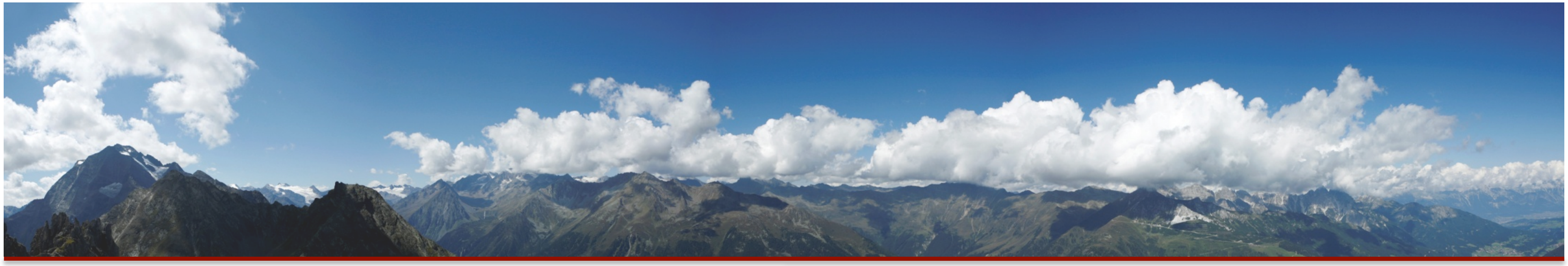
Intelligent Systems

CommonKADS



Where are we?

#	Title
1	Introduction
2	Propositional Logic
3	Predicate Logic
4	Reasoning
5	Search Methods
6	**CommonKADS**
7	Problem-Solving Methods
8	Planning
9	Software Agents
10	Rule Learning
11	Inductive Logic Programming
12	Formal Concept Analysis
13	Neural Networks
14	Semantic Web and Services

Agenda

STI · INNSBRUCK

1. Motivation
2. Technical solution, illustrations and extensions
    1. Overview of CommonKADS
    2. Knowledge model components
    3. Template knowledge models
    4. Knowledge model construction
    5. Knowledge elicitation techniques
3. Example
4. Summary
5. References

All slides are based on the book: Guus Schreiber, Hans Akkermans, Anjo Anjewierden, Robert de Hoog, Nigel Shadbolt, Walter Van de Velde and Bob Wielinga. *Knowledge Engineering and Management: The CommonKADS Methodology*, MIT Press, ISBN 0262193000. 2000.

And slides are partly based on the CommonKads Course http://www.commonkads.uva.nl

MOTIVATION

Knowledge engineering

- Process of
  - eliciting,
  - structuring,
  - formalizing,
  - operationalizing
- information and knowledge involved in a knowledge-intensive problem domain,
- in order to construct a program that can perform a difficult task adequately

Knowledge engineering problems

STI · INNSBRUCK

- Complex information and knowledge is difficult to observe
- Experts and other sources differ
- Multiple representations:
  - textbooks
  - graphical representations
  - skills

Importance of proper knowledge engineering

- Knowledge is valuable and often outlives a particular implementation
  - knowledge management
- Errors in a knowledge-base can cause serious problems
- Heavy demands on extendibility and maintenance
  - changes over time

TECHNICAL SOLUTION AND ILLUSTRATIONS

Overview of CommonKADS

CommonKADS principles

STI · INNSBRUCK

- CommonKADS: a comprehensive methodology for KBS development
- Knowledge engineering is ***not*** some kind of `mining from the expert's head', but consists of constructing different aspect models of human knowledge
- The knowledge-level principle: in knowledge modeling, first concentrate on the conceptual structure of knowledge, and leave the programming details for later
- Knowledge has a stable internal structure that is analyzable by distinguishing specific knowledge types and roles.

