, editing a CAX application like CAD poses a more difficult multiuser challenge because of design feature cascading, where model features (shape, form, attributes, tolerances, etc.) are often dependent on prior features and operations. For example, an extrude operation requires a prior defining curve or surface area. When combined under a Boolean operation or blended with another geometric object, the final object is uniquely related and ordered by the previous parameters and operations. Any change in a defining parameter or operation by a non-creating user can result in a nonsensical shape or algorithmic failure.

1.1 Paper Objectives. Given the premise of improved productivity for multiuser CAX, and considering prior research in collaborative CAX [4], this paper addresses two fundamental questions:

- Considering the single user architectures for modern CAX tools, how are product development personnel, particularly engineers, and companies collaborating today in the global marketplace?
- Can modern CAX applications be practically converted to multiuser and what productivity improvements might be expected?

Fundamental questions relating to multiuser collaboration have not been sufficiently addressed by the research community, such as practical issues of organizational adoption of new multiuser technologies, and change and security management of product models. Woolley [5] notes that diversity planning is important when organizing collaborative teams where experts are mixed with less experienced personnel, else, the team can be ineffective. Tools for organizing multiuser teams that can simultaneously evolve design models, considering social, cultural, and technical diversity and competence, do not really exist.

Successful businesses are not easily persuaded to adopt new organizational practices or technologies that may prove somewhat...
disruptive unless these technologies bring great competitive advantage, are fairly simple to implement, protect intellectual property (IP), or address fundamental collaborative deficiencies manifested by product development globalization.

In a Forbes article, Hansen [6] states: “According to the theory of structural inertia, organizations are limited in their capacity to change because they are selected – in evolutionary terms – for their highly reproducible behaviors. Stability is rewarded.” But this has long been a recognized trait of stable companies as noted by Hannan and Freeman [7]: “A prerequisite for reliable and accountable performance is the capacity to reproduce a structure with high fidelity. The price paid... is structural inertia.”

2 Current Collaboration Methods

Considering the single user architectures for modern CAx tools, how are product development personnel, particularly engineers, and companies collaborating today in the global marketplace?

MacCormack [8] notes that traditional centralized approaches to innovation are outdated and that a new model suggests that innovation involves more widespread collaboration among virtual teams and distributed networks of suppliers, firms, and personnel. Virtual teams make it easier for organizations to bring together an extremely diverse group of people with varying skills, experiences, and knowledge. For example, Volvo developed a station wagon through a global collaboration among designers in Sweden, Spain, and the United States. Software called Alias allowed designers on both continents to share and edit images [9].

Thus, the question posed above seems appropriate, considering that product development is now distributed around the globe, among suppliers, providers, subcontractors, divisions, factories, and among diverse cultures.

To answer this question, CAx researchers surveyed modern collaborative engineering practices at major industries in the U.S., all members of the Center for e-Design. The e-Handbook project was conducted for three purposes:

- evaluate the current effectiveness of collaborative practices and tools in engineering industry
- make recommendations to help companies improve their collaborative environment
- create a future vision for collaboration in engineering industry, and assess the potential for multiuser CAx tools

Guaranteeing anonymity to our surveyed companies in aerospace, defense, energy, manufacturing, and medical, we collected data from 9 large and respected U.S. companies (companies A-I) and 144 key personnel at various levels of technical responsibility. We used on-line Qualtrics surveys for all 9 companies. In addition, site visits were made to 6 companies. Visits involved observation of group meetings, facility observations, and personnel interviews, about a week at each site.

The results are summarized in an e-Handbook of Collaborative Engineering, an online website and derived collaborative ontology (Fig. 1) restricted to Center member access [10]. The survey data and results are extensive, thus, we only extract limited data and postulate conclusions appropriate to this paper. This paper postulates that many of the collaborative limitations noted from the survey results relate to the single user architectures inherent in modern CAx tools. Our surveys identified the collaborative challenges of using communication and design applications, including single user CAx engineering applications.

The results expanded the ontology hierarchically as shown in Fig. 2, showing that many factors outside of technical performance influence successful product development within large corporations: organization, social, cultural, communication modes, education and training, knowledge protection, and others.

Our first observation from the data was that product development personnel, particularly engineers, will find ways to collaborate, using available tools, and considering variability in location, time zone, and cultural background. The required knowledge domain for complex product design is simply too overwhelming for one or several technical personnel.

Figure 3 shows that, on average, product personnel collaborate for more than half the work day, either formally in meetings or informally in ad hoc meetings, or through other communication media. Some personnel collaborate most of the day.

