Considerations for Multi-User Decomposition of Design Spaces

Edward Red[1], Felicia Marshall[2], Prasad Weerakoon[3] and C. Greg Jensen[4]

[1]Brigham Young University, ered@byu.edu
[2]Brigham Young University, feliciamarshall@byu.edu
[3]Brigham Young University, prasadwee@yahoo.com
[4]Brigham Young University, cjensen@byu.edu

ABSTRACT

The advent of multi-user computer-aided applications (CAx) like multi-user-CAD/CAE/CAM will change personnel/organizational assignment processes in product development. In the near future collaborating personnel/organizations will enter design sessions and simultaneously edit/review design spaces. This paradigm shift will require new methods to be developed that decompose development tasks over personnel/organizations at both local and global locations. Experiential data will not be restricted to suppliers, organizations, or sites, or other grouping types, but reflect a different granularity where a particular group of individuals from a variety of organizations might be collected into design teams for optimal collaboration. This paper will consider general collaborative principles from two perspectives: 1) from an administrative perspective and 2) from simple CAx prototypes that demonstrate how to decompose complex design models among several multi-users. Thus, the paper considers the decomposition challenges that must be resolved from project administration to project conduct, i.e., from the top down. The paper demonstrates that modern CAx applications already have some of the tools needed for multi-user decomposition, if not the mentality.

Keywords: collaborative design, multi-user decomposition.
DOI: 10.3722/cadaps.2013.803-815

1 DECOMPOSITION PRINCIPLES

What principles can we use for decomposing a design space among several multi-users. We define a design space as the specifications, model, and CAx application(s) used by multi-user personnel to develop a product/component. Current practices of design space decomposition are limited by single user computer-aided applications (CAx), and the serially arranged steps used in the product development process, often referred to as the Engineering Design Process. Conceptualization, specification, alternative solutions, design, analysis, and production remain mostly serial activities because a single user is allowed to edit a CAx design model at a time. Even complex assembly models can only be entered by one user at a time. Collaboration still takes place using different forums, such as formal and ad-hoc meetings, calls, emails, texting, web conferencing, and screen sharing. In fact, engineers and related product development personnel, under development and schedule pressures, will find and apply tools that promote collaboration and efficiency.

When a new product is to be developed the *administrative system* must decompose the processes and models at different stages among personnel, departments, suppliers, sites, resources, etc. How might existing processes and organizations take advantage of new CAx tools that will permit multi-users in a design space/model simultaneously? Would multi-user decomposition change organizational and management structures or the process steps used to manage product development? Would it affect training methods, or training materials, or the set of applications that a company installs? It seems that the answer is obviously yes. But how would it affect the management of product related databases, and networks, or personnel hiring, or supplier selection? These are interesting questions to consider, and might only be answerable after some general principles are derived from further multi-user research.

What multi-user decomposition principles could we derive from current decomposition processes? Martinez [6] discusses decomposition principles applied to virtual enterprises (VE), and dedicates a section to task decomposition. The decomposition principle uses task decomposition by function, where each task is evaluated according to cost, technology and production resources. But note this sentence: "Successive process decomposition is sometimes needed to determine a set of tasks in which every task can be assigned entirely to a single organisation (this organisation would usually be a partner firm but it could be also another VE seen as a single firm)." The statement *"...every task can be assigned entirely to a single organisation ..."* seems to imply only one architectural viewpoint for decomposition is perceived today: ***a single organization***. Methods used today do not perceive simultaneous task collaboration among different entities such as companies, facilities, suppliers, departments, groups or organizations. This mindset is typical of modern management approaches to process decomposition at every level within product companies, and is due in part to the serial, single user architecture of the computer-aided applications (CAx) used by different organizations.

Consider further the contemporary top-down decomposition approaches from initial product decomposition into a number of sub-problems/projects distributed to divisions, departments, local or global sites/facilities, strategic development groups, or sub-contract suppliers. We postulate without proof how these contemporary methods might function for multi-user decomposition.

Effective decomposition is strongly dependent on good timely information and past experiences. Methods are needed for constructing relevant and pervasive product and resource information databases, including personnel capabilities, and that are easily searchable. Pervasive company-wide personnel and organizational experiential databases are not presently maintained and readily available. Significant research is needed to develop both the principles and tools used for the three levels below. We do note that there is limited research into this vital area; consider Ip's [4] CAD classification methods and the European ETIM classification standards [13], [14]. There are also some commercial efforts; see URL's [11] and [12].

- **Conventional Top-Level: specification decomposition** – Current approaches decompose design space based on review of design concepts and specifications for some new product or component. Experienced personnel will decompose product tasks based on similar product development histories. New products are often incremental improvements to existing products, motivated by consumer and market expectations, or by new emerging technologies, or by societal needs and constraints; thus, there is usually historical data and experience that make task decomposition simpler. These decisions are made by experienced personnel at a high level of product development management, yet Herrmann [2] notes that the process is still quite mysterious, and more effective if product decomposition can be stated as a set of sub-problems that are simpler to engage. This is the de-facto approach today.

