Hierarchical Role-based Viewing for Secure Collaborative CAD

Christopher D. Cera† Ilya Braude† Immanuel Comer† Taeseong Kim‡
JungHyun Han‡ William C. Regli†

† Geometric and Intelligent Computing Laboratory
Department of Computer Science
Drexel University
Philadelphia, PA 19104

‡ Computer Graphics Laboratory
School of Information and Communications Engineering
Sung Kyun Kwan University
Suwon, 440-746, Korea
Problem Scenarios

- **Protection of sensitive design information**
  - Designers given “need to know” rights based on legal, intellectual property, or national security requirements.

- **Collaborative supply chains**
  - Design and manufacturing activity is often outsourced.
  - Provide vital design data to one partner while protecting another partner.

- **Multi-disciplinary design**
  - Designers suffer from cognitive distraction when they must interact with unnecessary design details that they do not understand and cannot change.
  - Design space should be simplified if possible.
Approach

- **Role-based Access Control** [Sandhu, 1996]
  
  “Roles are closely related to the concept of user groups in access control. However, a role brings a set of users on one side and a set of permissions on another, whereas user groups are typically defined as a set of users only.”

- Develop a geometric approach to *Information Assurance* using Multiresolution surfaces

- Introduce **Role Hierarchies** and their correspondence to Multiresolution surfaces

- Provide alternatives to the “all or nothing” solution when lack of permissions exists
Related Work

Information Assurance and Security
- Authentication vs. Access Control
- Access Control Policies
  - Discretionary Access Control (DAC) [Lampson, 1971]
  - Mandatory Access Control (MAC) [Bell/La-Padula, 1973]
  - Role-based Access Control (RBAC)

Collaborative Design
- TeamCenter, Windchill, and ENOVIA
- DOME/Oculus Tech. (Product: CO) [Wallace et al., 1998]
- FIPER/Engineous [Roehl, 2000]

Computer Graphics
- Level of Detail
- View-dependent Rendering
- Geometry Compression
Example

<table>
<thead>
<tr>
<th>AR</th>
<th>Gus</th>
<th>Geri</th>
<th>Bob</th>
<th>Kathy</th>
<th>Rob</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manager</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Drivetrain</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Block</td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Support</td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Observer</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
</tr>
</tbody>
</table>
Problem Formulation

- **Entities**
  - set of actors, $A = \{a_0, a_1, \ldots, a_n\}$
  - set of roles, $R = \{r_0, r_1, \ldots, r_m\}$
  - set of security features, $F = \{f_0, f_1, \ldots, f_k\}$

- **Actor-Role Assignment** ($AR$)
  - relation (possibly many-to-many), $AR \subseteq A \times R$

- **Model-Role Assignment** ($MR$)
  - relation (possibly many-to-many), $MR \subseteq b(M) \times R$
  - $MR$ assignment can be simplified using features: $MR \subseteq F \times R$.

- **Role Hierarchy** ($RH$)
  - weighted directed acyclic graph (DAG), $w : H \rightarrow [0, 1]$  
  - $RH = (R, H)$ where $H \subset R \times R$ is the hierarchical set of relationships in $R$
Role-based Viewing Problem

Given:
- a set of roles and their relationships ($R$ and $RH$)
- a solid model and its security features ($M$, $F$, and $MR$)
- an actor ($a$ and $AR$)

Determine:
- the appropriate view $M'$ of model $M$ for actor $a$. 
Role-based Viewing

- Convert solid model $M$ to a high-fidelity mesh representation;

- For each security feature $f$:
  - If actor $a$ and $f$’s roles are the same, then add $f$ to $M'$
  - If actor $a$ and $f$’s roles are different, then use $RH$ to determine how much of $f$ can be viewed

- Return $M'$. 
Hierarchical RBAC Policy

- Hierarchical RBAC:
  - natural means for structuring roles that reflect an organization's lines of authority and responsibility
  - child inherits permissions from parent based upon a multiplicative weight \( w \)

- RBAC does not enforce a policy

- We adopt a MAC policy, using \( \lambda() \) to define the security level, as follows:
  - **Simple Security Property** - Actor \( a \) can read feature \( f \) iff \( \lambda(a) \geq \lambda(f) \). This is also known as the **read-down** property.
  - **Liberal Property** - Actor \( a \) can write feature \( f \) iff \( \lambda(a) \leq \lambda(f) \). This is also known as the **write-up** property.