CommonKADS Terminology

STI · INNSBRUCK

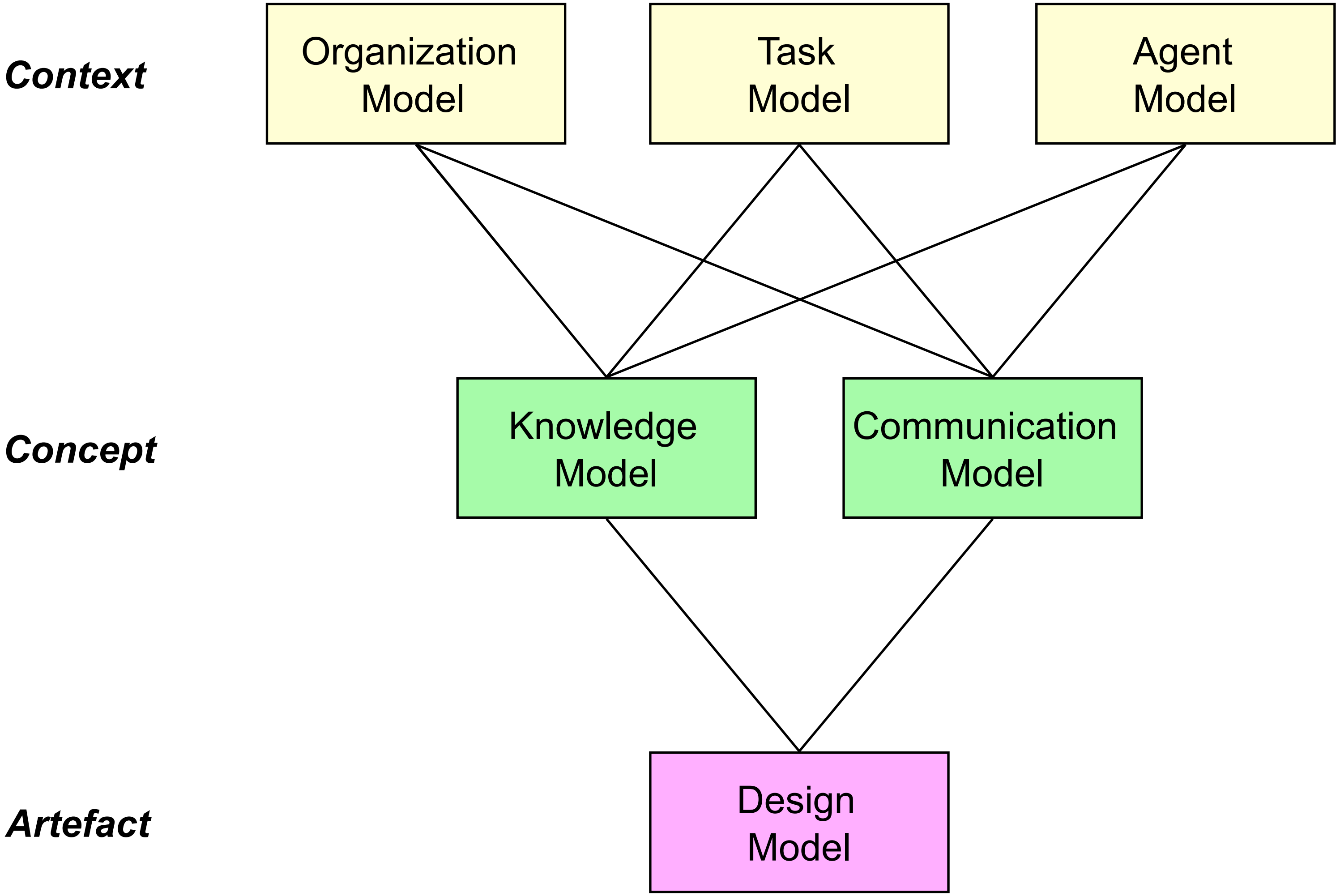
- Domain
  - some area of interest
    banking, food industry, photocopiers, car manufacturing
- Task
  - something that needs to be done by an agent
    monitor a process; create a plan; analyze deviant behavior
- Agent
  - the executor of a task in a domain
    typically either a human or some software system
- Application
  - The context provided by the combination of a task and a domain in which this task is carried out by agents
- Application domain
  - The particular area of interest involved in an application
- Application task
  - The (top-level) task that needs to be performed in a certain application

CommonKADS Terminology

STI · INNSBRUCK

- knowledge system (KS)
  - system that solves a real-life problem using knowledge about the application domain and the application task
- expert system
  - knowledge system that solves a problem which requires a considerable amount of expertise, when solved by humans.

CommonKADS Model Set



Model Set Overview (1)

- Organization model
  - supports analysis of an organization,
  - Goal: discover problems, opportunities and possible impacts of KBS (knowledge-based system) development.
- Task model
  - describes tasks that are performed or will be performed in the organizational environment
- Agent model
  - describes capabilities, norms, preferences and permissions of agents (agent = executor of task).

Model Set Overview (2)

STI · INNSBRUCK

- Knowledge model
  - gives an implementation-independent description of knowledge involved in a task.
- Communication model
  - models the communicative transactions between agents.
- Design model
  - describes the structure of the system that needs to be constructed.

Models exist in various forms

STI · INNSBRUCK

- Model template
  - predefined, fixed structure, can be configured
- Model instance
  - objects manipulated during a project.
- Model versions
  - versions of a model instance can exist.
- Multiple model instances
  - separate instances can be developed
  - example: "current" and "future" organization

The Product

- Instantiated models
  - represent the important aspects of the environment and the delivered knowledge based system.
- Additional documentation
  - information not represented in the filled model templates (e.g. project management information)
- Software

Roles in knowledge-system development

- knowledge provider
- knowledge engineer/analyst
- knowledge system developer
- knowledge user
- project manager
- knowledge manager

many-to-many relations between roles and people

Knowledge provider/specialist

- “traditional” expert
- person with extensive experience in an application domain
- can provide also plan for domain familiarization
  - “where would you advise a beginner to start?”
- inter-provider differences are common
- need to assure cooperation

Knowledge engineer

STI · INNSBRUCK

- specific kind of system analyst
- should avoid becoming an "expert"
- plays a liaison function between application domain and system

Knowledge-system developer

STI · INNSBRUCK

- person that implements a knowledge system on a particular target platform
- needs to have general design/implementation expertise
- needs to understand knowledge analysis
  - but only on the “use”-level

Knowledge user

- interact with the prospective system or are affected indirectly by the system
- Level of skill/knowledge is important factor
- May need extensive interacting facilities
  - explanation
- His/her work is often affected by the system
  - consider attitude / active role

