Introductions

• **Tell us**
  - Who you are
  - Educational / Professional Background
  - Research interests
  - Past experiences with architecture knowledge management
  - Expectations of tutorial
Tutorial Goals

- Comprehend the role and challenges involved in managing software architecture knowledge
- Understand domain modeling techniques for characterizing architecture knowledge
- Gain knowledge of the process for capturing and managing architecture knowledge
- Learn how to utilize architecture knowledge to improve the architecture design and evaluation process
- Appreciate the processes and tools required to build and manage a knowledge repository

Overview

- Section 1 – Architecture knowledge management
- Section 2 – A Tool and Approaches to Managing architecture knowledge
- Section 3 – Domain modeling for architecture knowledge
- Section 4 – Utilizing architecture knowledge during design and evaluation
Architecture Knowledge Management

• What is architecture knowledge?
• What is architecture knowledge management?
• Why is architecture knowledge management important?
  – Challenges
  – Strategies
  – Expected benefits
• State-of-the-art
  – Tooling support
  – R&D
• Case study description

Architecture Knowledge and its Management

• Architecture knowledge
  Software architecture design knowledge can be characterized by information that supports software architecture processes.
• Architecture knowledge management
  Software architecture knowledge management is an approach to improving software architecture process outcomes by introducing practices for identifying and capturing architecture knowledge and expertise, and making it available for reuse across projects.
Architecture Knowledge Management Issues

• Unavailability of architecture design knowledge
  – System evolution becomes hard
  – Difficult to identify design errors
• Use of COTS without fully understanding the assumptions
• Situation is worst in case of FLOSS components
• Lack of support in terms of practices and tools
  – What types of architecture knowledge are useful?
  – How to store and manage the knowledge?
  – Make knowledge capture cost-effective

Categories of Factors

• Organization
  – Leadership
  – Processes
  – Platform
  – Structure/Culture
• Human/Behavioral
  – Training
  – Common language
  – Culture
  – Motivation
• Technological
  – KMSS
  – Communication
  – Collaboration
  – Coordination
Knowledge Management Strategies

• Codification
  - High quality, reliable, and rapid use of knowledge
  - High IT investment
  - Rewards for sharing & reusing codified knowledge

• Personalization
  - Connecting people & channelling individual expertise
  - Moderate IT investment
  - Reward for sharing knowledge face to face

• Hybrid approach
  - Codify organization-wide knowledge
  - Personalize local & context specific knowledge

Knowledge Management (KM) Task Model

• KM task model consists of eight tasks
• Strategic tasks
  - Architecture knowledge goals
  - Architecture knowledge measurement
• Operational tasks
  - Architecture knowledge identification
  - Architecture knowledge acquisition
  - Architecture knowledge development
  - Architecture knowledge distribution
  - Architecture knowledge use
  - Architecture knowledge preservation
Supporting Knowledge Management Tasks

Improve management & reusability of architectural assets

Various Metrics to track the usage & growth of reusable architectural knowledge

Knowledge extraction and acquisition techniques

Architecture Knowledge Goals

Architecture Knowledge Measurement

Architecture Knowledge Identification

Architecture Knowledge Use

Architecture Knowledge Preservation

Architecture Knowledge Distribution

Architecture Knowledge Acquisition

Architecture Knowledge Development

Metrics for Knowledge Management

Some of the measures that can be used for most of the KM initiatives

Outcome

- Time, money or personal time saved
- Percentage of successful project compared to those before KM
- Time and number of employees trained

Output

- Usefulness surveys where users evaluate how useful KM initiative have been in helping them accomplish their objective
- Usage anecdotes where users describe how the initiative has contribute to their daily tasks and overall efficiency and effective of organisation

System

- Dwell time per page or section
- Latency (response time)
- Number of downloads
- Number of site access
- Usability survey
- Frequency of use
- Navigation path analysis
- Number of helpdesk calls
- Number of users
- Frequency of use and/or update
- Percentage of total employees using system
Benefits of Architecture Knowledge Management