The respondents felt their collaboration to be about 76% effective by median, see Fig. 4. Some inconsistencies surface though when we compare Fig. 5 with Fig. 6 and consider formal meetings as a collaborative context. Our conclusion, using respondent data, is that formal meetings are not considered as effective as other collaborative mediums, yet they consume one-fourth to one-third of a person’s collaborative opportunities in a work day. Figure 7 makes this conclusion clearer as responders assign a lower % to meeting effectiveness.

2.1 Communication Tools. If formal meetings are less effective, then other communication means would necessarily be employed, as shown by the tool diversity of Fig. 8. From the survey results, respondents reported that:

- They spend an average of 15% of their time at work on the phone.
- 76% of people use their personal cell phones for work
- 74% of respondents use text messaging for work
- All companies but one had instant messaging available. 79% of the respondents used it for work. Most used it more than once per day, although many reported using it a dozen or more times per day. Many use it to see if someone is at their desk so they can have a face-to-face conversation.
- The median number of emails received per day was 50. But about 35% percent of respondents receive more than 100 e-mails per day. Respondents complain about the number, particularly autogenerated email.
- At some companies e-mail was emphasized as an important way to document decisions and design rationale.
- People who receive more e-mails tend to send more instant messages ($r = 0.62$, strong correlation).
- Collaborators often communicate through a variety of means rather than selecting a single mode of communication.
- Most personnel interviewed used screen or application sharing tools sometimes, if not frequently. This appeared to be a key communication technology in engineering industry, because it allows engineers and technicians to visually communicate and narrate details while at separate locations.
- Face-to-face video conferencing was used infrequently by some respondents, and most said it did not add much value.
- Internal company wikis were used at a few of the companies we visited; some users see internal wikis as an important documentation and communication tool.
For responsiveness reasons, forums and discussion threads were not deemed very valuable by respondents.

Company-internal professional and/or social networks have not yet been implemented in the companies we surveyed.

### 2.2 Technical Tools

The most commonly used CAx tools by distribution are:

- six companies use Teamcenter, ANSYS, MATLAB/SIMULINK, NASTRAN, PowerPoint
- five companies use AutoCAD, Excel, Patran, SharePoint
- four companies use NX (UG), CATIA, LS-DYNA
- three companies use: Hypermesh, ENOVIA, Abaqus, iSight
- two companies use Pro/E, Minitab, Windchill, SAP, CATIA CAM, Solid Edge, MathCAD, C++, SolidWorks, Team Foundation Server
• Other major tools that at least one company mentioned using were ProCAST, Fluent, DELMIA, Gambit, Microsoft Visio, PLM Vis, Adams, Agile, Autodesk Vault, ANSYS CFX.

Often noted weaknesses of these tools are: (1) nonintuitive user interfaces, ease of use, hidden buttons, etc. (2) compatibility with other tools (3) lack of responsiveness (4) training materials (5) single user mode and (6) permission restrictions.

Unsolicited comments that seemed pertinent to multiuser mode and representing needs were:

• “I would like it if there was a way to have multiple users collaborating on one project.”
• “It would be really, really nice if you could use Excel in Google Docs mode.”
• “One person can only work on one part at a time.”
• “Communicating details of design, especially in assembly, is 1000 times faster and easier, when both parties can look at the exact files when coordinating space and intent.”

2.3 Security. Security is extremely important to all of the companies we visited, since intellectual property is a critical competitive advantage. In the case of engineering outsourcing or consulting companies, the intellectual property may belong to another company in which case good security practices must extend beyond company boundaries.

Multiuser security will present unique challenges when teams are organized by expertise, yet IP exposure must be limited because of affiliation or protectionists practices within the IP bearing company. Wang [11] proposes lean information modeling and sharing at a finer granularity than files or models. Methods are tested that extend role based access control (RBAC) methods to filter model views based on access permissions related to user functional roles. Cera [12] proposes a variable mesh algorithm to filter model detail among distributed users based on their design role and access rights within a collaborative team.

Identity management is an organizational initiative to integrate secure identity methods into collaborative commerce. Ahuja [13] makes a strong case for new management structures that manage rights and access to applications and proprietary data critical to distributed enterprises.

Companies often have export control issues (ITAR, EAR). Export control is a name for the controls put in place to ensure that no inappropriate information is given to external partners, suppliers, or other entities. Certain engineers are given the authority to approve or disapprove specific materials for sharing with external entities.

Export control takes a significant amount of time, but is a critical aspect of security. Workflow can get temporarily disrupted when export approval is needed, yet the person with approval authority is not available or is in meetings. One respondent said...
that in some cases the person with approval authority may be unavailable for days. Hence this issue can hinder collaboration with external entities.