Project manager

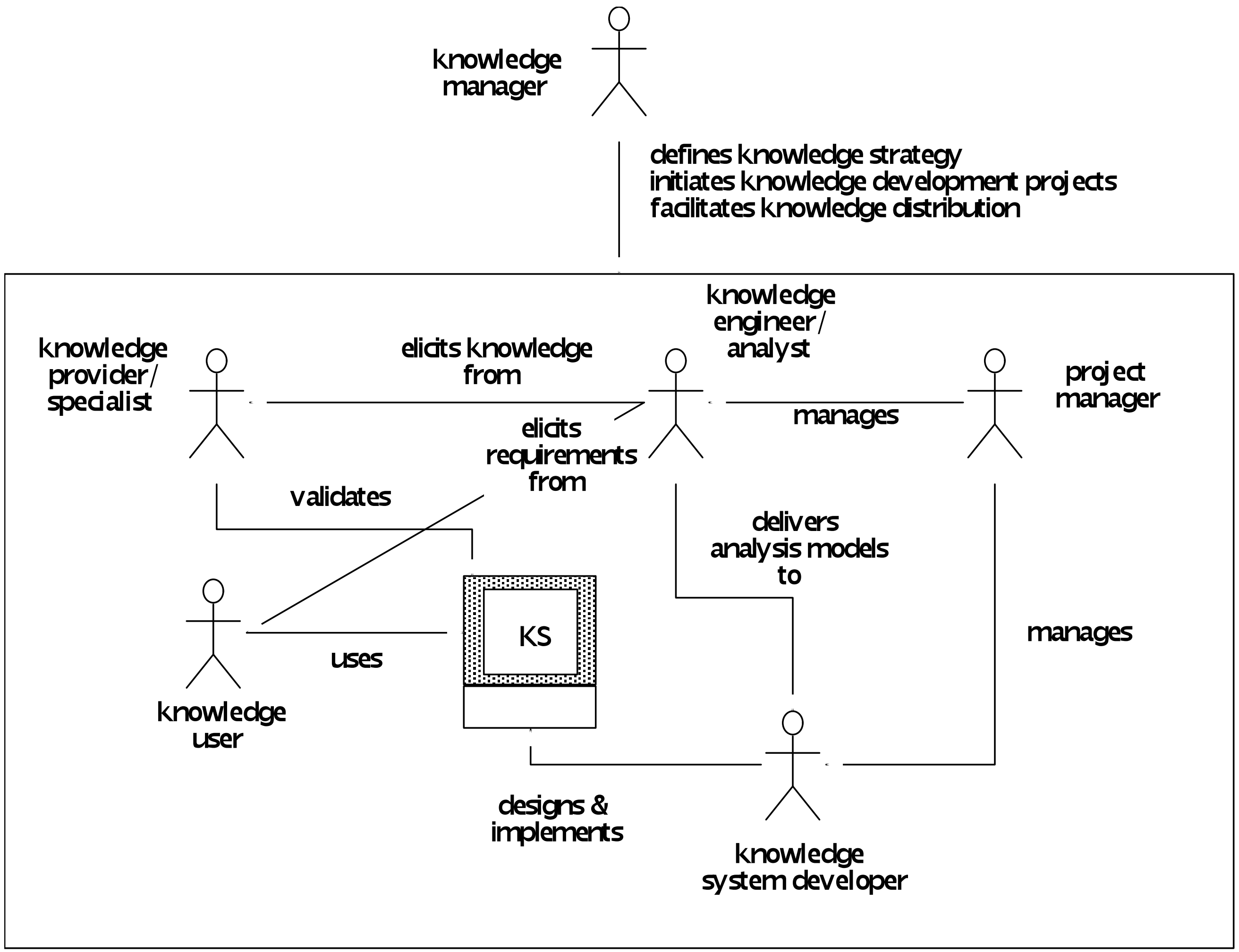
- responsible for planning, scheduling and monitoring development work
- liaises with client
- typically medium-size projects (4-6 people)
- profits from structured approach

Knowledge manager

- background role
- monitors organizational purpose of
  - system(s) developed in a project
  - knowledge assets developed/refined
- initiates (follow-up) projects
- should play key role in reuse
- may help in setting up the right project team

Roles in knowledge-system development

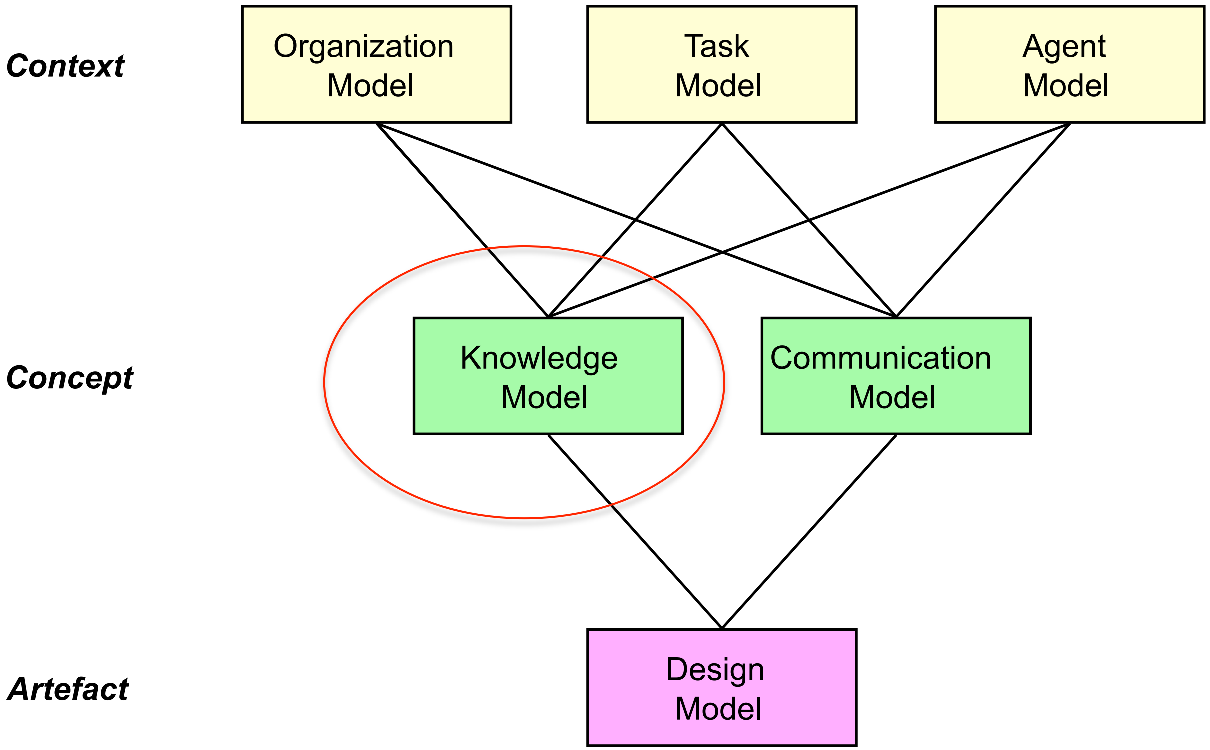
STI · INNSBRUCK



Knowledge model components

Knowledge model

- specialized tool for specification of knowledge-intensive tasks
- abstracts from communication aspects
- real-world oriented
- reuse is central theme



