As we know …

- Requirements are in problem domain terms; architecture often in solution domain terms.

- Rationale determines the mapping between the functional and non-functional requirements and the architecture.

- Requirements $\rightarrow$ (rationale) $\rightarrow$ Architecture

As we know …

- Rationale tells "WHY"
  - Criteria
  - Plans
  - Alternatives
  - Non-Functional Requirements

- Capturing Design Rationale is a key aspect in effective Communication (inter-organization), effective Documentation (external record), and system evolution
Why Formalize Design Rationale?

- Formalization will/can lead to identification of weak or inconsistent design logic
  - The development of more powerful/succinct SW
- Formalization will provide ideal mechanism for indexing/retrieving the design
  - Without some form of retrieval mechanism, Design Rationale capture is useless
  - Effective retrieval mechanism will reduce future maintenance costs

Design Rationale Roots

DR Roots (Formalizing Argumentation)

- Goal: A system in which creation and retrieval of trails of relevant information mimics human reasoning & memory
- Key Hypothesis: By making the structures of arguments explicit, they can be more rigorously constructed, explored, and communicated

DR Roots (Formalizing Argumentation)

- Formal. (ie. full written documentation)
  - Overly costly
  - Can bog down creative thought & development
- Informal. (ie. memos, email, meetings etc)
  - Created by numerous everyday artifacts
  - Difficult to coalesce into single coherent artifact
- Semi-Formal. (approaches to be discussed)
  - Provides framework without limiting cognitive exploration
### DR Roots (Representing Concepts)

- Hypertext capabilities
  - Auto-links, Pop-up Formats, Coloring, etc
  - Strikes a balance between computational needs (a structure) and natural human thought process

### DR Roots (Research and SW Support)

- Research into the development of computational support for reasoned discourse or argumentation
  - Toulmin – (1958) analyzed and depicted the structure of argumentation graphically
  - Engelbert – (1963) one of the first to envision the use of technology for manipulating “concept structures”
  - Kunz and Rittel – (1970) developed argumentative approaches to design (IBIS - Issue-Based Information System)
  - Xerox PARC – (1987) Argnoter tool (argumentation spreadsheet) for representing arguments in group design

### IBIS - Issue-Based Information System

- First true approach to representing design rationale. (1970’s)
- 3 components the model
  - Issues = Design Questions
  - Positions = Answers to the Design Questions
  - Arguments = Either Support or Refute Positions
- Used successfully in a number of non-engineering applications.
- glIBIS, tailored IBIS and PHI : attempts to use IBIS to model engineering design.
IBIS – Benefits / Drawbacks

- **Benefits**
  - (+) Captures human reasoning
  - (+) Foundation for numerous other models
  - (+) Highlighted importance of distinctly addressing rationale in a design process

- **Drawbacks**
  - (-) Extremely complex and hard to parse
  - (-) No mechanism for associating Issues to a design decision
  - (-) No mechanism to control/focus discussions
  - (-) Only models/captures design questions that can be deliberated (directly design related issues)

---

IBIS – graphical IBIS

- Hypertext system of the IBIS model
- Simplified the capture of design rationale
- Used the IBIS structure and Link types

---

glIBIS – improvements over IBIS

- Graphical / hypertext interface
- Provides mechanism to stop deliberation
- Provides “other” nodes/links to enable thoughts to be expressed outside of the IBIS framework/rule set
- Provides “external” node to link non-IBIS artifacts (ie. requirement documents, diagrams, code, etc)
- Permits Positions to “modify” other positions; Arguments to “modify” other arguments

---

Tailoring IBIS

- Addresses the shortfall of IBIS in not tying Issues to a Design Decision
EXAMPLE: Tailored IBIS-based Notation

Artifacts are listed in **bold**, I=Issue, A=Argument, J=Justification

PHI – Procedural Hierarchy of Issues

- Offshoot of IBIS and addresses IBIS shortfalls:
  - Lacking a mechanism to focus discussion
  - Lacking ability to model/discuss non-design issues
- Issues can “serve” to resolve another Issue (ie. a hierarchy of issues/work)
- Issues that do not serve the stated aims of the design are (can be) thrown out as irrelevant
  - Also for previous issues that become “stale”

Overview Benefits of IBIS/gIBIS/PHI

- Provides qualitative decision support tool
- Methods do decompose issues into sub-issues
- Can help to tailor conversations
- Provides means to serve as group memory
- Note: care must be taken not to unduly prolong the design process