Export control will be a significant concern when using multi-user CAx tools. There will need to be methods to decide what engineering data can be shared with other entities in a multiuser environment, not to mention whether or not external entities can access the multiuser environment due to firewall or other technical limitations.

2.4 Collaboration Observations. Although the collaborative tool set varies among the companies surveyed, modern communication tools (Fig. 8) are facilitating the 50% or more daily collaboration time required for modern product development.

Core CAx product development tools like CAD/CAE/CAM are designed for single users, including many supporting tools in PLM that manage design files. CAx tools are not designed for multiuser collaboration, yet these tools remain core to the design process. It seems that their collaborative deficiencies have somewhat promoted the widespread and observed use of collaborative tools not necessarily designed for product development.

3 Emerging Multi-User CAx

Can modern CAx applications be converted to multiuser and what productivity improvements might be expected?

We answer this question with a definitive yes, by providing several multiuser CAx prototypes, associated architectures and new limitation discoveries.

Researchers have been testing the multiuser paradigm for well over a decade [4], yet the major CAx companies have not offered simultaneous multiuser versions of their core software, although tools, such as screen sharing have filled some of these gaps. New collaborative offerings from these companies are mostly serial access solutions, with some relaxation of user file/model accessibility, and limited integration of the modern communication tools described earlier.

A host of prior researchers have demonstrated that multiuser collaboration is both feasible and desirable. Consider, for example, Bonneau’s hierarchical decomposition of building structures [14], Cera’s CAD feature access rights [15], Chen’s Internet adaptable collaboration [16,17], Fan’s peer-to-peer architectures for distributed collaboration [18], Fuh’s collaborative state-of-the-art review [19], Jing’s local locking model mechanisms for concurrent collaboration control [20], and Red’s multiuser collaborative CAE architectures and model space decomposition [21,22].

Several important architectural limitations have not been addressed sufficiently, such as multithreading of CAx API’s and GUI’s, access to CAx event handlers and interrupts, client session undo’s/redo’s (change management), and API’s that can provide feature parameter copies rather than address handles, along with a hosts of organizational and security features that must be considered by the research community. Our research prototypes were designed to discover these additional limitations, e.g., expanding into the multiuser CAE area (such as finite element analysis/preprocessing).

3.1 Multiuser Adoption Factors. When new technological methods are not adopted, such as multiuser CAx, it usually means that industry has not unanimously requested this new capability, possibly not considered the competitive advantages in light of required organizational and proprietary process changes, and/or the CAx companies view the architectural enhancements ill-defined, difficult, and/or risky. The authors suggest that all these factors have contributed to delaying collaborative multiuser CAx tools, plus additional factors that follow.

Another possible factor for delay is that our modern computer and associated applications have evolved over several decades to empower the individual, not the team. The desktop/PC, laptop, tablet, and mobile devices provide an environment where numerous applications give individuals desired functional autonomy and perceived job security. Horner [23] discusses several barriers to gathering design rationale from designers, noting one trait of intentional omission by designers: “Designers may be hesitant to simply give away knowledge without knowing who will use it or how it will be used.”

In product industries, information security is paramount. Single user practices reinforce a protectionist’s attitude. For example, it is much easier for PLM systems to manage IP for product designs.
if only one individual is authorized to edit a model, and rights access can be delegated to a responsible manager.

Although modern CAX tools are single user, their capabilities have continued to both improve and proliferate with each release version, including advances in parametric design and integrated design cycling to evolve model parameters through CAD and CAE design loops. Perhaps we have effectively isolated design knowledge within individuals rather than shared it.

Recently, globalization, virtual teaming, and product competition have begun to stress the single user applications tools, somewhat explaining the diversity of communication tools in Fig. 8. Reference [24] explains why: “The extended enterprises involved with product development processes are heterogeneous environments beset with disparate CAD models....producing a constantly shifting web of product development and delivery partnerships. The lack of effective interoperability among the extended enterprise threatens product quality, drives up costs, and lengthens time-to-market.”

Our premise in organizing the v-CAX site was that the large product development companies were not yet aware of the tremendous competitive advantages of multiuser CAX, one advantage being inherent interoperability in cloud serving architectures, and that any such change would require the support of the major CAX developers and vendors. In addition, research was needed to demonstrate multiuser CAX practicality by addressing architectural gaps and other limitations.

We have developed and are developing many multiuser prototypes (under the names_Connect) using the application programming interface (API) libraries of major CAX applications like Siemens NX, Dassault Systemes CATIA, CUBIT (FEA package developed by Sandia Corporation), Autodesk Inventor, and others. We will briefly present our multiuser prototypes, decomposition considerations, and related architectures for NX (API access), CATIA (API access), and CUBIT (source code access).