Knowledge categories

- Domain knowledge
  - relevant domain knowledge and information
  - static
- Inference knowledge
  - basic reasoning steps that can be made in the domain knowledge and are applied by tasks
- Task knowledge
  - goal-oriented
  - functional decomposition

Knowledge Categories: domain knowledge

STI · INNSBRUCK

- domain schema
  - schematic description of knowledge and information types
  - defined through domain constructs
- knowledge base
  - set of knowledge instances

Constructs for domain schema

- Concept
  - cf. object class (without operations)
- Relation
  - cf. association
- Attribute
  - primitive value
- Rule type
  - introduces expressions

Example: car concepts

gas dial

value: gas dial

fuel tank

status: { full, almost-empty, empty}

```
CONCEPT gas dial;
  ATTRIBUTES:
     value: dial-value;
END CONCEPT gas-dial;


VALUE-TYPE dial-value;
  VALUE-LIST:    {zero, low, normal};
  TYPE: ORDINAL;
END VALUE-TYPE dial-value;
```

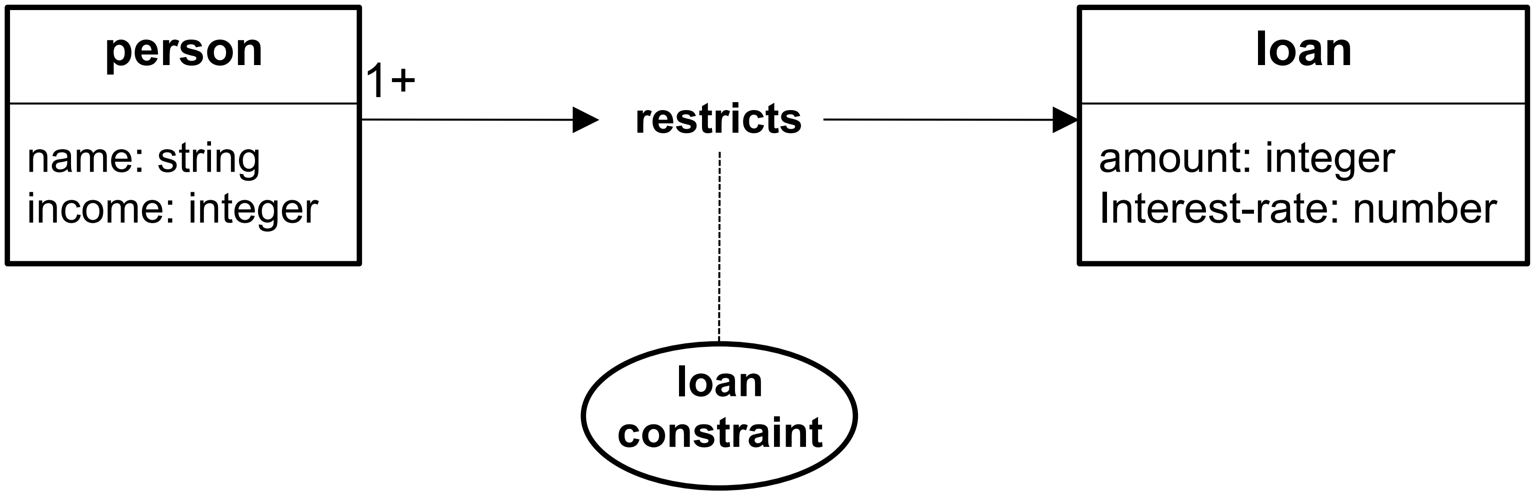
```
CONCEPT fuel-tank;
  ATTRIBUTES:
     status: {full, almost-empty,
                empyt};
END CONCEPT fuel-tank;
```

Modelling rules

- knowledge analysis is focused on finding rules with a common structure
- a rule as an instance of a rule type
- models a relation between *expressions about* feature values (e.g. attribute values)

  gas-dial.value = zero -> fuel-tank.status = empty
- models set of real-world "rules" with a similar structure

Example rule type



person.income <= 10,000
    RESTRICTS
loan.amount <= 2,000

person.income > 10,000 AND person.income <= 20,000
    RESTRICTS
loan.amount <= 3,000

Rule type structure

- <antecedent> <connection-symbol> <consequent>
- example rule:

