Computer Science Program, The University of Texas, Dallas

Repositories

To Be Departmentalized Or Not To Be

Simple Repository Database Architecture

Virtual Repository

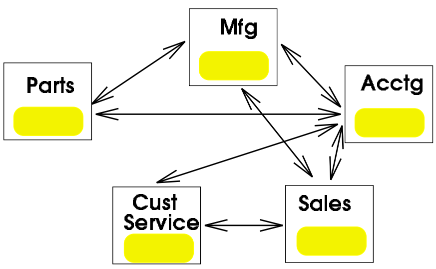
Blackboard Systems

Case Study: Mobile Robot Architecture

Lawrence Chung

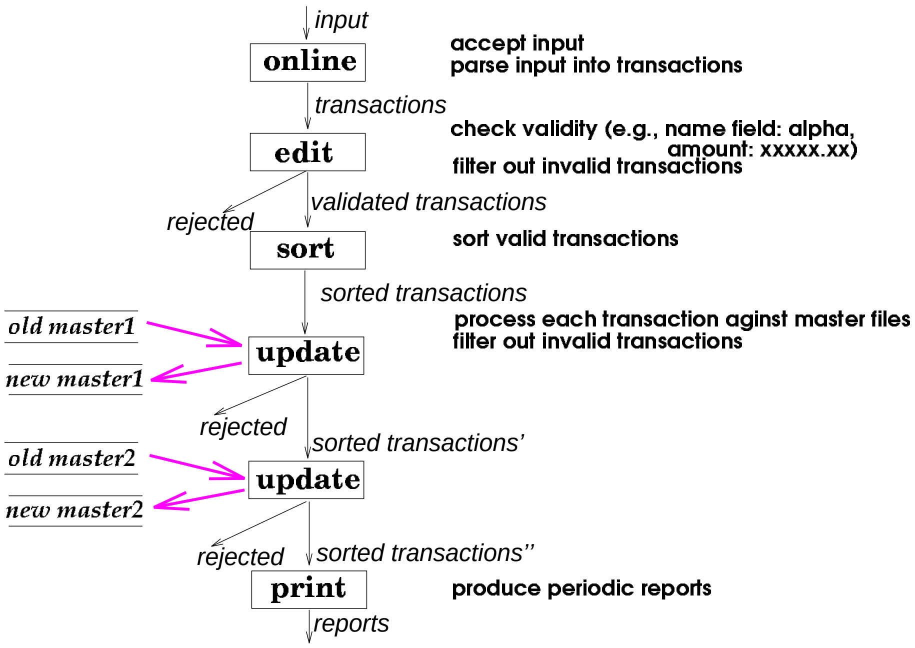
To Be Departmentalized Or Not To Be

- **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



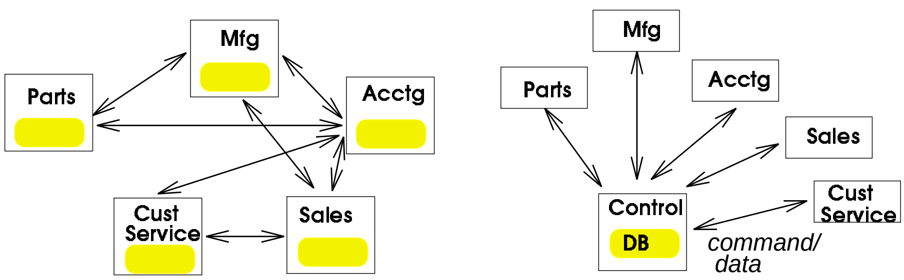
Constraints:

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**

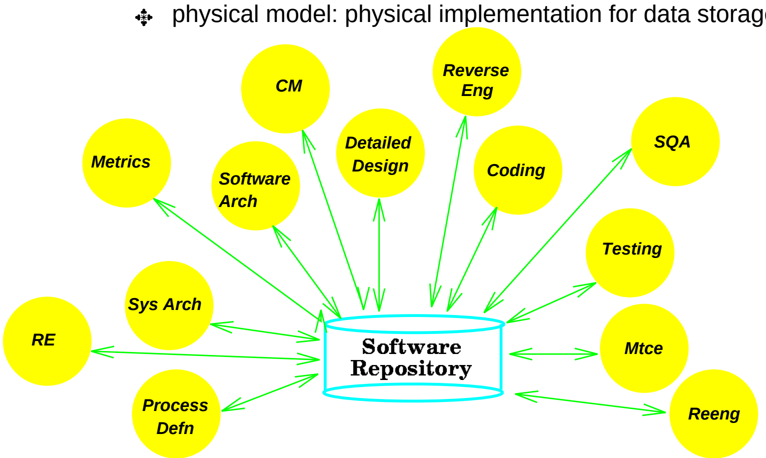


- each transaction (in each component) does an update or a query
- db stores persistent data shared among different transactions
- no fixed ordering among transactions (cf., batch sequential)
- concurrency control handled by "control"