  Of the notable theoretical approaches to decomposition, Browning [1] uses Design Structure Matrices (DSM) to establish relationships at several levels from top to bottom: 1) architectural/component; 2) organizational; 3) activity/schedule; and 4) parameter or feature-based (low level). Because DSM's are two dimensional they can expose dominant relationships and dependencies without the added complexity of increased dimensionality to disclose more subtle decomposition relationships. A number of commercial applications support the DSM identification process [10].

Multi-User Top-Level Postulate: specification decomposition – Multi-user decomposition would require additional specification examination to determine possible benefits to allowing multi-user access to the models, where decomposition decisions and related databases relate to model complexity, personnel types, resource limitations, supplier and outside contractor engagement, product team distribution (global, time zone management, security restrictions, etc.). At this level, multi-user would be more correctly interpreted as multi-entity, i.e., local or global organizations, companies, or suppliers that collaborate on a common task. Obviously, management could not make these decisions without access to resource databases that maintain experiential relevant data.

- **Conventional Mid-Level: experiential decomposition** – Current approaches decompose design space among personnel based on user or organizational experience and background (section, group, subsidiary, sub-contractor, supplier, etc.), when correlated with proposed model features and specifications. Consideration of resources and development team organization (local, national, global, suppliers, contractors, communication system and effectiveness) will guide the decomposition decisions also.

  Multi-User Mid-Level Postulate: experiential decomposition - Multi-user decomposition would require comparison of available user backgrounds against model specifications and features to determine whether personnel are available for multi-user mode. It would seem that new databases would maintain relevant experiential data about personnel based on their educational background, technical and model space experiences, and expertise both in technical matters and in human-interaction and management. It is likely that department or group level resources, i.e., experiential databases, can be searched to establish expertise and experience, and existing schedules can also be used to determine personnel availability.

- **Conventional Low-Level: feature decomposition** – Current approaches examine a particular design space/model to determine the important features and related parameters of either a conceptual model (how you think the model will look) or a developed model (developed already, and used for next design space, e.g., a geometric model used to develop a manufacturing process plan). In this approach a manager will typically assign the model to a single user based on experiences and availability. Note that extensive research was conducted in the late 80's and 90's relating to feature extraction from models, e.g., Joshi [4]. The basic concept is that you can assign attributes to features that would be useful in assigning manufacturing resources to producing the model. This research was motivated by a wave of research directed at new methods for concurrent engineering.

  Multi-User Low-Level Postulate: feature decomposition - Multi-user decomposition would examine the model design space to determine model complexity and whether there several regions that are independent or mildly independent, based on proposed model features, or similar products previously developed. If personnel resources are available, and model is sufficiently complex, multi-user decomposition of the design model among several users may be desirable. In the case where training of novice users by more experienced personnel is desired, model complexity may be relaxed

2 MULTI-USER LOW-LEVEL DECOMPOSITION

The principles suggested in the previous section will require significant R&D to formalize methods of decomposition at the higher administrative levels. This section considers lower level decomposition prototypes to establish practical feasibility using well known CAx applications. Multi-user decomposition among computer-aided applications (CAx) will be more effective for design spaces/models that can be divided among several users by region and constraint. Under design space control constraints it is possible that several users can edit models and finalize the model simultaneously. Multi-user applications can range from CAD designs and assemblies, to FEA pre-processing or analysis models, electronics IC modeling, to CAM process planning, architectural building designs, and many others.

Figures 1 and 2 illustrate two examples of more complex models: a casting and a FEA model, respectively. How might decomposition be applied and regulated so that several multi-users might

Fig. 1: Complex casting.

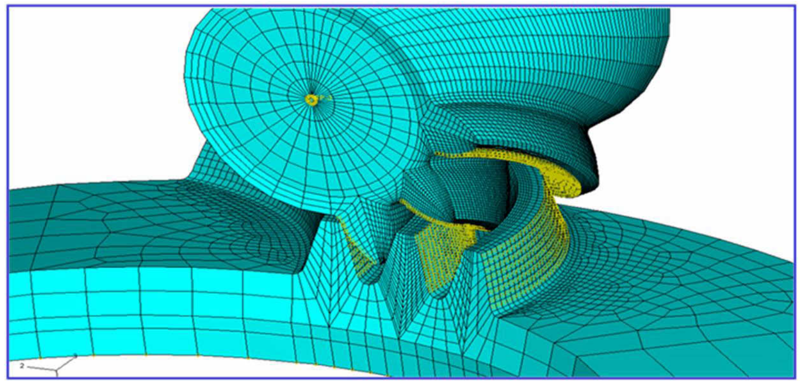


Fig. 2: Complex FEA model.