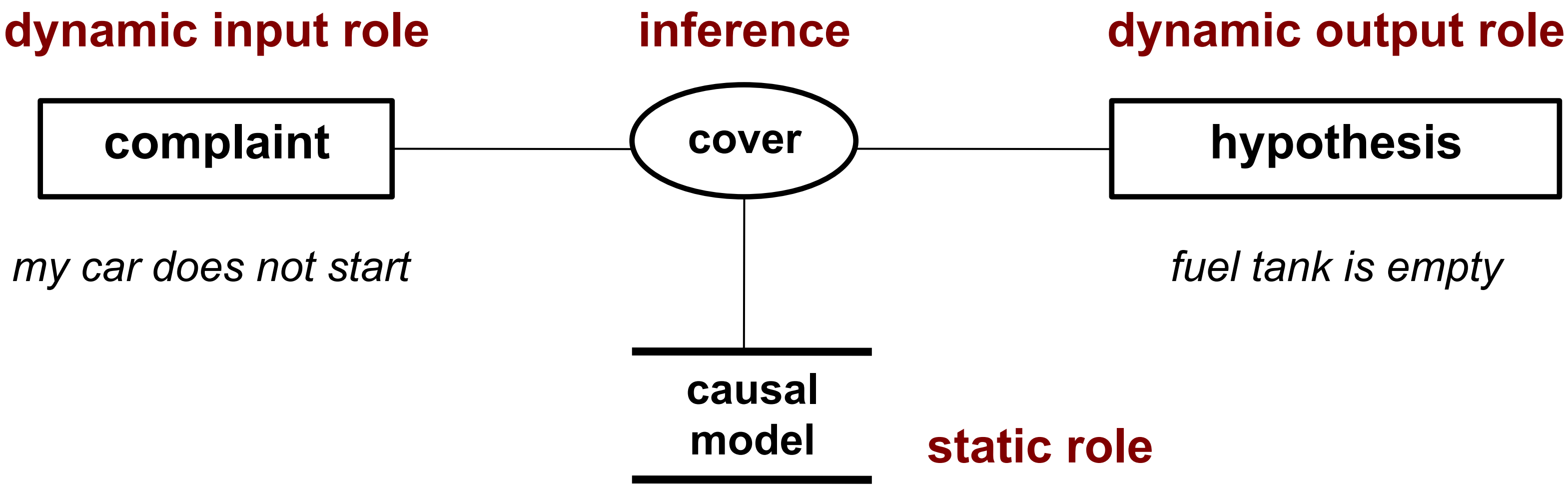
```
fuel-supply.status = blocked
  CAUSES
gas-in-engine.status = false;
```

- flexible use for almost any type of dependency
  - multiple types for antecedent and consequent

Inference knowledge

- describes the lowest level of functional decomposition
- basic information-processing units:
  - inference => reasoning
  - transfer function => communication with other agents
- why special status?
  - indirectly related to domain knowledge
  - enables reuse of inference

Example inference: cover



fuel tank is empty leads to lack of gas in engine
if there is no gas in the the engine, then the car does not start

Task knowledge

- describes goals
  - assess a mortgage application in order to minimize the risk of losing money
  - find the cause of a malfunction of a photocopier in order to restore service.
  - design an elevator for a new building.
- describes strategies that can be employed for realizing goals.
- typically described in a hierarchical fashion

Task

- Description of the input/output
- Main distinction with traditional functions is that the data manipulated by the task are (also) described in a domain-independent way.
  - example, the output of a medical diagnosis task would not be a “disease” but an abstract name such as “fault category”

Template knowledge models

Lessons

- Knowledge models partially reused in new applications
- Type of task = main guide for reuse
- Catalog of task templates

The need for reuse

STI · INNSBRUCK

- prevent "re-inventing the wheel"
- cost/time efficient
- decreases complexity
- **quality-assurance**

Task template

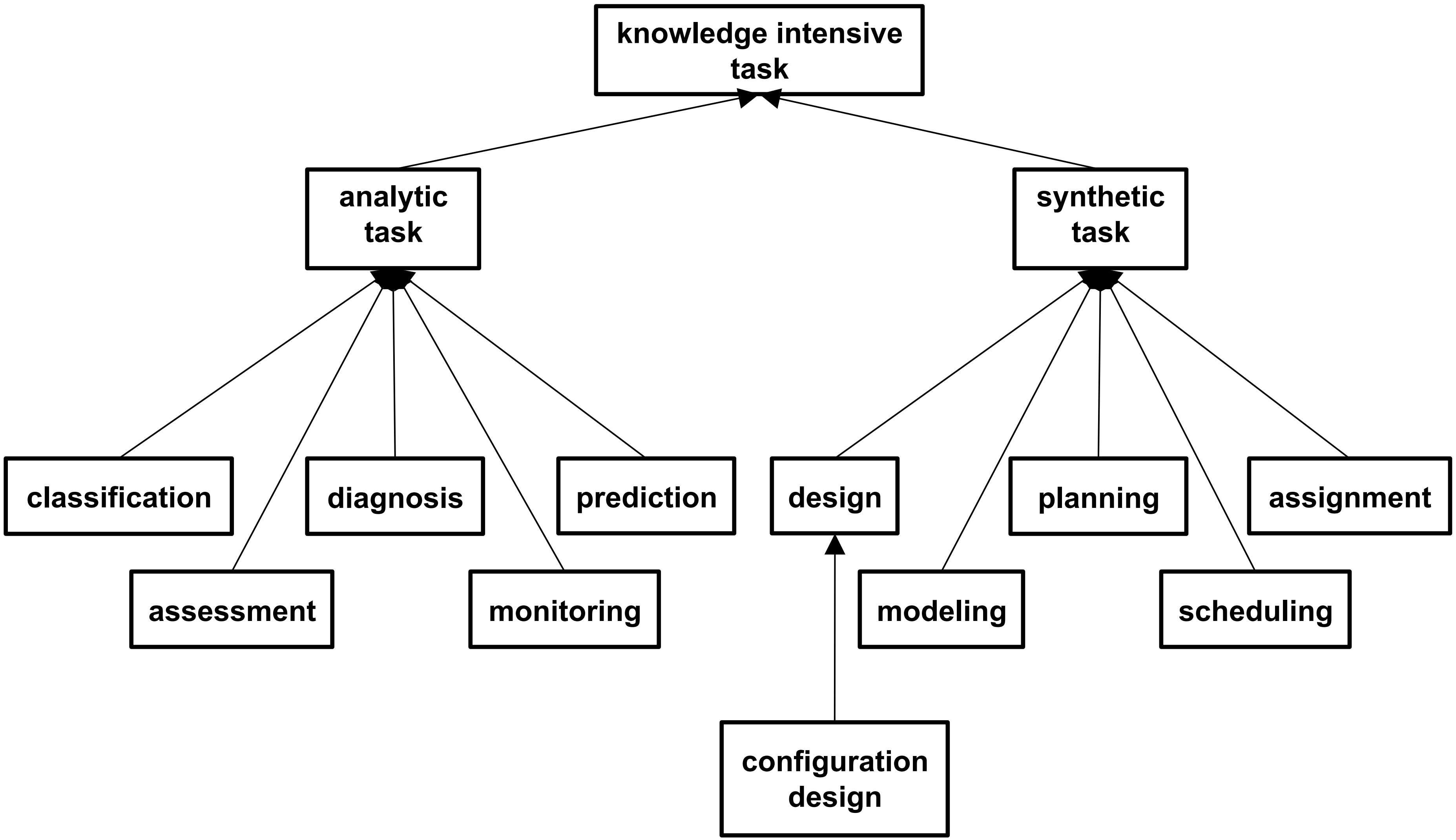
- reusable combination of model elements
  - (provisional) inference structure
  - typical control structure
  - typical domain schema from task point-of-view
- specific for a task type
- supports top-down knowledge modeling

