This video and all related information and materials ("materials") are owned by Carnegie Mellon University. These materials are provided on an "as-is" "as available" basis without any warranties and solely for your personal viewing and use.

You agree that Carnegie Mellon is not liable with respect to any materials received by you as a result of viewing the video, or using referenced websites, and/or for any consequences or the use by you of such materials.

By viewing, downloading, and/or using this video and related materials, you agree that you have read and agree to our terms of use (www.sei.cmu.edu/legal/).

Distribution Statement A: Approved for Public Release; Distribution is Unlimited

What did the Jeep experience illustrate

Running out of date software
Wide attack surfaces
Not understanding or appreciating interconnections
Components used in new operational situations
Assumed or misunderstood authentication and authorization needs
Lost opportunity to mitigate damage through disclosure
“Security through obscurity” is not enough
Catching software faults early saves money

Faults accounts for 30–50% percent of total software project costs

Software Development Lifecycle

<table>
<thead>
<tr>
<th>Where Faults are Introduced</th>
<th>70%</th>
<th>20%</th>
<th>10%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirements Engineering</td>
<td><img src="image1.png" alt="Image" /></td>
<td><img src="image2.png" alt="Image" /></td>
<td><img src="image3.png" alt="Image" /></td>
</tr>
<tr>
<td>System Design</td>
<td><img src="image4.png" alt="Image" /></td>
<td><img src="image5.png" alt="Image" /></td>
<td><img src="image6.png" alt="Image" /></td>
</tr>
<tr>
<td>Software Architectural Design</td>
<td><img src="image7.png" alt="Image" /></td>
<td><img src="image8.png" alt="Image" /></td>
<td><img src="image9.png" alt="Image" /></td>
</tr>
<tr>
<td>Component Software Design</td>
<td><img src="image10.png" alt="Image" /></td>
<td><img src="image11.png" alt="Image" /></td>
<td><img src="image12.png" alt="Image" /></td>
</tr>
<tr>
<td>Code Development</td>
<td><img src="image13.png" alt="Image" /></td>
<td><img src="image14.png" alt="Image" /></td>
<td><img src="image15.png" alt="Image" /></td>
</tr>
<tr>
<td>Unit Test</td>
<td><img src="image16.png" alt="Image" /></td>
<td><img src="image17.png" alt="Image" /></td>
<td><img src="image18.png" alt="Image" /></td>
</tr>
<tr>
<td>Integration</td>
<td><img src="image19.png" alt="Image" /></td>
<td><img src="image20.png" alt="Image" /></td>
<td><img src="image21.png" alt="Image" /></td>
</tr>
<tr>
<td>System Test</td>
<td><img src="image22.png" alt="Image" /></td>
<td><img src="image23.png" alt="Image" /></td>
<td><img src="image24.png" alt="Image" /></td>
</tr>
<tr>
<td>Acceptance Test</td>
<td><img src="image25.png" alt="Image" /></td>
<td><img src="image26.png" alt="Image" /></td>
<td><img src="image27.png" alt="Image" /></td>
</tr>
<tr>
<td>Operation</td>
<td><img src="image28.png" alt="Image" /></td>
<td><img src="image29.png" alt="Image" /></td>
<td><img src="image30.png" alt="Image" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Where Faults are Found</th>
<th>3.5%</th>
<th>16%</th>
<th>50.5%</th>
<th>9%</th>
<th>20.5%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nominal Cost Per Fault for Fault Removal</td>
<td><img src="image31.png" alt="Image" /></td>
<td><img src="image32.png" alt="Image" /></td>
<td><img src="image33.png" alt="Image" /></td>
<td><img src="image34.png" alt="Image" /></td>
<td><img src="image35.png" alt="Image" /></td>
</tr>
</tbody>
</table>

Sources: *Critical Code; NIST, NASA, INCOSE, and Aircraft Industry Studies*
Security is implemented across life cycle

Sustainment

Engineering and Development

Requirements and Acquisition

Mission thread (Business process)
Threat Analysis
Abuse Cases
Architecture and Design Principles
Coding Rules and Guidelines
Testing, Validation and Verification
Monitoring
Breach Awareness

Deployment and Operations
Polling Question

What tools do you use to support secure development?

Security requirements management tool?

Source code analyzers?

Dynamic fuzz or penetration testing?

Others?
Room for improvement

More than 81% do not coordinate their security practices in various stages of the development life cycle.

Cross life cycle issues

- Mission thread (Business process)  
  - Threat Analysis
  - Abuse Cases

- Engineering and Development  
  - Architecture and Design Principles
  - Coding Rules and Guidelines
  - Testing, Validation and Verification

- Automation (DevOps)

- Deployment and Operations  
  - Monitoring
  - Breach Awareness

- Requirements and Acquisition  
  - Acquisition (Supply chain)
  - Building skills (Workforce development)
  - Metrics, Models, and Measurement
Contact Information

Mark Sherman
(412) 268-9223
mssherman@sei.cmu.edu

Web Resources (CERT/SEI)
http://www.cert.org/
http://www.sei.cmu.edu/