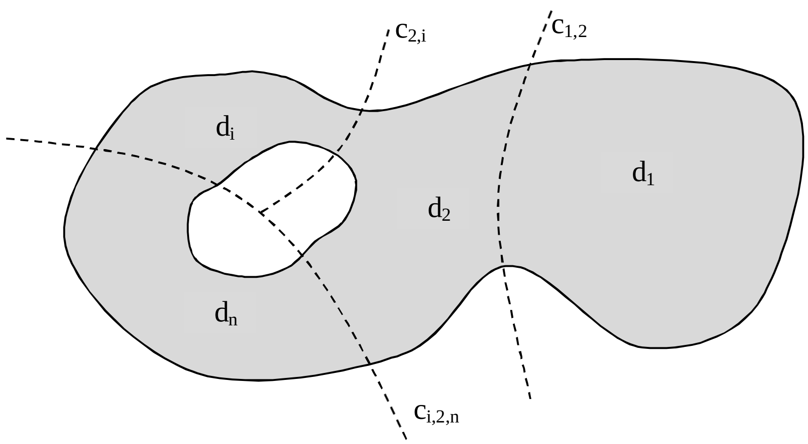


Fig. 3: Regional decomposition.

simultaneously review, develop or edit these models? We consider two processes for multi-user region management: *region blocking* and *region coordination*. See Red, et. al [7] for review of these techniques.

Region blocking - In this simplest of approaches each region is assigned to one multi-user. The user's feature selection cursor is only allowed to select features that lie within the assigned and constrained user region, such as edges, vertices, faces, etc. The cursor location in model space would be compared against the constraint boundaries and if the cursor intrudes into another user region, the feature in that region would not be selectable.

Region coordination – In this approach a user is allowed to edit features in multiple regions when an administrator assigns design regions to one or more multi-users. Editing stability would probably require that each region have an assigned primary multi-user, with others allowed to edit as secondary users. In a multi-user environment secondary user edits would require a regulating system of methods, priorities and timing constraints to manage the editing sequence and yet avoid the collisions that might occur by simultaneous editing of model features. Consider the transparent adaptation (TA) methods of Zheng [9] and Sun [8] where users negotiate model changes in unconstrained sessions and where data consistency and timing rules avoid chaotic interaction.

At this stage of multi-user decomposition research, we are interested in how to implement *region blocking*, simply because it is the easiest to implement. Region blocking would reduce product development times, and encourage design rationale understanding among multi-users. We are also interested in whether well-known CAx applications can implement these methods through their API set, since multi-user methods will be implemented sooner given CAx vendor support.

Region blocking is also a method for which implementation principles are rather intuitive. We consider general approaches that would enforce constraint boundaries between several multi-users, each assigned to a separate region. How would model feature dependence (dependent, independent, mildly independent, etc.) determine how constraints are applied and enforced?

2.1 Decomposition Regions

Fig. 3 shows an abstract representation of a complex design space divided into multi-user regions that we will refer to as design regions, d_i (i = 1, n). The space or model D is the sum of these regions. Region d_i is the set of features, attributes, operations, and/or geometric elements associated with the design region. Thus, the design space can be represented by $D = \Sigma\, d_i$, assuming that the decomposition is full, i.e., the set of design regions span the entire design space. Because of regional dependencies, the design space representation can be more complex as will be shown later.

The boundaries between two design regions are represented by user constraints c_{jk}, where j,k refers to the constraint between multi-user regions j and k and where $c_{j,k} = c_{k,j}$. User constraints can be represented as geometric equations or feature sets, depending on the design space. Some design regions are defined by only one constraint equation, such as d_1 (constrained by $c_{1,2}$) and d_n (constrained by $c_{i,2,n}$), while others may require comparison against several constraints, such as d_2 (constrained by $c_{1,2}$, $c_{2,i}$, $c_{1,2,n}$) and d_i (constrained by $c_{2,i}$, $c_{1,2,n}$).

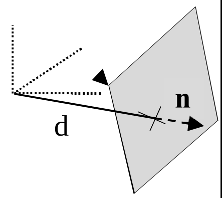
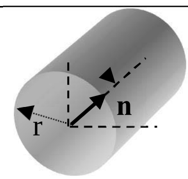
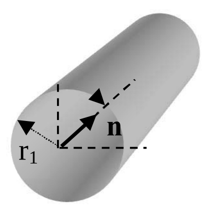
As a simple example consider the problem of developing a manufacturing process plan to machine a part. The part model could be comprised of several feature sets that encompass operations required to manufacture a part. The feature set could be used to decompose the process plan into regions represented by operations like these: 1) roughing tool paths; 2) semi-roughing tool paths; 3) finishing tool paths for surface features; 4) pocketing and slotting features; 5) drilling, tapping, and threading features; 6) profiling features; 7) fixturing hardware and setup; and 8) tooling. Some of these operations are reasonably independent of other operations (e.g., pocketing as compared to drilling/tapping/threading) and could be used to decompose the process planning among several multi-users for simultaneous process planning. When a design region is totally independent of other regions, it will not need a constraint relationship to constrain the assigned user actions and thus $c_{i,..} = 0$. Independent regions, or mildly independent regions, are the best candidates for multi-user decomposition. Regions that are strongly dependent, i.e., connected by dependent features, will require cooperation and intense interaction between the multi-users to simultaneously edit the design space.

Some constraints may be common to more than one multi-user region, e.g., $c_{i,2,n}$ as shown in Fig. 3. For example, a multi-user assigned to region i will be constrained by those relationships that contain i in the constraint subscripts: $c_{2,i}$ and $c_{i,2,n}$. Thus, user design region i is defined by the associated feature set associated and constrained by those functions $c_{i,j,..}$ with included i subscript.