Analytic versus synthetic tasks

- analytic tasks
  - system pre-exists
    - it is typically not completely "known"
  - input: some data about the system,
  - output: some characterization of the system
- synthetic tasks
  - system does not yet exist
  - input: requirements about system to be constructed
  - output: constructed system description

Task hierarchy

STI · INNSBRUCK



Structure of template description in catalog

STI · INNSBRUCK

- General characterization
  - typical features of a task
- Default method
  - roles, sub-functions, control structure, inference structure
- Typical variations
  - frequently occurring refinements/changes
- Typical domain-knowledge schema
  - assumptions about underlying domain-knowledge structure

Classification

- establish correct class for an object
- object should be available for inspection
  - "natural" objects
- examples: rock classification, apple classification
- terminology: object, class, attribute, feature
- one of the simplest analytic tasks; many methods
- other analytic tasks: sometimes reduced to classification problem especially diagnosis

Assessment

STI · INNSBRUCK

- find decision category for a case based on domain-specific norms.
- typical domains: financial applications (loan application), community service
- terminology: case, decision, norms
- some similarities with monitoring
  - differences:
    - timing: assessment is more static
    - different output: decision versus discrepancy

Diagnosis

- find fault that causes system to malfunction
  - example: diagnosis of a copier
- terminology:
  - complaint/symptom, hypothesis, differential, finding(s)/evidence, fault
- nature of fault varies
  - state, chain, component
- should have some model of system behavior
  - default method: simple causal model
- sometimes reduced to classification task
  - direct associations between symptoms and faults
- automation feasible in technical domains

Monitoring

STI · INNSBRUCK

- analyze ongoing process to find out whether it behaves according to expectations
- terminology:
  - parameter, norm, discrepancy, historical data
- main features:
  - dynamic nature of the system
  - cyclic task execution
- output "just" discrepancy => no explanation
- often: coupling monitoring and diagnosis
  - output monitoring is input diagnosis

Prediction

- analytic task with some synthetic features
- analyses current system behavior to construct description of a system state at future point in time.
- example: weather forecasting
- often sub-task in diagnosis
- also found in knowledge-intensive modules of teaching systems e.g. for physics.
- inverse: retrodiction: big-bang theory

Synthesis

- Given a set of requirements, construct a system description that fulfills these requirements

requirements (external)		constraints & preferences (internal)
soft requirement	→	preference
"fast system"		"prefer cheapest component"
hard requirement	→	constraint
"price lower than $2,000"		"P 166 processor requires 16Mb"

“Ideal” synthesis method

- Operationalize requirements
  - preferences and constraints
- Generate all possible system structures
- Select sub-set of valid system structures
  - obey constraints
- Order valid system structures
  - based on preferences

Design

STI · INNSBRUCK

- synthetic task
- system to be constructed is physical artifact
- example: design of a car
- can include creative design of components
- creative design is too hard a nut to crack for current knowledge technology
- sub-type of design which excludes creative design => configuration design

Configuration design

STI · INNSBRUCK

- given predefined components, find assembly that satisfies requirements + obeys constraints
- example: configuration of an elevator; or PC
- terminology: component, parameter, constraint, preference, requirement (hard & soft)
- form of design that is well suited for automation
- computationally demanding

Assignment

- create mapping between two sets of objects
  - allocation of offices to employees
  - allocation of airplanes to gates
- mapping has to satisfy requirements and be consistent with constraints
- terminology
  - subject, resource, allocation
- can be seen as a “degenerative” form of configuration design