- Expected benefits:
  - Capture rationale for architecture decisions
  - Help build architectural capabilities
  - Improve architectural reusability
  - Support better quality decisions
  - Minimize architecture risks
  - Minimize reoccurrence of design mistakes
  - Avoid dependency on key individuals
  - Gain competitive advantage
  - Encourage best architectural practices
  - Improve efficiency of architectural processes
  - Support case-based reasoning

Knowledge Management Benefits Tree

Source: David Skyrme associates
A Few Useful Tools

- **Protégé**
  - Creation & manipulation of Ontologies for domain modeling
  - Knowledge acquisition & management
- **Chimaera**
  - Creating & maintaining distributed Ontologies
  - Support for loading, browsing, reorganizing Ontologies
- **Swoop**
  - Hypermedia inspired Ontology editor
- **BRedB**
  - Support for capturing and contextualizing knowledge
  - Knowledge management and reuse

R&D in Architecture Knowledge

- Grady Booch – Handbook of software architecture
- Philippe Kruchten – Design decision ontology and use case model for tool support
- GRIFFIN Project – Architecture knowledge modeling, representation, and acquisition approaches
- Anton Jansen & Jan Bosch – Describing architecture as design decisions in Archium
- Antony Tang & Jun Han – Architecture rationale and elements linkage (AREL) for capturing and traversing rationale to support design decision reasoning
A Preview of our R&D

- Repository of architecture knowledge
  - templates for capturing design elements and diagrams
- Tool support for design decision Ontology and Use Cases
- Architecture knowledge model, Use case model, and knowledge acquisition approaches
- Describing architecture as design decisions using templates – different levels of abstractions
- Our approach complements AREL’s UML based rationale with templates and search mechanism

Case Study Description

- Organizational context
- Domain description
- Software architecture evaluation challenges
- Software architecture knowledge management initiative
- Logistics
- Progress so far!
Summary of Section 1

- What is architecture knowledge?
- What is architecture knowledge management?
- Why is architecture knowledge management important?
  - Expected benefits
  - Challenges
  - Strategies
- State-of-the art
  - Tooling support
  - R&D
- Case study description

A Tool and Approaches for Managing Knowledge

- An Introduction to BRedB – An Architecture Knowledge Management Infrastructure
  - Architecture
  - Features to support architecture design, documentation, and evaluation activities
- Capturing software architecture knowledge
  - Approaches
  - Advantages and disadvantage
- Using BRedB on the case study
Tool for Managing Architecture Knowledge

- **BRedB**: A Process-based Software Architecture Knowledge Management Infrastructure
  - A model of architecture knowledge management
  - Templates for capturing and structuring architecture knowledge
  - Repository of architectural knowledge and experiences
  - Resource for software architects
  - Source of adaptive software architecture processes
  - Support for owning technical and process knowledge

Component View of PAKME Architecture

- User Interface
- Knowledge Management
- Knowledge Search
- Reporting
- Data Management
- Knowledge Repositories
GUI for Start Using BRedB

Support for Architecture Design

- Support for case-based reasoning by human expert
- Repository of reusable architectural artifacts
- Capture/access rationale for design decisions
- Catalogue of architecture and design patterns/tactics
- Search architectural artifacts and knowledge
Capturing and Reusing Cases of Design Decisions

Design Decision Captured as a Case
Searching Design Decision Cases

Using a Design Decision Case
Navigating the Knowledge Base

Template for Capturing and Representing Patterns
Support for Architecture Documentation

- Templates for documenting design decisions
- Store architectural models and documents
- Support for standards such as IEEE 1471-2000
- Represent architectural decisions using views
- Attach process knowledge to architectural artifacts

Structuring and Representing Architecture Decisions
Support for Architecture Evaluation

- Repository of general scenario to support QAWs
- Capture rationale and contextual information surrounding design decisions
- Search and view rationale for previous design decisions
- Documenting findings of evaluating design decisions
- Categorize findings in suitable risk themes
- Generate evaluation reports for the management

General Scenario Captured in BRedB

<table>
<thead>
<tr>
<th>General Scenario Listing</th>
<th>Proposed General Scenarios</th>
<th>Accepted General Scenarios</th>
<th>Rejected General Scenarios</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date 5/22/2007</td>
<td>SQL - 31</td>
<td>SQL - 31</td>
<td>SQL - 31</td>
</tr>
<tr>
<td>ICSE 2007, Minneapolis, USA</td>
<td>SQL - 31</td>
<td>SQL - 31</td>
<td>SQL - 31</td>
</tr>
</tbody>
</table>
Utility Tree of Concrete Scenarios