The advantage of multi-user engagement in a process plan is that users more experienced in certain operations could be collaboratively and simultaneously engaged in the design space. It seems that the overall quality of the process plan and finished part will be improved, since several specialists are involved. The opportunity for cross-specialization training is also increased among a multi-user team. Development time would be expected to decrease in proportion to the number of multi-users.

2.2 Examples of Geometric Constraints

Tab. 1 illustrates constraint geometry that could be applied to CAD 3D models or FEA models. Simpler constraint equations in the form of lines and arcs could be developed for 2D applications, but multi-user decomposition is more likely to be used for complex 3D models. If an extrusion is extremely complex, then 2D constraint geometry could be applied similarly using line and arc constraint equations.

Constraint type	Graphical	Constraint Surface	Constraint Forms (IE = inequality; EQ = equality)
Plane (unbounded)		$\mathbf{n}^T\mathbf{p} = d$ **n** = outward plane normal **p** = point in plane d = plane distance	**u** = user selected point $\mathbf{n}^T\mathbf{u} < d$ (inward, IE) $\mathbf{n}^T\mathbf{u} \leq d$ (inward, EQ) $\mathbf{n}^T\mathbf{u} > d$ (outward, IE) $\mathbf{n}^T\mathbf{u} \geq d$ (outward, EQ)
Cylinder (unbounded)		$(\mathbf{p}-\mathbf{v})^T\mathbf{n} = r$ **n** = cyl axis unit vector **v** = point on cyl axis **p** = point on cyl surface r = cyl radius	**u** = user selected point $(\mathbf{u}-\mathbf{v})^T\mathbf{n} < r$ (inward, IE) $(\mathbf{u}-\mathbf{v})^T\mathbf{n} \leq r$ (inward, EQ) $(\mathbf{u}-\mathbf{v})^T\mathbf{n} > r$ (outward, IE) $(\mathbf{u}-\mathbf{v})^T\mathbf{n} \geq r$ (outward, EQ)
Conical Frustum (bounded, reduces to cone if $r_2 = 0$)		$r_c = (\mathbf{p}-\mathbf{v})^T\mathbf{n}$ $(r_2 \leq r_c \leq r_1)$ **n** = cone axis unit vector **v** = point on cone axis at base where $r_c = r_1$ **p** = point on conical surface r_c = cone radius at **p** h = frustum length	**u** = user selected point *Step 1:* $d = (\mathbf{u}-\mathbf{v})^T\mathbf{n}$ *Step 2: Inward* IE: if $(0 < d < h)$ and $r_c = r_1 + d(r_2 - r_1)/h$ $r = \lvert\mathbf{u} - \mathbf{v} - d\mathbf{n}\rvert < r_c$ EQ: if $(0 \leq d \leq h)$ and $r_c = r_1 + d(r_2 - r_1)/h$ $r = \lvert\mathbf{u} - \mathbf{v} - d\mathbf{n}\rvert \leq r_c$ *Step 2: Outward* IE: if $(d < 0)$ or $(d > h)$ or $r = \lvert\mathbf{u} - \mathbf{v} - d\mathbf{n}\rvert > r_c$ given $r_c = r_1 + d(r_2 - r_1)/h$ EQ: if $(d \leq 0)$ or $(d \geq h)$ or $r = \lvert\mathbf{u} - \mathbf{v} - d\mathbf{n}\rvert \geq r_c$ given $r_c = r_1 + d(r_2 - r_1)/h$

Tab. 1: Geometric constraint examples.

The constraints shown in Tab. 1 are a plane, cylinder, and conical frustum (cone a special case) in bounded and unbounded formats as appropriate for the constraint. Other constraints such as (hemi-) spheres, quadratic surfaces, NURBS, etc., could be added later as multi-user sophistication increases. Nevertheless, these few surface types serve to illustrate important constraint concepts.

The third column of Tab. 1 presents a mathematical formulation for each constraint surface. Notice that the vector forms are coordinate system independent and thus can be expressed in any relative part coordinate system. Also note that design regions can be fully or partially enclosed by combinations of these simple constraint surfaces. Full or partial enclosure might be convenient when models have volumetric or dimensioning constraints.

Note that there are inward and outward forms of the constraint equations. Any region constrained by one of these surfaces will use either the inward or outward comparison, depending on which side of the constraint surface the region is located. There is also allowance for a region to include the constraint boundary (EQ: equality constraint forms in Tab. 1) while the adjoining region may be restricted from the constraint boundary (IE: inequality constraint forms in Tab. 1).