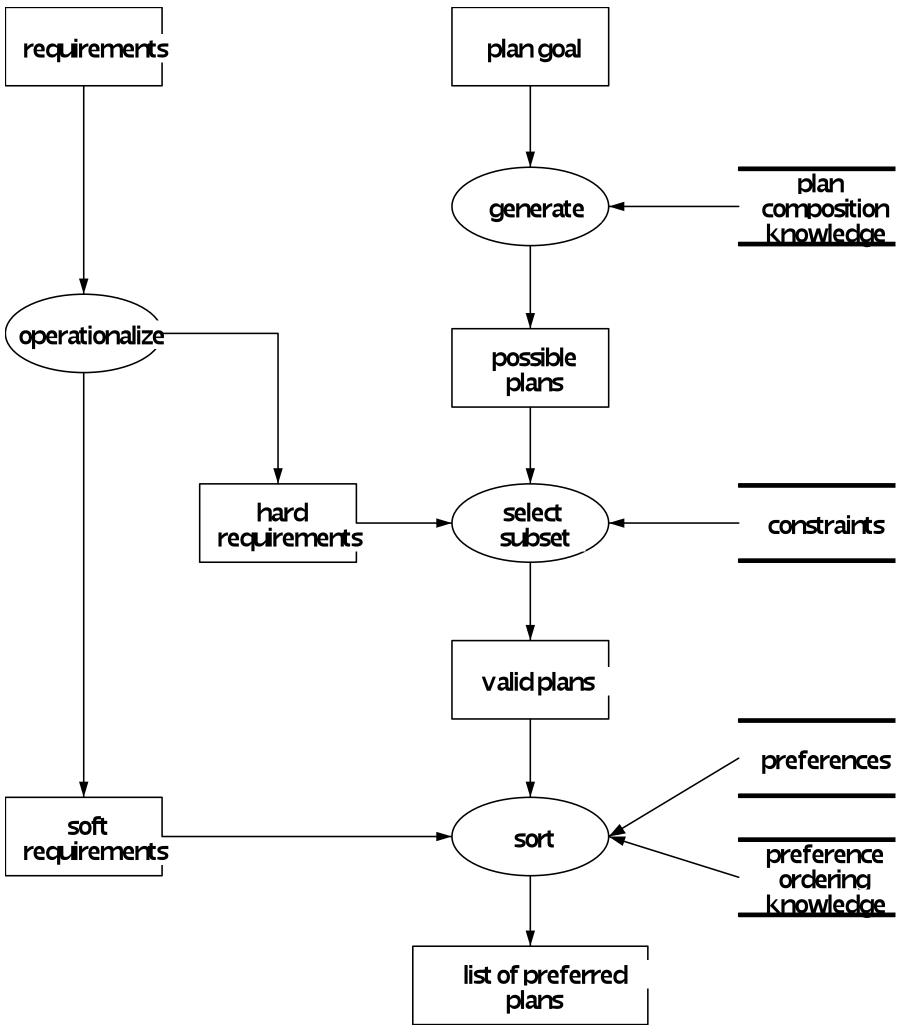
Planning

STI · INNSBRUCK

- shares many features with design
- main difference: "system" consists of activities plus time dependencies
- examples: travel planning; planning of building activities
- automation only feasible, if the basic plan elements are predefined
- consider use of the general synthesis method (e.g therapy planning) or the configuration-design method

Planning method

STI · INNSBRUCK



Scheduling

- Given a set of predefined jobs, each of which consists of temporally sequenced activities called units, assign all the units to resources at time slots
  - production scheduling in plant floors
- Terminology: job, unit, resource, schedule
- Often done after planning (= specification of jobs)
- Take care: use of terms “planning” and “scheduling” differs

In applications: typical task combinations

- monitoring + diagnosis
  - Production process
- monitoring + assessment
  - Nursing task
- diagnosis + planning
  - Troubleshooting devices
- classification + planning
  - Military applications

Knowledge model construction

Process & Product

- so far: focus on knowledge model as product
- bottleneck for inexperienced knowledge modelers
  - how to undertake the process of model construction.
- solution: process model
  - as prescriptive as possible
  - process elements: stage, activity, guideline, technique
- but: modeling is constructive activity
  - no single correct solution nor an optimal path
- support through a number of guidelines that have proven to work well in practice.
- knowledge modeling is specialized form of requirements specification
  - general software engineering principles apply

Stages in Knowledge-Model Construction

STAGES	TYPICAL ACTIVITIES
knowledge identification	– domain familiarization (information sources, glossary, scenarios) – list potential model components for reuse (task– and domain–related components)
knowledge specification	– choose task template (provides initial task decomposition) – construct initial domain conceptualization (main domain information types) – complete knowledge–model specification (knowledge model with partial knowledge bases)
knowledge refinement	– validate knowledge model (paper simulation, prototype of reasoning system) – knowledge–base refinement (complete the knowledge bases)

Stage 1: Knowledge identification

- goal
  - survey the knowledge items
  - prepare them for specification
- input
  - knowledge-intensive task selected
  - main knowledge items identified.
  - application task classified
    - assessment, configuration, combination of task types
- activities
  - explore and structure the information sources
  - study the nature of the task in more detail

Exploring information sources

- Factors
  - Nature of the sources
    - well-understood?, theoretical basis?
  - Diversity of the sources
    - no single information source (e.g. textbook or manual)
    - diverse sources may be conflicting
    - multiple experts is a risk factor.
- Techniques
  - text marking in key information sources
  - some structured interviews to clarify perceived holes in domain
- main problem:
  - find balance between learning about the domain without becoming a full

Guidelines

- Talk to people in the organization who have to talk to experts but are not experts themselves
- Avoid diving into detailed, complicated theories unless the usefulness is proven
- Construct a few typical scenarios which you understand at a global level
- Never spend too much time on this activity. Two person weeks should be maximum.

Results exploration

- Tangible
  - Listing of domain knowledge sources, including a short characterization.
  - Summaries of selected key texts.
  - Glossary/lexicon
  - Description of scenarios developed.
- Intangible
  - your own understanding of the domain
    - most important result

List potential components

- goal: pave way for reusing components
- two angles on reuse:
  - Task dimension
    - check task type assigned in Task Model
    - build a list of task templates
  - Domain dimension
    - type of the domain: e.g. technical domain
    - look for standardized descriptions

      AAT for art objects ontology libraries, reference models, product model libraries

Stage 2: Knowledge specification

- goal: complete specification of knowledge except for contents of domain models
  - domain models need only to contain example instances
- activities
  - Choose a task template.
  - Construct an initial domain conceptualization.
  - Specify the three knowledge categories

Choose task template

STI · INNSBRUCK

- baseline: strong preference for a knowledge model based on an existing application.
  - efficient, quality assurance
- selection criteria: features of application task
  - nature of the output: fault category, plan
  - nature of the inputs: kind of data available
  - nature of the system: artifact, biological system
  - constraints posed by the task environment:
    - required certainty, costs of observations.