Viewing Details about a Concrete Scenario
Documenting and Viewing Findings

Support for Architecture-Centric Development
Capturing Architecture Knowledge

A conceptual framework of capturing implicit architecture knowledge

Sources of implicit knowledge

Source of Architecture Knowledge

- **Human sources**
  - Stakeholders, BA, architects, developers

- **Secondary sources**
  - Design patterns, books, case studies
Knowledge Acquisition from Human Sources

Capturing architecture knowledge from human sources

<table>
<thead>
<tr>
<th>Individual Knowledge Acquisition Techniques</th>
<th>Team Knowledge Acquisition Techniques</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interviewing</td>
<td>Brainstorming</td>
</tr>
<tr>
<td>Questionnaires</td>
<td>Architecture reviews</td>
</tr>
<tr>
<td>Observation</td>
<td>Focus group interviews</td>
</tr>
<tr>
<td>Protocol analysis</td>
<td>Delphi technique</td>
</tr>
<tr>
<td>Repertory grid analysis</td>
<td>Group repertory grid analysis</td>
</tr>
<tr>
<td></td>
<td>Group support systems</td>
</tr>
</tbody>
</table>

Individual Knowledge Acquisition Approaches

- Interviewing
- Questionnaires
- Observations
- Prototype analysis
- Repertory grid analysis
Team Knowledge Acquisition Approaches

- Brainstorming
- Nominal group technique (NGT)
- Focus group interviews
- Delphi technique
- Group repertory grid analysis
- Architecture reviews

Disadvantages

- Individual Knowledge Acquisition Approaches
  - Time consuming
  - Subjective opinion or interpretation
  - Need considerable domain knowledge
  - May result in fabricated information because of defensive attitude
  - Inconsistent knowledge because of integration issue
- Team Knowledge Acquisition Approaches
  - Expensive activities with multiple logistical issues
  - Require an expert moderator
  - May cause non-participation and conflicts
A Comparative View of Other Approaches

<table>
<thead>
<tr>
<th>Method category</th>
<th>Description</th>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method of familiar</td>
<td>Analyzing the tasks that the expert usually performs.</td>
<td>The expert feels comfortable</td>
<td>Can be fairly time-consuming</td>
</tr>
<tr>
<td>tasks</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Structured and</td>
<td>The expert is queried with regard to knowledge of facts and procedures.</td>
<td>For a first-and second pass at a data base, it can</td>
<td>Typically very time consuming</td>
</tr>
<tr>
<td>unstructured</td>
<td></td>
<td>generate much information.</td>
<td></td>
</tr>
<tr>
<td>interviews</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Limited information</td>
<td>A familiar task is performed, but the expert is not given</td>
<td>Can be tailored to extract information on selected sub-</td>
<td>Expert feels uncomfortable and is</td>
</tr>
<tr>
<td>tasks</td>
<td>certain information that is typically available.</td>
<td>domains of knowledge</td>
<td>hesitant to make judgments</td>
</tr>
<tr>
<td>Constrained processing</td>
<td>A familiar task is performed, but the expert must do so</td>
<td>Can be tailored to extract information on selected sub-</td>
<td>Expert feels uncomfortable and is</td>
</tr>
<tr>
<td>tasks</td>
<td>under time or other constraints.</td>
<td>domains of knowledge</td>
<td>hesitant to make judgments</td>
</tr>
<tr>
<td>Method of “tough</td>
<td>Analyses of a familiar task that is conducted for a set of data that</td>
<td>Can yield information about refined reasoning</td>
<td>Occur unpredictably; the knowledge acquirer</td>
</tr>
<tr>
<td>cases”</td>
<td>presents a “tough case” for the expert.</td>
<td></td>
<td>may not be present</td>
</tr>
</tbody>
</table>