DRL – Decision Representation Language

- Major Components:
  - ISSUE: represents the problem that admits alternatives
  - ALTERNATIVE: represents an option being considered
  - GOAL: represents a desirable property used for comparing the alternatives and is elaborated further in terms of its sub-goals
  - Every relation in DRL is a subclass of a CLAIM
- Possible to manipulate different decisions to see shortfalls/ramifications before commitment
  - Dependency, Precedence, Viewpoint, Plausibility Management
DRL – Decision Representation Language

- Has been used in computer-related applications and implemented in a system called SIBYL
  - Provides graphical user interface for authoring DR
  - Provides computational services such as dependency management, view management, etc.
- Increased vocabulary enables making design deliberation explicit which in other models were handled with exceptions to the language
  - Therefore, is significantly more complex model
- Meant to be constructed asynchronously (ie. historical record of deliberations/rationale)

QOC – Questions, Options, Criteria

- Similar to IBIS
  - Questions = Design Questions
  - Options = Various Answers to Design Questions
  - Criteria = Provides way to weigh various Options
- Clarifies how a particular design decision sits with respect to alternatives (defines the Design Space)
- Constructed synchronously with design process
  - * Develops a logically coherent representation
  - * Depicts “HOW” the design space was explored
QOC – Example (High Level Model)

QOC – Example (Criterion Tree)

QOC – Example (Decision View)

Observed Benefits of QOC

- Serves as a record of what’s been discussed
- Includes both constructive and “deadend” ideas
- Encourages greater completeness in evaluating options
- Structure identifies task-relevant information (and missing assessments as dangling links)
- The process of constructing QOC model helps clarify vague, unarticulated ideas
- Having all the rationale (the complete process) laid out helps in final decision making
- A relatively simple model
QOC vs …

- **vs IBIIS / glIBIS**
  - IBIIS arguments only implicitly refer to “Arguments”
  - QOC Options are specifically design options

- **vs DRL**
  - QOC maps closely to DRL
  - Much simpler and composed at “run time”

### Comparing the Models

<table>
<thead>
<tr>
<th>Approach</th>
<th>Reusability</th>
<th>Process Capture</th>
<th>Computational Support</th>
<th>Authoring Overhead</th>
</tr>
</thead>
<tbody>
<tr>
<td>gIBIS</td>
<td>(-) Capture of design only: little abstraction of issues</td>
<td>(+) Constructed synchronously with design process</td>
<td>(ok) Small vocabulary yields simplicity</td>
<td>(-) Structuring conversations IAW IBIIS rhetoric is not natural</td>
</tr>
<tr>
<td>Tailored IBIIS</td>
<td>(+) Improved rationalization over gIBIS</td>
<td>(+) Most types of arguments are able to be represented</td>
<td>(+) Representations of design artifacts can be easily integrated</td>
<td>(+) More expressive notation may take longer to construct</td>
</tr>
<tr>
<td>DRL</td>
<td>(+) Can assess decisions tradeoffs</td>
<td>(+) Procedural Management</td>
<td>(+) Services for managing, linking, and retrieving design rationale</td>
<td>(-) Large/expressive notation may take longer to construct</td>
</tr>
<tr>
<td>QOC</td>
<td>(+) Design Space Analysis relates multiple designs in comparable form</td>
<td>(+) Capture filters out narrative elements of the design process</td>
<td>(ok) Small vocabulary yields simplicity</td>
<td>(+) Simple but capturing complete design space analysis can take much effort</td>
</tr>
</tbody>
</table>

### General Issues with DR Models

- In order for a model to be useful a balance must be maintained between human usability, machine usability, and expressiveness
- There is a tradeoff between expressiveness and real-time demands
- It is important to makes clear what the intended use of the model is
- A representation must allow the users to say exactly what they want to
- Users must be immediately motivated to record design rationale
Why has Design Rationale not caught on?

- Successes in capturing design rationale have been few and usually associated with a sole DR champion
- Capturing design Rationale is “costly”
  - Requires a large investment of time
  - Serves no immediate need
- Capture often results in loss of info/subtleties
  - Must divide rationale into discrete “chunks”
  - Categorize and assign attributes to the “chunks”
  - Relate “chunks” to each other

Questions?
(Hopefully, that we can answer)