Mfg	Sales	
#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 ***What's 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

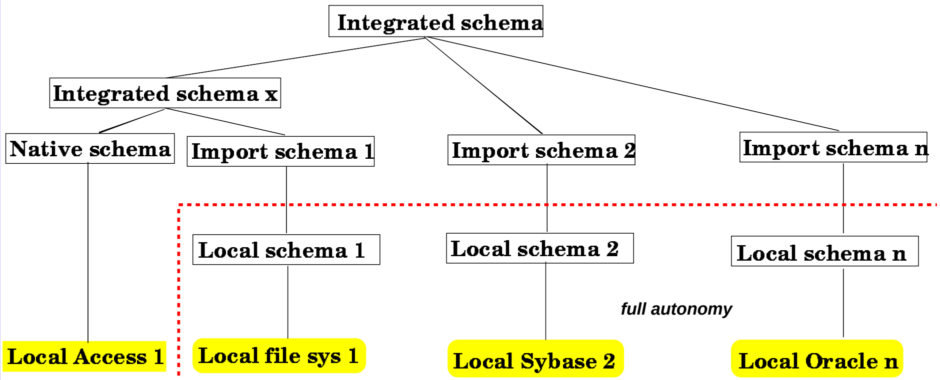
Federated approach

- 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)

numeric 32



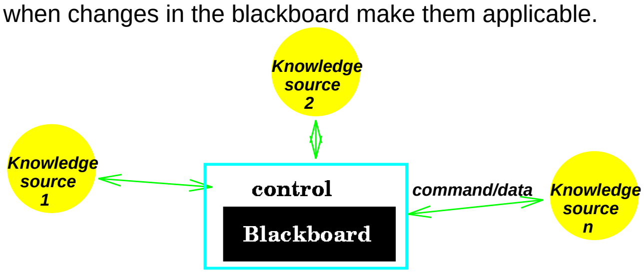
Blackboard Systems

Basic concepts

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

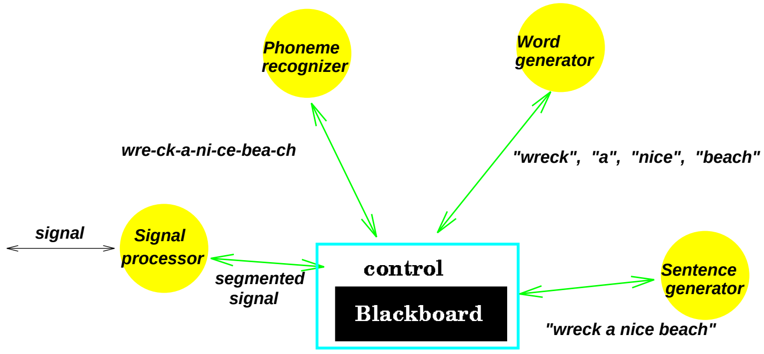
3 main parts

- *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 segmemtation 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

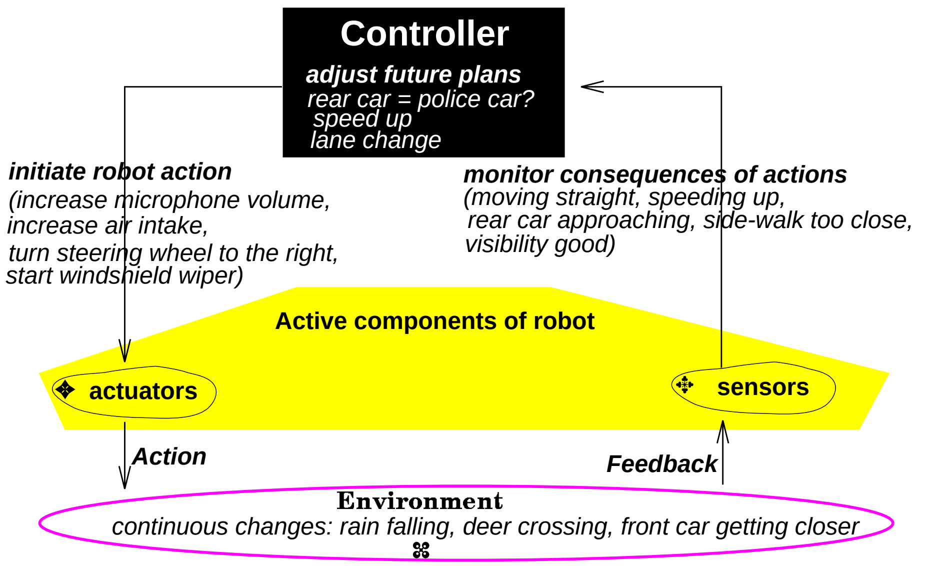
- complicating factors:
  - *obstacles blocking the robot's path:*
    - pedestrians, rocks, birds, animals on the highway
    - road under construction, closed road, detour, merging lanes
    - accident on the road, malfunctioning traffic lights
  - *imperfect sensors:*
    - slow TV cameras, distance-limited 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