4 NX Connect

NX Connect allows multiple users to access and make changes simultaneously to a single part file as shown in Fig. 9.

The client-server (CS) integration software designed in C# allows multiple users (clients) to simultaneously access and edit a part file. Each user can independently edit and view the part, zoom and rotate views. When a user edits the part, the server broadcasts changes to each client workstation. Each client maintains a local copy of the part file which is constantly updated.

The CS architecture uses a thinner server and a stronger client. The server maintains the master data for the part file in the information storage module (ISM) as stored in the Fig. 10 format. Although the software functionality changes little, we are currently implementing a cloud serving architecture where the client uses an application like HP’s RGS to turn a workstation into a terminal window/portal. The CAX application, design model(s), files, collaborative database, and interfaces are distributed on cloud CPU’s (blades) to provide a more secure and interoperable framework. This has implications for network configuration and latencies, cloud server location relative to company firewalls, and licensing requirements.

NX Connect uses three custom modules shown in Fig. 9(b) and described in the following sections. We conclude these three sections with an example and other considerations.

4.1 Information Storage Module. This module uses Microsoft’s sql Server and a hierarchical structure to store and sync the part features and related data, including all information relating to users, parts, and features. It broadcasts changes from each user to all other users as it receives them.

4.2 Data Capture Model. This module uses the NX API to parse the feature tree and find any new (unnamed) features. It then names those features and creates an NXOpen “builder” object for each feature which auto populates with the feature parameters and related information. The feature parameters and information are then extracted from the builder and stored in database objects (LINQ to sql data classes), which are pushed to the ISM for distribution to the other clients. Each database object represents a change to the part.

4.3 NX Controller. This module receives client model changes from the ISM in the form of database objects, and extracts the feature information to construct the corresponding feature in the NX session on the user’s machine using the NX API. Creating a new feature using the API requires creating a new
NXOpen builder object of the appropriate type and then passing that builder the feature parameters.

4.4 Prototype Example. Figure 11 shows three clients simultaneously building a jet engine front frame in various stages of coordination (Figs. 11(a)–11(c)). The final model design is shown in Fig. 11(d).

The design session time required for the three users was about one-third the time for a single user, although some preliminary coordination and decomposition planning was required.

4.5 Decomposition Considerations. The example of Fig. 11 required that the multiuser group negotiate an ad hoc spatial decomposition of the model before the design session.

Multiuser assignment and model decomposition will be a non-trivial process. New organizational methods will extend beyond simple employee familiarity and work histories. This research topic has been neglected thus far and is a strong contributor to the nonadoption of multiuser methods.

5 Constraint Decomposition

Marshall [25] investigated the use of constraint surfaces to decompose clients into defined spatial regions of a CAx model. With the aid of Siemens NX developers, a method was implemented that used constraint planes for spatial decomposition.

Figure 12 shows a model divided into four design spaces assigned to four client designers. If a user tries to move to a nonassigned region and select a feature in that region, the feature selection is blocked, according to the selection filtering tool described next.

5.1 Selection Filtering Tool. The filtering tool is a graphical user interface (GUI) and software that runs as a .dll inside of NX. The GUI remains open while the .dll is running and filters the allowable selection based on the user selected feature.

Due to NX architectural limitations the .dll has to be triggered manually and filtering only lasts as long as the GUI remains open. In this simple prototype the constraint boundaries are planar constraint equations associated with each user.

The selection filtering portion of the implementation is integrated within the CAD system (mouse cursor event combined with feature selection ray cast normal to the viewing window) and has a single dialog window that allows for selection among different multiusers.

Depending on the user, a selection filter is applied to all possible selections based on four constraint planes. This early prototype allows for the selection of edges and faces. The selectable edges and faces make up a model which can be described by $P$, where any point $p$ is described by coordinates $x$, $y$, and $z$. Let $X, Y,$ and $Z$ represent the $x, y,$ and $z$ ranges of points $p$ in $P$ such that

$$x \in X; \ y \in Y; \ z \in Z \quad (1)$$

For the model of Fig. 11 the following simple constraint equations in (1)–(5) have been implemented using inch units. ACCEPT $P$ means a feature on the model is selectable by the multiuser. A selectable feature is one that can be edited by the multiuser in the CAx application.

User 1: only select edges and faces for which

$$\text{if any } x \in X > 2.15, \ \text{ACCEPT } P \quad (2)$$

User 2: only select edges and faces for which

$$\text{if any } z \in Z > 1.013, \ \text{ACCEPT } P \quad (3)$$

User 3: can select edges and faces for which

$$\text{if any } y \in Y < 0, \ \text{ACCEPT } P \quad (4)$$

User 4: can select edges and faces for which

$$\text{if any } y \in Y > 0, \ \text{ACCEPT } P \quad (5)$$
if any $x \in X < 2.15$ and $y \in Y > 0$
and $z \in Z < 1.013$, ACCEPT $P$

Normally a feature would highlight as the mouse hovers over it to show what would be selected if the user were to click the mouse. However, if a feature is not selectable based on the current filter applied, the features will not highlight at all when the mouse hovers over it.

