Repositories

To Be Departmentalized Or Not To Be

Simple Repository Database Architecture
Virtual Repository
Blackboard Systems
Case Study: Mobile Robot Architecture

Two distinct kinds of components
✦ a central data store (repository), representing the current state
✦ a collection of independent components, operating on the central data store

Not To Be
✦ a collection of independent components, each with its own data store, communicate with each other.

2 Main Categories of Repositories
according to the type of interactions between the repository and its external components:
✦ (traditional) repository database:
  input transactions activate processes to execute
✦ blackboard
  its current state is the main trigger for selecting processes to execute
Batch Sequential Pipeline Systems

Transactions

- Input
- Accept input
- Parse input into transactions

- Online
- Check validity (e.g., name field: alpha, amount: xxxxxx, xxx)
- Filter out invalid transactions

- Edit
- Filter out invalid transactions

- Sorted
- Sort valid transactions

- Update
- Process each transaction against master files
- Filter out invalid transactions

- Old master
- New master

- Sorted transactions
- Produces periodic reports

**Constraints:**
- Processes run in a fixed sequence, but they do not know each other
- Each runs to completion, producing an output, before the next process begins

Simple Repository Database Architecture

**Two trends away from batch sequential processing**

- Interactive technology for on-line incremental updates and queries
- Growth in the set of transactions and queries

**Architecture**

- Each transaction (in each component) does an update or a query
- Db stores persistent data shared among different transactions
- No fixed ordering among transactions (cf., batch sequential)
- Concurrency control handled by "control"

```
Mfg
#robot = 1  [#robot ?>0] -> sell  [#robot ?>0] -> sell
```
Software Repository

**Purpose**
- to allow the user to define, store, access and manage all the information about any software
- tools access data thru open representation standard, CDIF (CASE data interchange format)

**Architecture** commonly based on the ANSI SPARC 3-level schema:
- external (logical) schema: individual users’ view
- conceptual model: comprehensive view of entire contents
- physical model: physical implementation for data storage

Virtual Repository

**Multiple databases**
- distributed, heterogeneous but (often) transparent
- due to corporate reorganizations, mergers, consolidations, etc.

**Heterogeneity**
- different schemas, names (tables, attributes), data representations

**UTD-Library**
- Items
  - Item# | title | author name
  - numeric 32
- LC-Number
  - item# | c-letter | f-digit
- Publisher
  - item# | name | address

**UNC-Library**
- Items
  - i# | title | author lat name
  - alpha 10
- Item-Subject
  - i# | subject
- Publisher
  - i# | p-name | str-num
Virtual Repository

✈ Federated approach
✦ combine multiple distributed schemas
✦ reconcile representational differences
✦ communicate results across distributed systems

✈ Virtual integrated schema
✦ if imported schemas are consistent, simple merging
  otherwise, create a superset schema including every definition in each
  imported schema (metadata)
✦ to users, the integrated schema acts as the virtual database
  (i.e., local schemas are transparent)

Integrated schema

Native schema Import schema 1 Import schema 2 Import schema n

Local schema 1 Local schema 2 Local schema n

Local Access 1 Local file sys 1 Local Sybase 2 Local Oracle n

Blackboard Systems

✈ Basic concepts
✦ the abstract model for access is "direct visibility"
✦ many human experts watch each other solve a problem at a real blackboard

✈ 3 main parts
✦ the knowledge sources:
  separate, independent parcels of application-dependent knowledge;
  Interaction takes place solely thru the blackboard
✦ the blackboard:
  problem-solving state data, organized into an application-dependent hierarchy.
  Knowledge sources make changes to the blackboard that incrementally
  lead to a solution to the problem
✦ control:
  driven entirely by state of blackboard.
  Knowledge sources respond opportunistically
  when changes in the blackboard make them applicable.
Traditional Applications

✈ AI systems
✦ signal processing (speech and pattern recognition)
✦ shared data to data with loosely coupled agents

✈ Wreck a nice beach
✦ signal segmentation for speech understanding
✦ phoneme recognition
✦ word candidate generation
✦ syntactic-semantic connection

Case Study: Mobile Robot Architecture

["An architecture for Sensor Fusion in a Mobile Robot," Shafer, Stentz & Thorpe,
Proc. IEEE Int. Conf. on Robotics and Automation, 86]

✈ Context
✦ a mobile robotic system controls a manned or partially-manned vehicle
  (e.g., a car, submarine, space vehicle; but not R2D2, C3PO)
✦ useful for "driving impaired", underwater exploration, space exploration,
hazardous waste disposal, etc.
✦ external sensors, actuators and software system:
  ✦ external sensors (e.g., rangefinders, TV cameras) work in parallel for detecting
    stop signs, traffic lights, intersections, etc.
  ✦ multiple sensors have different times -> requires asynchronous sensor fusion
    (i.e., integration of multiple parallel sensors in a single system)
  ✦ actuators at real-time rates as well (e.g., apply pressure to the break system;
    activate alarm sound; turn the steering wheel; apply pressure to the accelerator)
  ✦ software functions include:
    ✦ acquiring sensor inputs
    ✦ controlling the motion of the steering wheel and other (moveable) parts (e.g.,
      break, windshield wiper, defogger, temperature control)
    ✦ planning its future path
Case Study: Mobile Robot Architecture