Pattern-Based Architecture Knowledge

- Exploiting relationships among scenarios, quality attributes, and patterns as architecture knowledge
- Patterns widely codified
  - Links among scenarios, quality attributes and patterns
- We have observed that patterns
  - Include implicit scenarios and links between quality attributes and patterns
  - Provide a previously untapped source of reusable architecturally significant information
- Extracting the information
  - Relatively time-consuming and required expertise
A mechanism for extracting (mining) architecture data from patterns

- General process model
  - Steps involved in mining patterns with guidelines
- Templates to record
  - Generic architecture information
  - Project related information relating to concrete scenarios

Intended to reduce the time and expertise needed to extract architecturally significant information from patterns

Pattern-Mining Process Model
Pattern-Mining Process

- Select software pattern to be explored for architectural information
- Understand the pattern documentation format
- Identify architectural information described in a pattern’s documentation
- Capture each type of information separately
- Organise the extracted information using the provided template
- Validate and refine documented information

Diagrammatic Guidelines for Mining Pattern
Documenting the Extracted Information

<table>
<thead>
<tr>
<th>Pattern Name: Name of the pattern</th>
<th>Pattern Type: Architecture, design, or style</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>A brief description of the pattern.</td>
</tr>
<tr>
<td>Context</td>
<td>The situation for which the pattern is recommended.</td>
</tr>
<tr>
<td>Problem</td>
<td>What types of problem the pattern is supposed to address?</td>
</tr>
<tr>
<td>Suggested</td>
<td>What is the solution suggested by the pattern to address the problem?</td>
</tr>
<tr>
<td>Forces</td>
<td>Factors affecting the problem and solution and pattern’s justification.</td>
</tr>
<tr>
<td>Tactics</td>
<td>What tactics are used by the pattern to implement the solution?</td>
</tr>
<tr>
<td>Affected Attributes</td>
<td></td>
</tr>
<tr>
<td>S1</td>
<td>Positively</td>
</tr>
<tr>
<td></td>
<td>Negatively</td>
</tr>
<tr>
<td></td>
<td>Attributes supported</td>
</tr>
<tr>
<td></td>
<td>Attributes hindered</td>
</tr>
<tr>
<td>General scenarios</td>
<td></td>
</tr>
<tr>
<td>S</td>
<td>A textual, system independent specification of a quality attribute.</td>
</tr>
<tr>
<td>S</td>
<td></td>
</tr>
<tr>
<td>Example</td>
<td>Some known examples of the usage of the pattern to solve the problems.</td>
</tr>
</tbody>
</table>

Using the Template

<table>
<thead>
<tr>
<th>Pattern Name: Business Delegate</th>
<th>Pattern Type: Design pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brief description</td>
<td>This pattern reduces coupling between tiers by providing an entry point for accessing the services another tier. It also supports results caching to improve performance…</td>
</tr>
<tr>
<td>Context</td>
<td>A client may be exposed to the complexity of dealing with the distributed components…</td>
</tr>
<tr>
<td>Problem description</td>
<td>Presentation-tier components interact directly with business services. Such a direct interaction makes the clients vulnerable to any changes in the business services…</td>
</tr>
<tr>
<td>Suggested solution</td>
<td>Reduce coupling between presentation-tier clients and business services. The Business Delegate hides the underlying implementation details of the business service…</td>
</tr>
<tr>
<td>Forces</td>
<td>Presentation-tier clients require access to business service. It is desirable to minimize coupling to hide implementation details from clients.</td>
</tr>
<tr>
<td>Available tactics</td>
<td>Delegate Proxy and Delegate Adapter</td>
</tr>
<tr>
<td>Affected Attributes</td>
<td></td>
</tr>
<tr>
<td>S1</td>
<td>Positively</td>
</tr>
<tr>
<td></td>
<td>Negatively</td>
</tr>
<tr>
<td></td>
<td>Reduce coupling, manageability, performance</td>
</tr>
<tr>
<td></td>
<td>Introduce new layer, increased complexity</td>
</tr>
<tr>
<td>General scenarios</td>
<td></td>
</tr>
<tr>
<td>S1</td>
<td>Presentation-tier components shall not be exposed to the implementation details of the business services they use.</td>
</tr>
<tr>
<td>S2</td>
<td>System shall provide a caching mechanism to improve response to business service request.</td>
</tr>
<tr>
<td>S3</td>
<td>Services calls across network or tiers shall be minimized to avoid degraded performance.</td>
</tr>
<tr>
<td>Examples</td>
<td>E-commerce portals, online content providers, sports websites.</td>
</tr>
</tbody>
</table>
## A Template for Documenting Concrete Knowledge