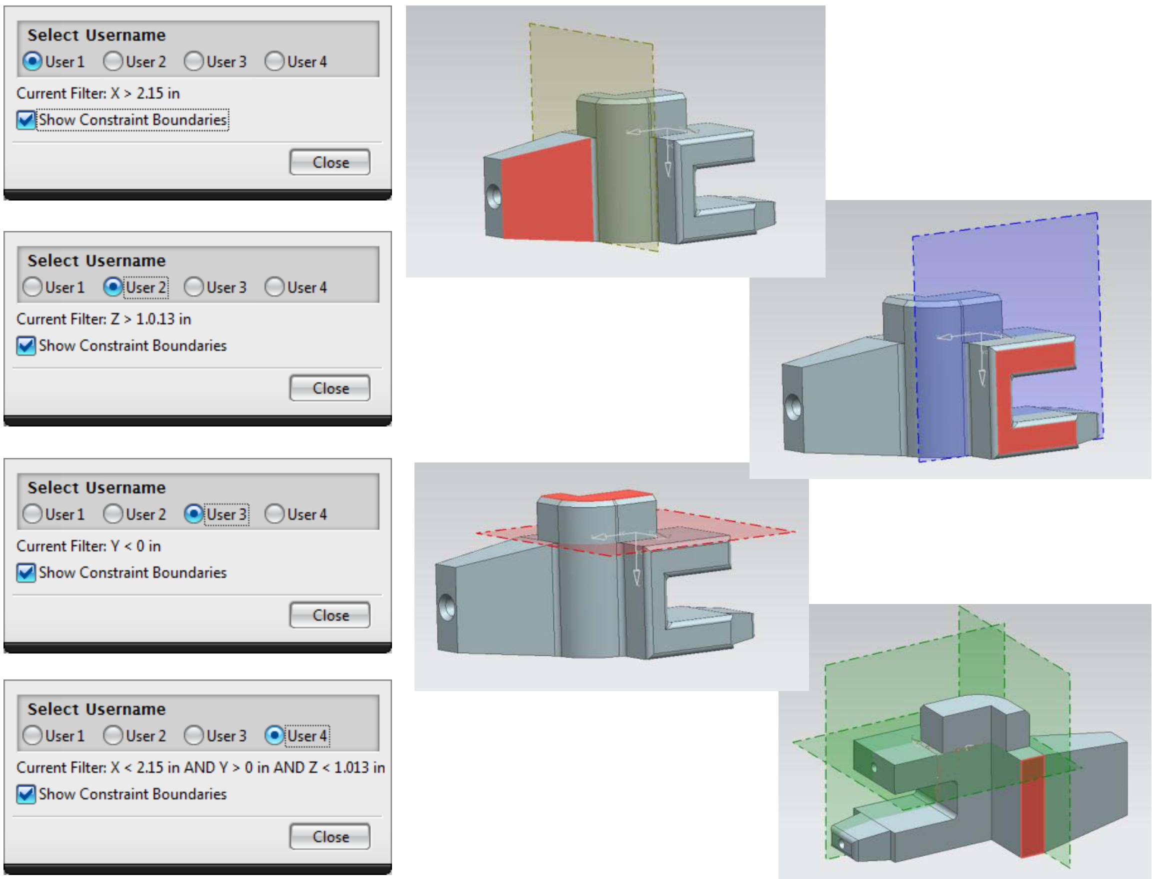


Fig. 4: Geometric constraint limiting of user feature selection.

3 DECOMPOSITION REGION BLOCKING PROTOTYPE FOR CAD MODEL

Marshall [5] recently demonstrated the use of constraints to limit feature selection to CAD regions bounded by planes. Her thesis integrated new methods into the Siemens NX CAD application using C++ coding and NX API’s that would block a user from accessing features in an unassigned region. The user’s cursor could be moved to other regions but selection of features like a point, edge, face, etc. were blocked. Fig. 4 shows features selected by four multi-users users in their assigned region as limited by the current constraint plane filter. If a user tries to move to a region and select a feature in that region, the feature selection is blocked. The following section describes a selection filtering tool that Marshall implemented within NX.

3.1 Selection Filtering Tool

The selection filtering tool is a GUI 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 architectural limitations in NX 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 constants associated with each user.

The selection filtering portion of the implementation is integrated within the CAD system (mouse cursor combined with feature selection ray cast normal to the viewing window) and has a single dialog window that allows for selection among different multi-users. Depending on the user, a selection filter is applied to all possible selections based on four constraint planes; see Fig. 4. 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

$$p \in P \tag{3.1}$$

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

$$x \in X;\, y \in Y;\, z \in Z \tag{3.2}$$

For the model of Figure 4, the following constraints have been implemented using inch units. ACCEPT P means a feature on the model selectable by the multi-user. A selectable feature is one that can be edited by the multi-user in the CAx application.

User 1: only select edges and faces for which

$$\text{if any } x \in X > 2.15,\ \text{ACCEPT } P \tag{3.3}$$

User 2: only select edges and faces for which

$$\text{if any } z \in Z > 1.013,\ \text{ACCEPT } P \tag{3.4}$$

User 3: can select edges and faces for which

$$\text{if any } y \in Y < 0,\ \text{ACCEPT } P \tag{3.5}$$

User 4: can select edges and faces for which

$$\text{if any } x \in X < 2.15 \text{ and } y \in Y > 0 \text{ and } z \in Z < 1.013,\ \text{ACCEPT } P \tag{3.6}$$

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 or off the visible constraint boundaries. The constraint planes placed at the edge of the user's selection boundary are colored differently for each user.