There is also an option in the menu to toggle on/off the visible constraint boundaries. The constraint planes placed at the edge of the user’s selection boundary are colored differently for each user.

5.2 Decomposition Conclusion. This simple prototype shows that a primary CAx application like NX can be configured through the API to provide regional blocking for multiuser simultaneous editing of design space.

Adding more complex constraint surfaces like those in Table 1 would not be difficult to implement. Note that the scalar comparisons are coordinate system independent.

6 CATIA Connect

The CATIA Connect prototype uses a similar CS architecture and when ported to C# the prototype accesses CATIA’s API through the COM object interface. The model data is stored on a server running Microsoft SQL Server 2008.

The C# version of CATIA Connect is able to run on a separate thread from CATIA. This allows a timed sync to run automatically in the background every few seconds. The program first checks for new features created on the local client. It then retrieves the primitive data associated with those features and sketches, and sends the new data to the server.

All features and sketches are assigned a time-stamp. The local client then downloads features created by other users since the last sync.

The multiuser test session recreated the same jet engine front frame used in the NX Connect demo by creating pads, shafts, pockets, grooves, and circular patterns. The instructional steps are a sample of the coordinated actions. We note that user sessions were staggered to demonstrate that users could enter or depart a multiuser asynchronous session at any time. Again, the production time is reduced in proportion to the number of client designers.

User 1:
Step 1 - To create the front frame’s inner most section, make the sketch of Fig. 13 in CATIA V5 on the XY plane and use it to create a $360^\circ$ shaft around the horizontal axis.

Step 2 - Update CATIA Connect after shaft is complete. Figure 14 shows the shaft along with the middle rim and a fin created by users 2 and 3.

User 2:
Step 1 - Create the middle rim by making the sketch of Fig. 15 on the XY plane and revolving it as a $360^\circ$ shaft. The thickness is 0.1 in. (0.25 mm) with the front constrained along the V axis.

Step 2 - Create the sketch for the fin bosses on the surface of the middle rim, Fig. 16. Create a circular pattern with ten instances of the feature using the complete crown method and any of the rims as the reference axis, Fig. 17.

User 3:
Step 1 - Build the fin by making the sketch shown in Fig. 18 on the new plane and pad it 1 in. (25.4 mm) to –2 in.

Fig. 12 Geometric constraint limiting of user feature selection

031001-8 / Vol. 13, SEPTEMBER 2013 Transactions of the ASME
### Table 1 Constraint surface equations