<table>
<thead>
<tr>
<th>Project Name: Which project needs this scenario?</th>
<th>Date: When was proposed?</th>
<th>Scenarios No: Serial number assigned to the scenario</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project domain: Domain of the project</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Business goals</th>
<th>Which business goals does this scenario achieve?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stakeholders</td>
<td>Which class of the stakeholders did suggest this scenario?</td>
</tr>
<tr>
<td>Attributes</td>
<td>Which quality attributes are required by this scenario?</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Description</th>
<th>A brief description of the scenario.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Concrete scenario</th>
<th>Design rational</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stimulus</td>
<td>Name of the architectural pattern or style that can support this scenario.</td>
</tr>
<tr>
<td>Context</td>
<td></td>
</tr>
<tr>
<td>Response</td>
<td>What are the design tactics used by the pattern/style to support the scenarios?</td>
</tr>
<tr>
<td>Complexity</td>
<td></td>
</tr>
<tr>
<td>Priority</td>
<td>What are reasons for using the patterns/tactics? How does it provide the desired quality attributes?</td>
</tr>
</tbody>
</table>

### Documenting Concrete Knowledge Example

<table>
<thead>
<tr>
<th>Project Name: Qualification Verification System</th>
<th>Date: 12/06/2005</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project domain: E-Commerce application</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Business goals</th>
<th>Customer satisfaction and process efficiency.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stakeholders</td>
<td>Business Manager, System sponsors, and End User.</td>
</tr>
<tr>
<td>Attributes</td>
<td>Improved performance.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Description</th>
<th>The response to a business service request shall be improved to avoid users’ frustration and system shall be able to handle up to 1000 users concurrently without any delay in the response time.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Concrete scenario</th>
<th>Design tactics</th>
<th>Design rational</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stimulus</td>
<td>Delegate proxy and Caching</td>
<td>This pattern exposes an interface to the business service API by using proxy function to pass the client methods to the session bean. It can cache any necessary data and references to the session bean’s home or remote objects to improve performance by reducing the number of lookups.</td>
</tr>
<tr>
<td>Context</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Response</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Complexity</td>
<td>Medium</td>
<td></td>
</tr>
<tr>
<td>Priority</td>
<td>High</td>
<td></td>
</tr>
</tbody>
</table>
Value of Mining Pattern

- **During architecture design**
  - Helps identify patterns used in a software architecture and understand the reasons for using those patterns.
  - Helps identify appropriate patterns for system design.
- **During architecture analysis**
  - Provide confidence in an architecture’s capability of supporting certain scenarios
  - Help develop concrete scenarios based on general scenarios extracted from the patterns in the architecture

Empirical Evidence

- **Research Program**
  - Observational study and controlled experiments
- **Pattern-Mining Framework**
  - Framework is effective and it takes 30-45 minutes to mine one pattern
- **Understanding and designing architecture**
  - ASIP compared with standard pattern documentation is more helpful in understanding and designing architectures
- **Architecture evaluation**
  - Improves scenario gathering activity by helping stakeholders to develop better quality scenarios
Trial of the Technology

- Tailoring BRedB for DSTO
  - Define domain-specific evaluation criteria
  - Capture knowledge underpinning COTS’ architectures
  - Support standards compliance analysis
  - Document rationale for COTS acquisition decisions
  - Rank and compare architectural solutions
  - Track architecture decision makers and evaluators
  - Generate reports for management decision making

Summary of Section 2

- An Introduction to BRedB – An Architecture Knowledge Management Infrastructure
  - Architecture
  - Features to support architecture design, documentation, and evaluation activities
- Approaches to capturing architectural knowledge
- Using BRedB on the case study
Domain Modeling for Architecture Knowledge

- Knowledge modeling
- Knowledge modeling process
- Knowledge modeling techniques
- Architecture knowledge constructs & relationships
- Models characterizing architecture knowledge
- Some example queries for case study