Conclusion – This simple prototype shows that a primary CAx application like NX can be configured through the API to provide regional blocking for multi-user simultaneous editing of a design space, albeit with architectural enhancements to the CAx application.

3.2 Regional Dependence

Marshall's example in Fig. 4 is interesting because it shows the problem when geometric features and related parameters transition a constraint boundary. User 1 is the only multi-user that can edit geometry reasonably independent of the other multi-user regional geometry because the regional geometry is effectively within a single region – see Figure 5. Users other than User 1 edit geometry that transitions adjoining regions. The constraint boundaries chosen do not effectively isolate the regional geometry, mostly because the design model is not very complex. Multi-user collaboration would require methods for *regional coordination* as discussed earlier.

Now consider the front frame for a jet engine shown in Fig. 6. It would seem reasonable to regionally decompose the model into at least 3 regions for simultaneous editing by 3 multi-users. Two cylindrical constraint boundaries of appropriate radius could be used to confine the inner cylinders, the stiffening radial struts, and the outer rim. This decomposition recognizes mildly independent regions. The final design would require a user to merge (blend, fillet, etc.) the decomposed design regions.

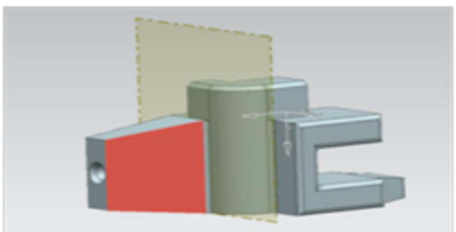


Fig. 5: User 1 edits mildly independent design region.

Fig. 6: Object with axis symmetric design regions.

4 DECOMPOSITION REGION BLOCKING PROTOTYPE FOR FEA MODEL

A multi-user version of CUBIT [14] is called CUBIT Connect, and is being used to test multi-user FEA at BYU's ν-CAx NSF Site. Since we have access to CUBIT source code, user region locking can be done at the source level rather than relying on APIs.

A model or assembly is essentially completed before being passed to an FEA application; therefore, model decomposition is easier to program. The administration system can examine the model and assign regions based on the areas of expertise of the engineering team. An early approach to FEA decomposition uses entity ID tags to allocate workspaces as discussed in Section 4.1.

As mentioned earlier, expertise can vary from member to member of an engineering team. If we consider the meshing of a race car as an example, there may be some team members who are experts in meshing cylindrical regions such as wheels; others may have expertise in meshing wings, or the cockpit area, nose cone, etc. This concept is not new to collaborative engineering as this is how tasks are currently divided in industry. However, with the single user check-in/check-out system, only one user can work on the model at a given time. CUBIT Connect allows multiple users to work on their assigned regions simultaneously, thus bypassing the check-in/check-out system.

4.1 CUBIT Numbering System: Body, Volume, Surface ID

When CUBIT imports a geometry file (eg: STL, IGES), regions of the model/assembly are given body, volume or surface IDs. This inherent regional scheme, though not used to allow users to simultaneously mesh and clean up geometric models can be used to assign multi-users to specific regions and block users from certain regions. Note that similar entity ID methods are used in other CAx tools as well.

In CUBIT a collection of surfaces makes a volume and a collection of volumes makes a body. Figure 7 shows a simple example of how these different IDs relate to each other.

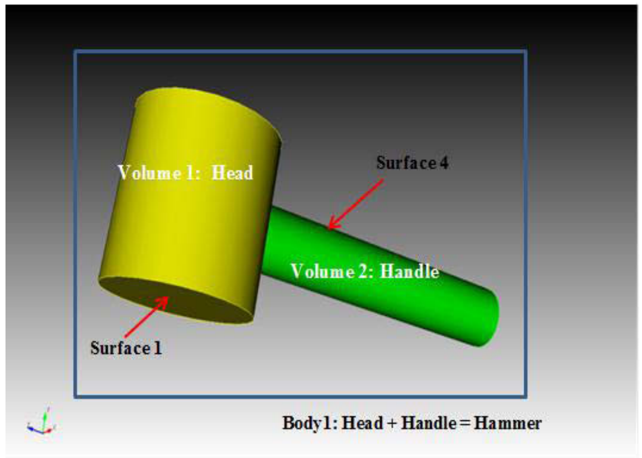


Fig. 7: CUBIT's geometry numbering system.