+ 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*
Sensor Integration	*interpretation of data from one sensor*
Robot Control	*actuators: motors, joints, ...;* *sensors: TV cameras, rangefinders, microphones*

Action *Feedback*

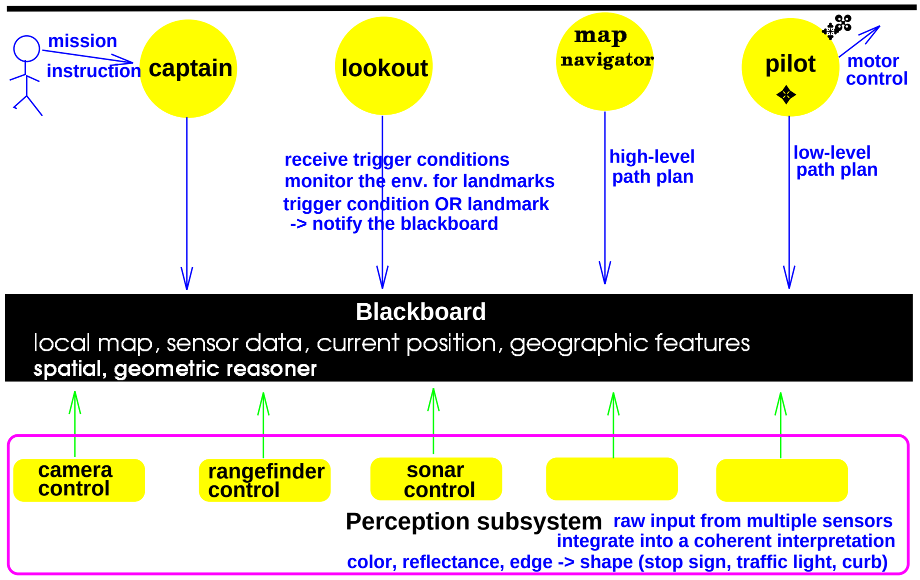
Environment
continuous changes: rain falling, deer crossing, front car getting closer

+ 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 dehydrant) robots usually do not follow this kind of orderly scheme
- mergin two abstraction hierarchies:
  data hierarchy raw sensor input (1)
  interpreted & integrated results (2 & 3)
  the world model (4)
  control hierarchy motor control (1)
  navigation (5)
  scheduling (6)
  planning (7)
  user-level control (8)

=> complex relationships between layers can be hard to decipher
=> low modifiability

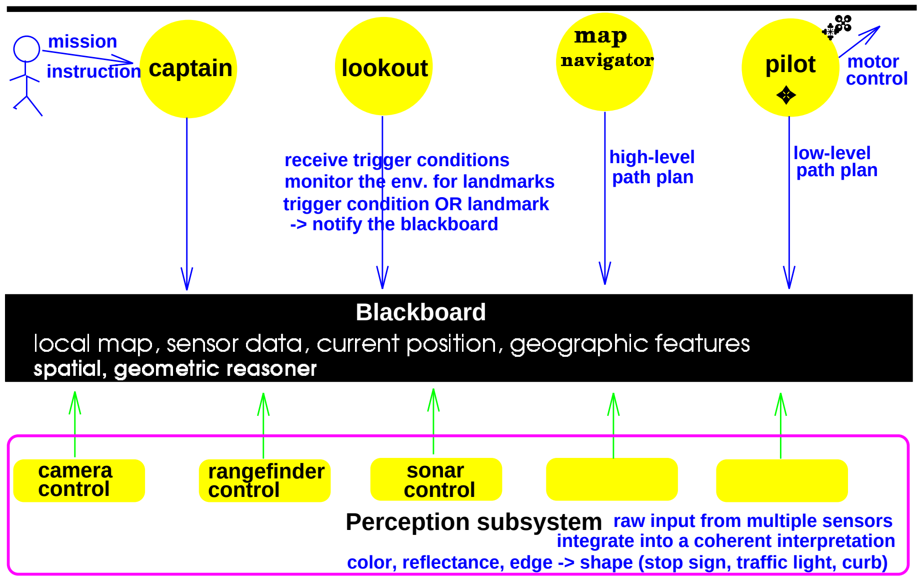
Solution 3: Blackboard Architecture



mission

current position: DFW Airport
destination: UTD
constraint: via Hw635
action at destination: wake-up call, spray perfume
trigger conditions:
- **travel 1 mile -> slow down & pay a the tollgate**
- **a sign for Hw635 -> slow down and turn**
- **a traffic jam -> early exit**

Solution 3: Blackboard Architecture



- 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

	Control Loop	*Layers*	*Blackboard*
Task coordination	+-	-	+
Dealing with uncertainty	-	+-	+
Fault tolerance	+-	+-	+
Safety	+-	+-	+
Performance	+-	+-	+
Flexibility	+-	-	+

- *Prioritize*
- *Other NFRs*
- *More Scenario Analysis*