✦ complicating factors:
  ✦ obstacles blocking the robot’s path:
    ✦ pedestrians, rocks, birds, animals on the highway
    ✦ road under construction, closed road, detour, merging lanes
    ✦ accident on the road, malfunctioning traffic lights
  ✦ imperfect sensors:
    ✦ slow TV cameras, distance-limited rangefinders
    ✦ vision impaired by rain, animal debris
    ➾ can miss speed limits, school district signs, etc.
  ✦ mechanical limitations:
    restrict accuracy of movement
    max 45 degree angle turn, 10 second delay before a full stop
  ✦ power shortage

Solution 1: Control Loop

Controller

\[ \text{adjust future plans} \]
\[ \text{rear car = police car?} \]
\[ \text{speed up} \]
\[ \text{lane change} \]

initiate robot action

- increase microphone volume
- increase intake
- turn steering wheel to the right
- start windshield wiper

monitor consequences of actions

- moving straight, speeding up
- rear car approaching, sidewalk too close
- visibility good

Active components of robot

+ actuators
+ sensors

Environment

- continuous changes: rain falling, deer crossing, front car getting closer

Lawrence Chung

+ simplicity
+ fault tolerance and safety supported by the closed loop
- no decomposition of cooperating components (sensing, planning, acting)
- inability to handle complex coordination
**Solution 2: Layered Architecture**

- **Supervisor** UI and overall supervisory functions
- **Global Planning** planning & replanning robot's actions dealing w. problems
- **Control** schedule actions
- **Navigation** manage robot's navigation
- **Real-world Modelling** maintain robot's model of the world
- **Sensor Integration** combined analysis of different sensor inputs
- **Robot Control** ✦ actuators: motors, joints, ...; ✦ sensors: TV cameras, rangefinders, microphones

+ decomposition of cooperating components (control vs. navigation vs. sensor integration)
+ world model can disambiguate conflicting sensor data (sunshine & no cloud -> no rain)
+ fault tolerance and safety

- no direct interaction between sensors/actuators and global planning (fire -> spray fire hydrant) robots usually do not follow this kind of order scheme
- merge into abstraction hierarchies:
  - data hierarchy: raw sensor input
  - control hierarchy: motor control
  - navigation
  - scheduling
  - planning

> **Solution 3: Blackboard Architecture**

**Perception subsystem**
- raw input from multiple sensors
- integrate into a coherent interpretation
- color, reflectance, edge -> shape (stop sign, traffic light, curb)

**Blackboard**
- local map, sensor data, current position, geographic features
- spatial, geometric reasoner

**Mission**
- current position: DFW Airport
- destination: UTD
- constraint via I-635
- action at destination: wake-up call, spray perfum
- trigger conditions:
  - travel 1 mile -> slow down & pay the tollgate
  - a sign for I-635 -> slow down and turn
  - a traffic jam + early exit

**Captain**
- receive trigger conditions
- monitor the env. for landmarks
- trigger condition OR landmark
- notify the blackboard

**Lookout**
- high-level path plan

**Map Navigator**
- low-level path plan

**Pilot**
- motor control

Lawrence Chung
Solution 3: Blackboard Architecture

- Captain
- Lookout
- Navigator
- Pilot

- Mission instruction
- Receive trigger conditions
- Monitor the environment for landmarks
- Trigger condition or landmark -> notify the blackboard
- High-level path plan
- Low-level path plan

Blackboard
- Local map, sensor data, current position, geographic features
- Spatial, geometric reasoner

Perception Subsystem
- Camera control
- Rangefinder control
- Sonar control
- Raw input from multiple sensors
- Color, reflectance, edge -> shape (stop sign, traffic light, curb)

- Use of implicit invocation based on the contents of the db
- The architecture is capable of modelling the cooperation of tasks and coordination
  - E.g., lookout may watch for certain geographic features (stop sign)
  - The db informs it when perception subsystem stores images matching the description
- Uncertainty resolution in a flexible manner
  - E.g., landmark detection by lookout and map provides a reality check for the distance estimation

Summary Comparison

<table>
<thead>
<tr>
<th>Control Loop</th>
<th>Layers</th>
<th>Blackboard</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task coordination</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>Dealing with uncertainty</td>
<td>-</td>
<td>+-</td>
</tr>
<tr>
<td>Fault tolerance</td>
<td>+-</td>
<td>-</td>
</tr>
<tr>
<td>Safety</td>
<td>+-</td>
<td>-</td>
</tr>
<tr>
<td>Performance</td>
<td>+-</td>
<td>-</td>
</tr>
<tr>
<td>Flexibility</td>
<td>+-</td>
<td>-</td>
</tr>
</tbody>
</table>

❖ Prioritize
❖ Other NFRs
❖ More Scenario Analysis