- Our adaptation of RBAC allows permissions to be modified through \( RH \)
Hierarchical Viewing

- Weights along a path gives the *Degree of Visibility (DoV)*

- Compute graph reachability among all pairs of roles assigned to both $a$ and $f$

- Multiple paths possible, return maximum

<table>
<thead>
<tr>
<th>DoV</th>
<th>$f_0$</th>
<th>$f_1$</th>
<th>$f_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bob</td>
<td>1.0</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>Kathy</td>
<td>1.0</td>
<td>1.0</td>
<td>n/a</td>
</tr>
<tr>
<td>Rob</td>
<td>0.5</td>
<td>0.5</td>
<td>1.0</td>
</tr>
</tbody>
</table>
Multiresolution and LOD

- Meshes are faster than alternatives but requires tessellation

- How to choose the best simplification technique?
  - speed
  - dynamic
  - non-topology preserving
  - boundary preserving
  - view-independence

- Quadric Error Metrics (QEM) [Garland/Heckbert, 1997]

- User-guided simplification [Kho/Garland, 2003]
### Example: Motorcycle Engine – Project Manager

<table>
<thead>
<tr>
<th>AR</th>
<th>Gus</th>
<th>Geri</th>
<th>Bob</th>
<th>Kathy</th>
<th>Rob</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manager</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Drivetrain</td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Block</td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Support</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>Observer</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
</tr>
</tbody>
</table>

![Motorcycle Engine Diagram]

- Engine Block
- Crankshaft
- Gear
- Piston
Example: Motorcycle Engine – Lead (Drivetrain)

<table>
<thead>
<tr>
<th>AR</th>
<th>Gus</th>
<th>Geri</th>
<th>Bob</th>
<th>Kathy</th>
<th>Rob</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manager</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Drivetrain</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Block</td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Support</td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Observer</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
</tr>
</tbody>
</table>

High Detail Internals

Low Detail Outer Shell
Example: Motorcycle Engine – Lead (Block)

<table>
<thead>
<tr>
<th>AR</th>
<th>Gus</th>
<th>Geri</th>
<th>Bob</th>
<th>Kathy</th>
<th>Rob</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manager</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Drivetrain</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Block</td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Support</td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Observer</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
</tr>
</tbody>
</table>

Simplified Crankshaft

Simplified Piston

Omitted Detail
Example: Motorcycle Engine – Support (Block)

<table>
<thead>
<tr>
<th>AR</th>
<th>Gus</th>
<th>Geri</th>
<th>Bob</th>
<th>Kathy</th>
<th>Rob</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manager</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Drivetrain</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Block</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>Support</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>Observer</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
</tr>
</tbody>
</table>

Simplified Crankshaft
Simplified Piston
Omitted Detail
Example: Motorcycle Engine – Observer (Block)
Implementation

- **Framework for Access-control in Computer Aided Design Environments (FACADE)**
  - C++, OpenGL
  - Win32, Linux, Solaris
  - Network-enabled, access via *thin* or *fat* clients

- **QEM**
  - collapses only connected vertices (more restrictive)
  - combine parts or connected regions with equivalent *MR*

- **Modes**
  - Role-authoring (*AR* and *MR*) Mode
  - Designing Mode
Contributions and Future Work

- Developed a geometric approach to *Information Assurance* using geometry processing techniques
- Introduced **Role Hierarchies** and their correspondence to Multiresolution surfaces
- Provided alternatives to the “all or nothing” permissions problem

**Other Directions:**
- Feature-based Envelopes [Shyamsundar/Gadh, 2002]
- Role-based Viewing Envelopes [Cera et al., 2003]

**Other Surfaces:**
- Superquadrics [Nishino et al., 1999]
- Wavelet Decomposition of B-Splines [Kazinnik/Elber, 1997]
Q & A

Geometric and Intelligent Computing Laboratory:
http://gicl.cs.drexel.edu

Computer Graphics Laboratory:
http://graphics.skku.ac.kr

This work was supported in part by:

- National Science Foundation (NSF) Knowledge and Distributed Intelligence in the Information Age (KDI) Initiative Grant CISE/IIS – 9873005
- CAREER Award CISE/IIS – 9733545
- Office of Naval Research (ONR) Grant N00014 – 01 – 1 – 0618
- Korea Research Foundation Grant KRF – 2001 – 013 – E00073