Knowledge Modeling

- Why do we need models?
- Importance of knowledge modeling
- Role of models in knowledge acquisition and validation
- Modeling in software engineering and knowledge engineering: similarities and differences
- How to model knowledge?
A Generic Process of Modeling Knowledge

- Domain analysis for identifying constructs and their attributes
- Identify relationships among different constructs
- Determine properties of relationships
- Apply some formal modeling approach, e.g. Entity-Relation models
- Refine the conceptual model using the principles of a formal modeling approach
  - Avoid fully normalizing the model
- Assessment - many ways such as expert opinion or theoretical

Knowledge Modeling Techniques

- CommonKADS
- Protégé
- UML & OCL
- Multi-perspective modeling
Comparison of Knowledge Modeling Techniques

<table>
<thead>
<tr>
<th>Technique</th>
<th>CommonKADS</th>
<th>Protege 2000</th>
<th>Multi-Perspective</th>
<th>UML</th>
</tr>
</thead>
<tbody>
<tr>
<td>K.E. methodology</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>O-O approach</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Platform independent</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Hybrid approach</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Editor tool</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Standard modeling language</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Documentation</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Evolving</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Domain</td>
<td>Medical, Business, &amp; Legal</td>
<td>Medical, Business, &amp; Legal</td>
<td>Medical, Business, &amp; Legal</td>
<td>Medical, Business, &amp; Legal</td>
</tr>
<tr>
<td>Other features (OKSC, RDF, Semantic Web)</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>

Source: M.S. Abdullah et al. 3rd European Conf. on KM, 2002

Exercise – Domain modelling

- Form a group of 2-3 and identify constructs, their properties, and relationships that characterise architecture knowledge in your domain/industry.
Architectural Knowledge – Some Constructs

- Architecture decision
- Design option
- Design rationale
- Pattern – (Architectural, design, platform)
- Tactic
- Architectural significant requirement (ASRs)
- Scenarios
  - Abstract
  - General

Some of the Sources for Identifying Constructs

- SARA report
- IEEE 1471 standards
- Views and Beyond
- Books on software architectures
- Literature on software architecture
A Few Heuristics for Modeling Knowledge

- Identify most-frequent, synonyms or related terms
- Identify relationships between terms and models
- Identify basic constraints
  - Pre-requisite constraint
  - Temporal constraint
  - Mutually inclusive constraint
  - Mutually exclusive constraint
- Identify domain constraints
- Identify domain dependencies
Summary of Section 3

- Knowledge modeling
- Knowledge modeling techniques
- Architecture knowledge modeling process
- Architecture knowledge constructs & relationships
- Models characterizing architecture knowledge
- Some example queries for case study

Using knowledge during design & evaluation

- Industrial case study – Avionics Architectures
  - Problem domain description
  - Current evaluation methods
- A framework for avionic architecture evaluation using knowledge management
  - Quality attributes
  - Evaluation framework
  - Risk management
- Demonstration: Using BRedB for avionics architecture evaluation
Airborne Mission Systems (AMS) Branch

• Research, analytical studies and experimental work in Avionics and AMS.
• Development of innovative methods and tools to aid the understanding of advanced avionics technologies
• Provide short-term advice to Australian Defence Force (ADF) project teams and develop long-term strategies to support the ADF on mission systems acquisitions.

Airborne Mission Systems (AMS) Branch

• Responsible for performing Technical Risk Assessments (TRA) of mission systems
  – critical for the ADF acquisition process
• AMS is required to understand and organize large amounts of architecture design knowledge
• “Software intensive projects are historically considered the most risk prone in the Defence domain”
• Building capabilities in systematically evaluating architectures and maintaining architecture knowledge
ADF Acquisitions and Software related Issues

- Navy's $100m chopper can't fly in bad light
  - $1.1b were spent before finding that Seasprite had major software problems
- Billions wasted on 'Collin Class Submarines'
  - Software and architectural issues made them 'Duds'
- Airborne Early Warning and Control project has 4 million lines of code
- Software literacy is vital but quite low
- Poor record keeping makes it impossible to trace reasons for bungles in acquisition projects