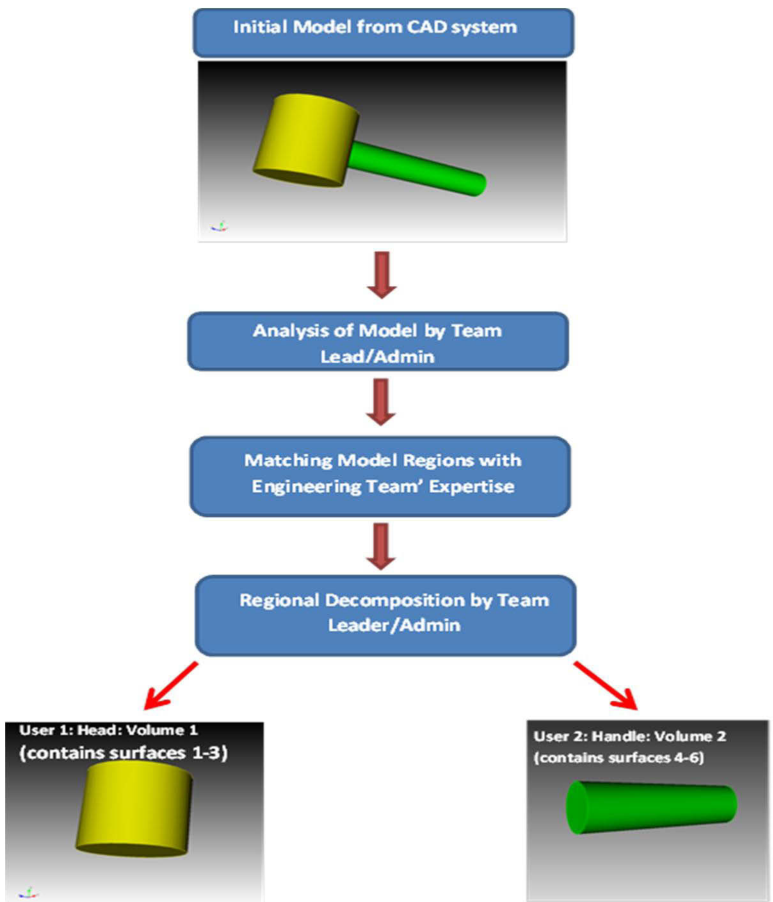


Fig. 8: Regional Decomposition in CUBIT Connect.

MASTER MODEL (PACE CAR)
http://byuracing.byu.edu/content/pace

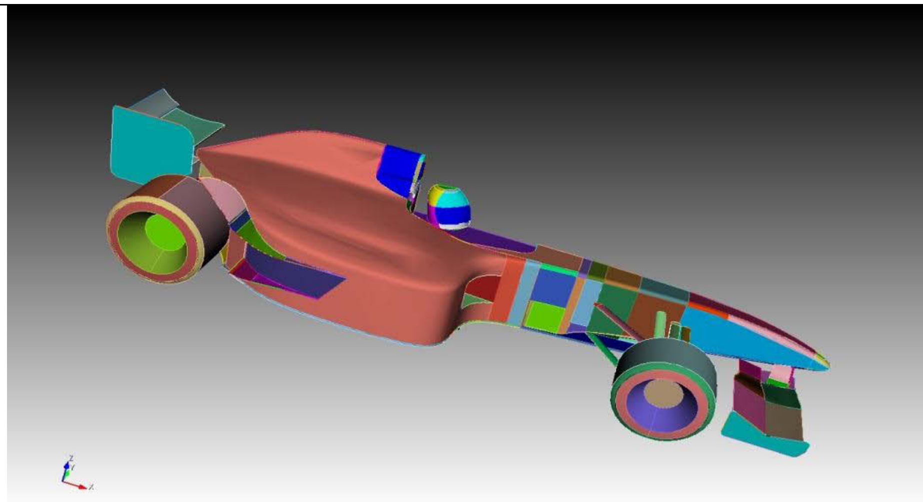
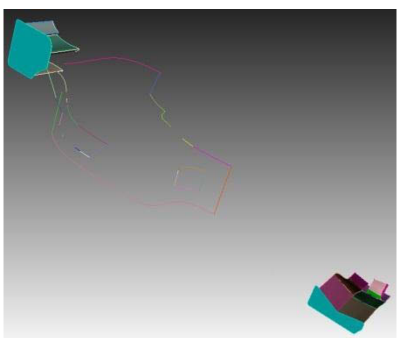
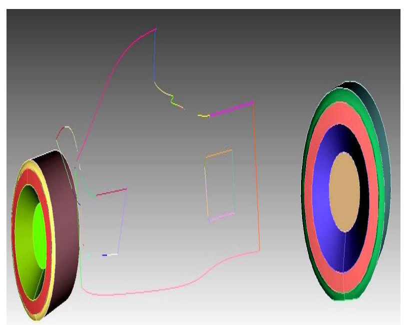
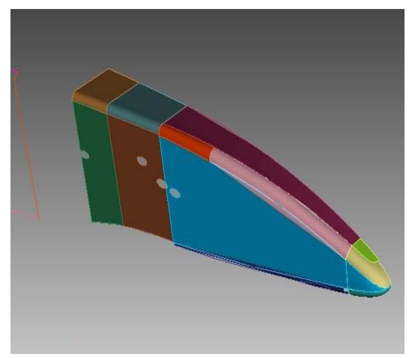
<table>
<tr><td colspan="3"></td></tr>
<tr><td colspan="3">Decomposed Work Spaces in CUBIT</td></tr>
<tr><td>User 1</td><td>User 2</td><td>User 3</td></tr>
<tr><td>Wings
Only allowed to interact with surface IDs
Front Wing (IDs:44-50, 28-32, etc.)
Tail Wing (IDs:354-357, 122, 379, etc.)</td><td>Wheels
Only allowed to interact with surface IDs
Front Wheel (IDs:258, 357-360, 130, etc.)
Back Wheel (IDs:363-366,147, etc.)</td><td>Nose Cone
Only allowed to interact with surface IDs : 21,22, 25,238, 239. 264, 381,etc.</td></tr>
<tr><td></td><td></td><td></td></tr>
<tr><td>User 1 can select only the surfaces that are contained within the wings sections of the car model</td><td>User 2 can select only the surfaces that are contained within the wheels sections of the car model</td><td>User 3 can select only the surfaces that are contained within the nose cone section of the car model</td></tr>
<tr><td colspan="3">All users can see the entire car model regardless of their assigned regions. If they wish, users can isolate the assigned regions as shows in the 3 screenshots above.</td></tr>
</table>

