



# Simplifiers and Complicators

**CS 577a**

**Dan Port, USC**

**2002**

# Unmet Expectations Problems

- **LCO success condition**
  - Describes at least one feasible architecture
  - Satisfying requirements within cost/schedule/resource constraints
  - Viable cost-effective business case
  - Stakeholder concurrence on key system parameters
- **Projects That Failed LCO Criteria**
  - 1996: 4 out of 16 (25%)
  - 1997: 4 out of 15 (27%)

**why?**

# Requirements and Expectations: Domain Model Clashes

- **Easy/hard things for software people**

**“If you can do queries with all those ands, ors, synonyms, data ranges, etc., it should be easy to do natural language queries.”**

**“If you can scan the document and digitize the text, it should be easy to digitize the figures too.”**

- **Easy/hard things for librarians**

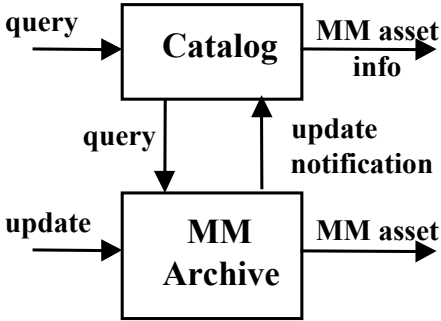
**“It was nice that you could add this access feature, but it overly (centralizes, decentralizes) control of our intellectual property rights.”**

**“It was nice that you could extend the system to serve the medical people, but they haven’t agreed to live with our usage guidelines.”**

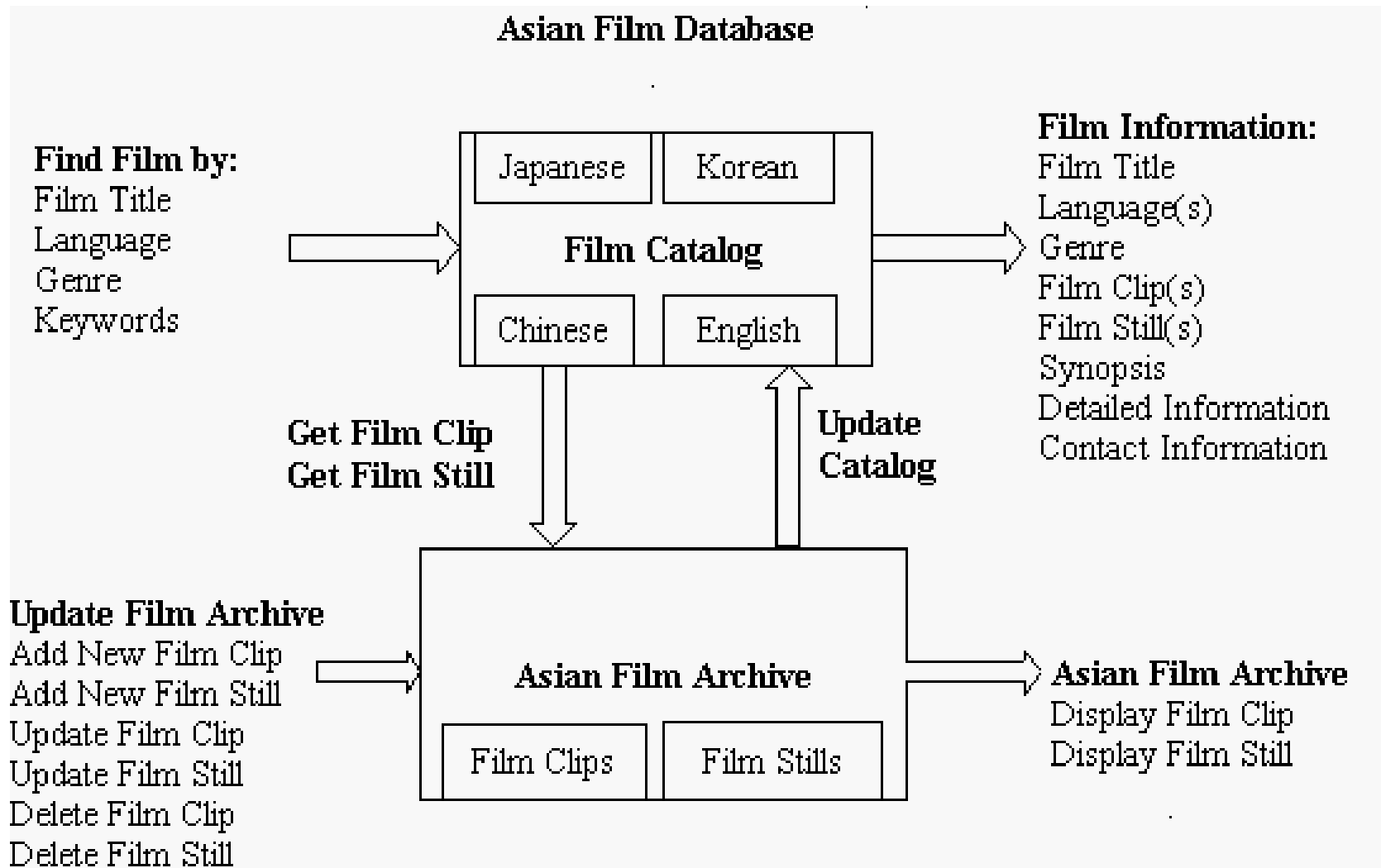
# 1998 Simplifier/Complicator Experiment