Trial's Background

- Collaborative R&D project between NICTA and DSTO has been undertaken during 2006
- Exploit NICTA architecture evaluation methods and tools for improving AMS capabilities
- Codifying architecture evaluation process and design knowledge and rationale is a vital goal of this project
- 8 members team (4 on the each side)
Trial’s Objectives

- **Support a systematic and repeatable evaluation process**
- **BRedB** is expected to help AMS to achieve several goals:
  - Capture rationale for architecture decisions
  - Help build architectural capabilities
  - Improve architectural reusability
  - Provide an audit trail for TRA findings
  - Reduce demands on subject matter experts
  - Encourage best architectural practices
  - Improve efficiency of architectural processes
  - Accelerate the training process of new employees

BRedB Supported Architecture Evaluation Process
Tailoring BRedB

1/3

• AMS doesn’t design - it evaluates architectures proposed by contractors in response to a Request for Proposal (RFP)
• AMS only needs BredB features that support architecture evaluation activities
• Gathering customization requirements by analysing activities/tasks of the current process

Tailoring BRedB

2/2

• High priority requirements implemented:
  – Classification of project data according to the Defence classification scheme.
  – Mechanism for recording compliance of architectural decisions with respect to requirements.
  – Store and evaluate several contractors’ proposals for the same set of scenarios within one project.
  – Different levels of access to project data based on the Defence security scheme.
  – Ability to import/export data from the tool based on the classification scheme.
  – Integration with requirements management and architecture modelling tools.
AMS Case Study

- Assessing customized BRedB by supporting architecture evaluation process of AMS
- Evaluating the proposed architecture for an airborne mission system
- Goal was to investigate BRedB’s role in the architecture evaluation process and how it could help capture and manage architecture knowledge and rationale

Case Study – Use of the Knowledge Base

- BRedB was populated with AMS domain knowledge, starting with a preliminary domain specific quality model.
- Quality model was based on ISO 9126 and AMS domain knowledge elicited from domain experts
- Quality model is used to assess the potential risks of architectural designs
- Quality model consists of six quality attributes:
  - Performance
  - Reliability
  - Usability
  - Maintainability
  - Portability
  - Functionality
### Quality Framework - Performance

<table>
<thead>
<tr>
<th>Metric</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Latency</td>
<td>What are average/worst case latencies for behavior such as: Radar to display, Algorithm used to calculate tracks/bearings, and Video frame updates.</td>
</tr>
<tr>
<td>Accuracy</td>
<td>Accuracy relates to timestamps. Check the architecture for: place of time reference, multiple clocks are maintained synchronized, clock signal distribution through network?</td>
</tr>
<tr>
<td>Capacity</td>
<td>Look for the following data related to capacity: Number of tasks a system can support/processor and database size/storage, and processor capacity.</td>
</tr>
<tr>
<td>Throughput</td>
<td>Check if the proposed processor and bus architecture can support the likely throughput in terms of events per second (both average and worst case)</td>
</tr>
<tr>
<td>Resource usage</td>
<td>Look to assess several things in terms of resource usage: CPU (peak CPU load, spare capacity), Power consumption, Cooling, Memory, Disk (usage for each disk), and I-O rates.</td>
</tr>
<tr>
<td>Scalability</td>
<td>Various questions to be asked such as: Are any key software elements single-threaded? Do the specifications indicate spare CPU capacity that could be used for additional processing?</td>
</tr>
<tr>
<td>Schedulability</td>
<td>Is it likely that high priority tasks can be scheduled when needed? What is the clock granularity?</td>
</tr>
</tbody>
</table>

---

### Scenarios – Performance

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initialization</td>
<td>Must perform all initialization activities within 10 minutes.</td>
</tr>
<tr>
<td>Latency</td>
<td>Run simulations with no instantaneous lags greater than five seconds, no average lags greater than three seconds.</td>
</tr>
<tr>
<td>Capacity</td>
<td>Run-time simulation with debug enabled.</td>
</tr>
<tr>
<td>Latency</td>
<td>Finish data collection within 30 seconds of simulation termination.</td>
</tr>
<tr>
<td>Throughput</td>
<td>Finish data collection request from three network sensors within x seconds every one minute.</td>
</tr>
</tbody>
</table>
### Quality Framework – Reliability