<table>
<thead>
<tr>
<th>Constraint type</th>
<th>Graphical</th>
<th>Constraint surface</th>
<th>Constraint forms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plane (unbounded)</td>
<td><img src="image" alt="Plane Diagram" /></td>
<td>( \mathbf{n}^T \mathbf{p} = d )</td>
<td>( \mathbf{u} = \text{user selected point} )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( \mathbf{n} = \text{outward plane normal} )</td>
<td>( \mathbf{n}^T \mathbf{u} &lt; d ) (inward, IE)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( \mathbf{p} = \text{point in plane} )</td>
<td>( \mathbf{n}^T \mathbf{u} \leq d ) (inward, EQ)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( d = \text{plane distance} )</td>
<td>( \mathbf{n}^T \mathbf{u} &gt; d ) (outward, IE)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( \mathbf{n}^T \mathbf{u} \geq d ) (outward, EQ)</td>
<td></td>
</tr>
<tr>
<td>Cylinder (unbounded)</td>
<td><img src="image" alt="Cylinder Diagram" /></td>
<td>( (\mathbf{p} - \mathbf{v})^T \mathbf{e} = r )</td>
<td>( \mathbf{u} = \text{user selected point} )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( \mathbf{n} = \text{cyl axis unit vector} )</td>
<td>( (\mathbf{u} - \mathbf{v})^T \mathbf{e} &lt; r ) (inward, IE)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( \mathbf{e} = \text{unit vector normal to cyl axis unit} )</td>
<td>( (\mathbf{u} - \mathbf{v})^T \mathbf{e} \leq r ) (inward, EQ)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( \mathbf{v} = \text{point on cyl axis} )</td>
<td>( (\mathbf{u} - \mathbf{v})^T \mathbf{e} &gt; r ) (outward, IE)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( \mathbf{p} = \text{point on cyl surface} )</td>
<td>( (\mathbf{u} - \mathbf{v})^T \mathbf{e} \geq r ) (outward, EQ)</td>
</tr>
<tr>
<td>Conical frustum (bounded, reduces to cone if ( r_2 = 0 ))</td>
<td><img src="image" alt="Conical Frustum Diagram" /></td>
<td>( r_c = (\mathbf{p} - \mathbf{v})^T \mathbf{e} )</td>
<td>( \mathbf{u} = \text{user selected point} )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( r_2 \leq r_c \leq r_1 )</td>
<td>( \text{Step 1: } d = (\mathbf{u} - \mathbf{v})^T \mathbf{n} )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( \mathbf{e} = \text{unit vector normal to cyl axis unit} )</td>
<td>( \text{Step 2: Inward} )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( \mathbf{v} = \text{point on cone axis at base where} )</td>
<td>IE: if ( 0 &lt; d &lt; h ) and ( r =</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( \mathbf{p} = \text{point on conical surface} )</td>
<td>( r_c = r_1 + d(r_2 - r_1)/h )</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( r_c = \text{cone radius at } \mathbf{p} )</td>
<td>( r =</td>
</tr>
<tr>
<td></td>
<td></td>
<td>( h = \text{frustum length} )</td>
<td>\text{Step 2: Outward}</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>IE: if ( 0 \leq d \leq h ) and ( r =</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>( r_c = r_1 + d(r_2 - r_1)/h )</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>\text{Step 2: } \mathbf{e} &gt; r_c \text{ or } \text{Step 2: } \mathbf{e} \leq r_c \text{ given } r_c = r_1 + d(r_2 - r_1)/h</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>\text{EQ: if } \text{Step 2: } \mathbf{e} &gt; r_c \text{ or } \text{Step 2: } \mathbf{e} \leq r_c \text{ given } r_c = r_1 + d(r_2 - r_1)/h</td>
</tr>
</tbody>
</table>

---

![Fig. 13 Sketch plane cross section of shaft](image)
(50.8 mm). The edge of fin is constrained to edge of rim and arc is tangent to both lines. Now update the part.

**Step 2** - Pattern the fin in a complete crown circular pattern with ten instances using any rim as a reference, see Fig. 19.

**Conclusion** – Like NX Connect the CATIA API and prototype show that other CAx applications expose their processes sufficiently to construct desired multiuser environments.

7 CUBIT Connect

CUBIT, a mesh generation tool developed primarily by Sandia National Laboratories, is comprised of the CUBIT Core that executes the meshing algorithms, and Claro, the GUI, Fig. 20.

The Core and Claro GUI communicate using a C++ API called CubitInterface as shown in Fig. 20. Since we had access to the CUBIT source code, we were able to program in C++ to develop our CUBIT multiuser prototype within CubitInterface.

Whenever the user creates geometry, mesh, etc., a command string is automatically generated and passed to the CUBIT Core through the CubitInterface. The command is then executed and the results passed back to the GUI for display through the same interface.

Figure 21 shows a race car developed in collaboration with 26 universities over four years as part of the PACE Global Vehicle Project. This model was decomposed into three regions so that CUBIT multiuser editing functionality could be tested. Three separate users simultaneously edited and meshed their assigned components in three separate regions.

During these sessions one interesting problem surfaced that complicates FEA multiuser collaboration. Some meshing algorithms can take minutes or hours to perform. Any transmitted client command could possibly halt all multiusers from editing their model for minutes or hours while the meshing algorithms execute.

We are still investigating how to filter and/or stack update commands from other multiuser clients. Algorithmic delays are not a substantial problem in CAD applications, a minor problem in CAM applications, but can be significant in CAE applications like CUBIT. Current research is considering operational vectors that are tagged to each multiuser and that can be applied at optimal times.

The CS version of CUBIT Connect utilizes Windows named pipes (NP) for interprocess communication (IPC) and TCP/IP sockets for network communications, see Fig. 22. Two External and Internal networking clients reside on each local computer. Because of superior network programming abilities, C# was
Fig. 18 Sketching/revolving the middle rim

Fig. 19 Pattern the fin
chosen to program the external client (EC) that runs on the local computer as well as to program the server.

7.1 Client. As mentioned previously, the client consists of two separate programs: internal client (IC), which is built into the source code of CUBIT and the external client (EC), which runs outside of CUBIT. To facilitate the communication between the IC and the EC, two NP’s are created as illustrated in Fig. 23.