Fig. 9: CUBIT Connect Assignment of a car model using surface ID tags.

4.2 Access Rights using CUBIT's Numbering System

Using CUBIT's numbering system, regions can be assigned to different users and be used to restrict users from modifying other regions. Users can view other multi-user regions, but they cannot select or modify the geometry. In this implementation of regional decomposition, the Engineering Team Lead reviews the model first and determines which volumes, surfaces or bodies should be assigned to a particular user. A region could be assigned by locking the user to a certain range of body, volume or surface ID's.

Once the region is assigned, CUBIT Connect enables mouse cursor interactions only to those regions assigned to a multi-user. The user can select, click, mesh and do any operation within that region. As mentioned, locking does not restrict the user from browsing other regions of the model or assembly. Figure 8 shows how a simple CUBIT model in CUBIT Connect is decomposed among two multi-users, but would work for more complex models with a number of different IDs, such as the racecar example in Fig. 9.

4.3 Further Research

Research is underway to implement a GUI for the administrator to interactively assign workspaces using the different geometry ID's. Furthermore, developers are working on adding user accounts and access rights information to CUBIT Connect. This would make it more secure and user-friendly in a multi-user environment.

Some interesting questions arise from having restricted workspaces in FEA. What happens at a boundary of two workspaces? How can boundaries be merged in a multi-user setting so that the mesh stays consistent through the boundary? How do you handle an action of a user that affects the workspace of some other user?

5 CONCLUSIONS

Functional decomposition is used every day in product development, but the advent of new multi-user tools will change the current paradigms that depend on single user tools. This paper has suggested that for multi-user development to be practically administered, experiential databases will necessarily be used to record and maintain various levels of organizational and personnel experience. The paper has also shown that fundamental regional blocking techniques are feasible in modern CAx applications, such as CAD and FEA. This is an important result because these techniques are more likely to be commercially implemented if they can be practically implemented within CAx applications commonly used by industry.

REFERENCES

[1] Browning, T.: Applying the design structure matrix to system decomposition and integration problems: a review and new directions, IEEE Transactions on Engineering Management, 48, 2001, 292-306.

[2] Herrmann J.: Decomposition in product development, Technical Report 2004-6, Institute for Systems Research, University of Maryland, College Park, 2004.

[3] Ip, C.: Automatic classification of CAD models, M.S. Thesis, Drexel University, April, 2005

[4] Joshi, S.; Chang, T.: Feature extraction and feature based design approaches in the development of design interface for process planning, Journal of Intelligent Manufacturing, 1(1), 1990, 1-15.

[5] Marshall, F.: Model decomposition and constraints to parametrically partition design space in a collaborative CAx environment, Master's Thesis, Brigham Young University, December, 2011.

[6] Martinez, M.; Fouletier, P.; K.H., Park, J.: Virtual enterprise - organisation, evolution and control, Int. J. Production Economics 74, 2001, 225-238.

[7] Red, E.; Holyoak, V.; Jensen, G.; Marshall, F.; Ryskamp, J.; Xu, Y.: ν-CAx: A Research Agenda for Collaborative Computer-Aided Applications, Computer-Aided Design and Applications, 7(30), 2010, 387-404.

[8] Sun, C.; Xia, S.; Sun, D.; Chen, D.; Shen, H.; Cai, W.: Transparent Adaptation of Single-User Applications for Multi-User Real-Time Collaboration, ACM Transactions on Computer-Human Interaction, 13(4), 2006, 531–582.

[9] Zheng, Y.; Shen, H.; Sun, C.: Leveraging single-user AutoCAD for collaboration by transparent adaptation, 2009 13th International Conference on Computer Supported Cooperative Work in Design, Santiago, Chile, April 22-24.

[10] http://www.dsmweb.org/ commercial DSM tools
[11] http://www.eclass.de/ product classification company and system (cross-industry product data standard for classification and clear description of products and services.)
[12] http://e-pro.de/en/products/e-proCAT.php (e-proCAT links tasks like no other software: product data processing, product group recoding, multiple classification, manual and automatic feature recording, MIME type management, product line generation and adjustment, plus image management.)
[13] http://en.wikipedia.org/wiki/ETIM_(standard)
[14] http://www.etim-international.com/Classification/Modelinformation/tabid/292/language/en-US/Default.aspx