<table>
<thead>
<tr>
<th>Quality Aspect</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fault tolerance</td>
<td>Look for fault tolerance in terms of software and hardware redundancy, separation of flight mission functions, and potential single points of failure, mechanisms of notify operators of faults.</td>
</tr>
<tr>
<td>Recoverability</td>
<td>Check for time required for cold/warm start ups, for cold start to full performance, amount of data loss on a failure/restart, any loss of situational information and how to recover/readjust to that.</td>
</tr>
<tr>
<td>Maturity</td>
<td>Are existing tested components used? Level of process maturity of the organization.</td>
</tr>
<tr>
<td>Survivability</td>
<td>Check for the kind of redundancy for key software/hardware components, communication channels.</td>
</tr>
<tr>
<td>Availability</td>
<td>Check for restart/recovery times after failure and supportability.</td>
</tr>
</tbody>
</table>

### Scenarios – Reliability

<table>
<thead>
<tr>
<th>Quality Aspect</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Availability</td>
<td>Simulation controller initiates simulation execution (a game), starts subordinate processes, loads parameter files, and simulation runs from T=inf to T=0 within 10 minutes.</td>
</tr>
<tr>
<td>Availability</td>
<td>Runs scenarios without crashes: no hardware and/or software failures. Terminates normally.</td>
</tr>
<tr>
<td>Fault tolerance</td>
<td>Ability to change the location and orientation of assets and still meet the performance, reliability, and credibility criteria.</td>
</tr>
<tr>
<td>Maturity</td>
<td>Organization has been developing similar systems for the last 20 years and has been certified as CMMI Level 4.</td>
</tr>
<tr>
<td>Recoverability</td>
<td>System encounters serious data corruption and needs to restart. System should recover in 30 seconds.</td>
</tr>
</tbody>
</table>
Quality Framework in BRedB

Utility Tree of the Quality Framework
Case Study – Use of the Project Base

- Concrete scenarios relating to the quality factors were added to the Project Base.
- The architecture design was captured and compared with alternative designs with respect to the scenarios.

Case Study – Reporting

- A report of the evaluation shows the findings of an architectural decision that is aimed at satisfying a concrete scenario.
Challenges

• Domain understanding
• Access to the required information/knowledge
• Security modelling and implementation
• Modification of the underlying data model
• Communication/coordination issues

Initial Findings

• The use of an evaluation framework and knowledge management tool brings added rigour to the evaluation process.
• It is anticipated that access to past project experience stored in BRedB will be valuable in the evaluation process.
• The modified version of BRedB provides AMS with an effective and efficient mechanism to organise and understand large amounts of architecture knowledge
Tool Demo

- Short Video clips on using BRedB and Wiki based tool

Summary of Section 4

- Industrial case study – Avionics Architecture
- A framework for avionic architecture evaluation using knowledge management
  - Quality attributes
  - Evaluation framework
  - Risk management
- Demonstration: Using BRedB for avionics architecture evaluation
Some useful references

- Bass et al. – Software Architecture in Practice
- Clements et al. - Documenting Software Architectures : Views and Beyond
- Ian Gorton – Essential Software Architecture
- IEEE Recommended Practice for Architecture Description of Software-Intensive System (IEEE Std 1471-2000)
- Philippe Kruchten - http://philippe.kruchten.com/
- Grady Booch’s site

Acknowledgement

- Tutorial incorporates material provided by Andrew Northway, Paul Heuer, and Thong Nguyen of DSTO. Discussions with our collaborators Antony Tang and Jun Han resulted in many ideas presented in the tutorial.
- Shirley Xu worked on the engineering side of the BRedB. Before that several students of the UNSW helped us build the first version of that tool.
- Most of the reported work was produced while the presenters were working at NICTA
Collaborative Opportunities

- We welcome collaborative opportunities to deploy our tools and techniques to help design better architectures and effectively and efficiently evaluate and manage them.

Contacts:
- Ali Babar, Lero, Ireland (malibaba@lero.ie)
- Ian Gorton, PNNL, USA (ian.gorton@pnl.gov)

Thank you!