7.2 Internal Client. The IC is written in C++ and has two main functions:

• intercept the command strings from CUBIT and send them via NP to the EC
• gather incoming multiuser commands from the EC and execute them in CUBIT

The IC is built directly into the CUBIT source code. It consists of two running threads, one dedicated to sending commands and the other dedicated to receiving. Whenever a client computer connects to both reading and writing pipes generated by the EC, it will create a client listening thread (CLT) dedicated to checking the pipe for incoming messages from the EC. While the main thread is constantly sending unfiltered messages, the CLT is constantly reading for incoming messages in the writing pipe as shown in Fig. 23.

If there is a message, the reading thread will immediately place the message inside of a client queue (CQ) for CUBIT Connect to extract and process. This constant process is placed in a while loop, and the IC will constantly go through the CQ and update accordingly.

7.3 External Client. The EC is where the majority of client identification and message categorization takes place. Every time the EC is executed it generates reading and writing pipes that wait for the internal client to finish the network hand shaking process. Once the pipes are established, it will initiate the connection to the central server through TCP/IP sockets. All of these processes have to be done sequentially in order to prevent any race conditions.

The EC is not only an important transition point between CUBIT and the server, but it is also where different message types are organized through a serialization process. Here, different message types, such as the client’s unique ID and the original message, are combined into message structures. Some common message structures established in the EC are command message, master trigger, and database reset. Command messages are generated from the CUBIT GUI for the CUBIT core to process (e.g., “create sphere” or “mesh volume 1”).

8 Multi-User Productivity

In 2008, Ian Ziskin, chief human resources and administrative officer for Northrop Grumman, noted that in the next decade 50% of aerospace engineers would reach retirement age [26]. This number also approximates retirement futures for other product industries. The concern is that, in complex product industries, productivity could significantly decline with a rapid shift from experienced to inexperienced engineers.

This retirement scenario requires new learning technologies to accelerate experiential training of novice engineers. If CAx
applications could be made universally multiuser, then experienced engineers could train novice engineers in group sessions and without localization constraints. Novice engineers could be trained by experienced engineers as they simultaneously edit/analyze models, considering decision rationale, and working within diverse teams, distributed over several time zones.

8.1 Session Productivity. To compare session productivity we tested our multiuser prototypes against commercial single user applications using the jet engine front frame structure of Fig. 11(d). Our research indicates that, for reasonably complex models, design times can be reduced in proportion to the number of multiusers; see the Table 2 time comparisons for the front frame design presented earlier. Table 2 includes the latest CATIA V6 Collaborate commercial offering (let modeler = multiuser).

8.2 Organizational Decomposition. Productivity estimates are realistically more complex than a simple proportional prediction. Simple questions like these make this apparent:

- What makes a model suitably complex for a multiuser assignment?
- Considering a CAD or CAE application, can feature or spatial independencies (e.g., fillet, pocket, flange, boss, spatial variances in shape/curvature) within the design be used to guide the decomposition strategy?
- Can a team with experiential, locational (global, local), and cultural variances be effectively joined into a simultaneously concurrent team?
- How would CAx GUI’s be modified to provide communication and design consistency among varying cultures on a multiuser team? Xu [27] showed that multiuser GUI’s could integrate automatic language translation of text messages between a culturally variant team of CAx collaborators.
- Do pervasive CAx experiential databases exist within companies, suppliers, and subcontractors to guide the organization of a CAx multiuser team by experience, social compatibility, and by schedule?
- Are modern tools available to guide the creation and management of a multiuser team with an optimal mix of expertise?

8.3 Productivity Future. Although any organizational productivity measurement scheme appears immature at the present, the v-CAx Site has tested distributed collaborative effectiveness in academic situations.

In one case 10 BYU, Georgia Tech and University of Puerto Rico Mayaguez engineering students engaged in a simultaneously concurrent project. NX Connect and CUBIT Connect were used by the collaborating students to redesign an F-86 Saber Jet wing structure by simultaneously editing and analyzing the design [28]. A Boeing prepared video proclaimed the success of this distributed multiuser experiment.

Another test case used BYU’s introductory ME EN 172 CAD course to engage 35 freshmen in simultaneously designing a carbon fiber monolithic composite wing structure for a glider. Each student was assigned a rib in the composite design [28]. They were able to successfully create, scale, and transform NACA points for each rib, connect each set of points into a sketch of a rib, and then extrude their sketch.

Entering engineering students are often experienced in modern gaming techniques where distributed team members collaborate in a gaming landscape of dynamically changing models and goals. A normal feature of these applications for global teaming is team matching, where users are automatically grouped on expert level and other cultural/social data.

Given that similar team matching tools are developed for multiuser CAx, and considering that these tools will be electronic and networked, there will be methods to capture user experiential data, including raw design session rationale that can be associated with a design/analysis model.