- **Identify application simplifiers and complicators**
  - For each digital library sub-domain
  - For both developers and clients
- **Provide with explanations to developers and clients**
  - Highlight relation to risk management
- **Homework exercise to analyze simplifiers and complicators**
  - For two of upcoming digital library projects
- **Evaluate effect on LCO review failure rate**

# Example S&C's

Type of Application	Simple Block Diagram	Examples	Simplifiers	Complicators
<p><b>Multimedia Archive</b></p>	 <pre> graph TD     Query((query)) --&gt; Catalog[Catalog]     Update((update)) --&gt; MM[MM Archive]     Catalog -- "MM asset info" --&gt; Out1(( ))     MM -- "MM asset" --&gt; Out2(( ))     Catalog -- "query" --&gt; MM     MM -- "update notification" --&gt; Catalog             </pre>	<p>1, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 20, 31, 32, 35, 36, 37, 39</p>	<ul style="list-style-type: none"> <li>• Use standard query languages</li> <li>• Use standard or COTS search engine</li> <li>• Uniform media formats</li> </ul>	<ul style="list-style-type: none"> <li>• Natural language processing</li> <li>• Automated cataloging or indexing</li> <li>• Digitizing large archives</li> <li>• Digitizing complex or fragile artifacts</li> <li>• Rapid access to large Archives</li> <li>• Access to heterogeneous media collections</li> <li>• Automated annotation/description/ or meanings to digital assets</li> <li>• Integration of legacy systems</li> </ul>

# Specialized S&C's



Simplifiers	Risks and Trade-offs
<p><b>Generic</b> Uniform Media Formats</p> <p><b>Specific</b> All video clips are stored using an open file format for video/audio (e.g., MPEG). All film stills are stored using an open image file format (e.g., JPEG). The inverse complicator is to store film clips using streaming video technologies</p>	<p>This means that we may have to convert existing digital assets or digitize the original media, which may be costly.</p> <p>A unique file format limits the user base to those who have viewers for that particular file format</p> <p>The chosen file format may not be the most efficient for the various types of media (in terms of compression rates, quality, etc...)</p>
<p><b>Generic</b> Use Standard Query Languages</p> <p><b>Specific</b> Organize catalog and archive relationally so that queries will be limited to standard search formats,; match exactly by value on any of the fields with or without using boolean combinations (AND, OR, NOT, etc...), or using pattern matching (SQL <i>LIKE</i> keyword)</p>	<p>May not be as effective for "discovering" assets in the archive: users must know what they're looking for, in order to search for it</p>
<p><b>Generic</b> Use Standard COTS</p> <p><b>Specific</b> Use a standard Relational Database Management System (RDBMS) that supports storing multi-media assets</p>	<p>A Relational Database Management System may not be most suited for archival of multi-media assets.</p> <p>A Relational Database Management System may have a high initial cost, high implementation, and high administration cost (requires specialized knowledge skills)</p>

<b>Complicators</b>	<b>Risks and Trade-offs</b>
<p><b>Generic</b> Natural Language Processing</p> <p><b>Specific</b> Store the information only in one language (e.g., English) and provide dynamic translation into Chinese, Japanese and Korean The inverse simplifier is to store the same information in 4 different languages (English, Chinese, Japanese and Korean).</p>	<p>The first approach is a complex, error-prone, expensive natural language processing issue</p> <p>The second approach will require more storage space, in addition to acquiring the translations</p>
<p><b>Generic</b> Digitizing Large Archives</p> <p><b>Specific</b> Digitizing film clips from the entire collection of films (which grows at a very fast rate of 800 films per year for Indian films alone)</p>	<p>If each film clip requires around 10 MB, then the rate of growth of the database will be of 8GB a year (exclusive of catalog information)</p>
<p><b>Generic</b> Integration of "Legacy" Systems</p> <p><b>Specific</b> Do not require Real-Video plug-in for Web browsers to allow users to view streamed film clips</p>	<p>We cannot use more effective multi-media formats, which are becoming standard technologies</p>

# The Results

- **Projects That Failed LCO Criteria**
  - **1996: 4 out of 16 (25%)**
  - **1997: 4 out of 15 (27%)**
  - **1998: 1 out of 20 (5%)**
  - **1999: 1 out of 22 (4%)**
- **40% of Student critiques cited S&C's as helpful (and more since)**
  - **In focusing on achievable requirements set within tight schedule**
  - **In understanding project risks and tradeoffs**