Electronic multiuser formats will ease the gathering of raw session collaborative decision data, see Mix [29]. Automatic design rationale methods like those of Myers et al. [30] can then be applied to these data, or management can simply review the raw decision history and derive their own conclusions.

Design rationale gathering is vital to a company’s IP, yet single user CAx tools simply do not provide for strategic rationale capture. In the broad design rationale survey by Regli et al. [31], he said, “The need for design rationale is a common problem, but successful design rationale systems are rare. The need to record and preserve intellectual capital drives organisations to manage knowledge.”

9 Current CAx Architectural Limitations

The research and prototype implementations have enabled us to draw conclusions about the single user limitations in commercial CAx applications. These limitations presently force multiuser capabilities to be implemented as an extension to the main CAx application and programmed bit-by-bit for each type of user action.

Some actions may be difficult or impossible to program due to current API access limitations. In CUBIT Connect we had access to the source code and therefore the event handler. This allowed us to make virtually all of CUBIT’s commands multiuser with less effort. Where we had no direct access to the event handler, we used other methods such as parsing undo marks to access model parameter changes in a multiuser session.

API’s provide handles to geometric objects (memory addresses) that cannot be passed to other multiusers over a network since memory locations on each computer vary. Our prototypes required extensive programming to extract the parameters from the data structures. Multiuser CAx will need API’s that provide the parameters directly, like copying the object, rather than just memory addresses to the object.

CAx applications have a concept of a single display screen, a single viewpoint, and a single cursor. They often run on a single serial interface thread. Modern CAx GUI’s are built for single users and are not intended to be shared by multiusers using multiple threads. Users cannot simultaneously view the model from different viewpoints or simultaneously edit the model parameters in the same GUI, which requires more threading sophistication.

Event interrupts are not available in API calls that signal parameter changes inside the CAx application. Interrupts would keep the computer from busy-wait polling, and switch the context to an interrupt handler. Interrupts can add new functions such as a sound alert when the server receives a new update and sends it to each client.

Undo sequences are vulnerable to undo actions by multiusers. This is further complicated by algorithms that take substantial time to complete. These major architectural deficiencies must be addressed by CAx developers before multiuser collaboration can be practical.

9.1 Undo Command (Ctrl Z). Undo (and redo) operations, persistent only during an editing session and not stored permanently, are far more complicated in a multiuser environment.

Error recovery has not really been addressed by the research community although Li [32] notes that “a feature of a product...
model might depend on other features and modifying an early feature may cause later features to become invalid." This is further complicated by any user in a multiuser session that uses the feature undo function.

An undo can cascade such dependencies and cause chaotic collaboration. This is why some researchers are proponents of locked/blocked feature editing or decomposed regional assignment, including the authors. Research is currently underway to implement and test new undo/redo handling methods among a multiuser team.

10 Conclusions

Product development requires collaboration. The engineering design process is taught and adopted as a collaborative process for evolving designs. Yet the primary CAx tools are single user.

Technical personnel at the surveyed companies are using modern communication technologies to collaborate about 50% of each work day, whether designed for technical product development or not. We postulate that this dependence is related in part to the lack of collaborative capability in the engineering CAx tools, like CAD, CAE, and CAM.

Despite incomplete multiuser research breadth (multiuser undo/redo, distributed security, cloud serving and firewalls, API memory access limitations, nonaccessible event handlers, etc.), multiuser CAx is both feasible and architecturally necessary. The rapid absorption of communication tools into product development activities indicates serious collaborative deficiencies in existing CAx tools. Given new multiuser CAx tools that overcome the architectural limitations identified in this paper, productivity will be improved, particularly for complex products and large assemblies.

In the absence of multiuser organizational tools we cannot generalize productivity improvement numbers. Nevertheless, our research has shown that session productivity will scale with the number of multiusers that can simultaneously edit a model, assuming new tools for multiuser team matching and management, along with more sophisticated tools for model decomposition.

Contemporary decomposition methods use product modularization, and related interface specifications and requirements data, along with new "cyberinfrastructures" to evolve a product design among a globally distributed development team, see Yoo [33]. Multiuser CAx methods offer new advantages by simultaneously integrating a virtual team, thereby reducing the need for modularization by localization and specification assignment.

Perhaps the greatest potential of multiuser CAx is transference of design knowledge and rationale among virtual teams, or among new engineers that need to gain experience quickly, particularly when companies conduct global product development.

Acknowledgment

The National Science Foundation, $\nu$-CAx member companies, and $\nu$-CAx research students are acknowledged for their funding and conducting of this research and Center Site. Most importantly, our member companies, by providing access to facilities and personnel, have helped us assess their current collaborative environments and the potential for multiuser CAx improvements.

References