Guidelines for template selection

- prefer templates that have been used more than once
  - empirical evidence
- construct annotated inference structure (and domain schema)
- if no template fits: question the knowledge-intensity of the task

Guidelines

STI · INNSBRUCK

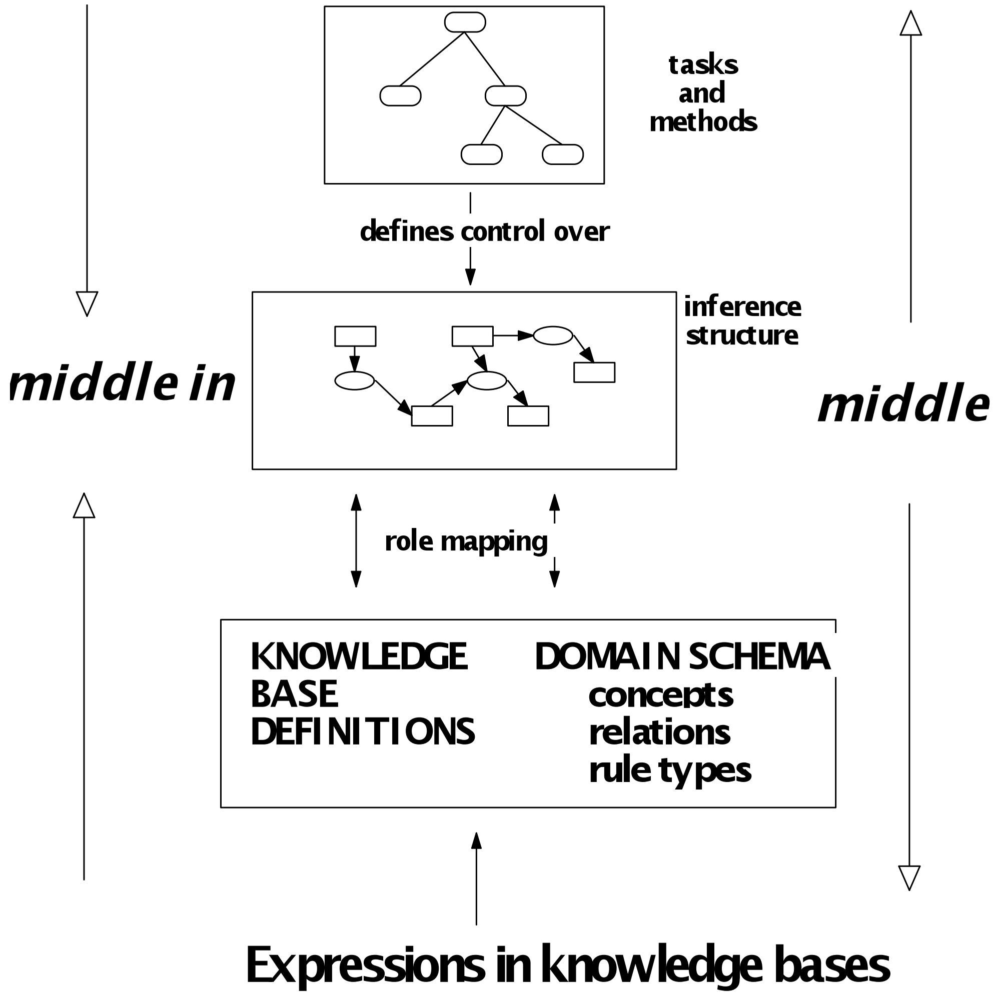
- use as much as possible existing data models:
  - useful to use at least the same terminology basic constructs
  - makes future cooperation/exchange easier
- limit use of the knowledge-modeling language to concepts, sub-types and relations
  - concentrate on "data"
  - similar to building initial class model
- If no existing data models can be found, use standard SE techniques for finding concepts and relations
  - use “pruning” method
- Constructing the initial domain conceptualization should be done in parallel with the choice of the task template
  - otherwise: fake it

Complete model specification

STI · INNSBRUCK

- Route 1: Middle-out
  - Start with the inference knowledge
  - Preferred approach
  - Precondition: task template provides good approximation of inference structure.
- Route 2: Middle-in
  - Start in parallel with task decomposition and domain modeling
  - More time-consuming
  - Needed if task template is too coarse-grained

Middle-in and Middle-out



Guidelines

- inference structure is detailed enough, if the explanation it provides is sufficiently detailed
- inference structure is detailed enough if it is easy to find for each inference a single type of domain knowledge that can act as a static role for this inference

Guidelines for specifying task knowledge

STI · INNSBRUCK

- begin with the control structure
  - "heart" of the method
- neglect details of working memory
  - design issue
- choose role names that clearly indicate role
  - "modeling is naming"
- do not include static knowledge roles
- real-time applications: consider using a different representation than pseudo code
  - but: usage of "receive"

Guidelines for specifying domain knowledge

STI · INNSBRUCK

- domain-knowledge type used as static role not required to have exactly the “right’” representation
  - design issue;
  - key point: knowledge is available.
- scope of domain knowledge is typically broader than what is covered by inferences
  - requirements of communication, explanation

Stage 3: Knowledge Refinement

- Validate knowledge model
- Fill contents of knowledge bases

Fill contents of knowledge bases

- schema contains two kinds of domain types:
  - information types that have instances that are part of a case
  - knowledge types that have instances that are part of a domain model
- goal of this task: find (all) instances of the latter type
- case instances are only needed for a scenario

Guidelines for filling contents

- filling acts as a validation test of the schema
- usually not possible to define full, correct knowledge base in the first cycle
- knowledge bases need to be maintained
  - knowledge changes over time
- techniques:
  - incorporate editing facilities for KB updating, trace transcripts, structured interview, automated learning, map from existing knowledge bases

Validate knowledge model

STI · INNSBRUCK

- internally and externally
- verification = internal validation
  - “is the model right?”
- validation = validation against user requirements
  - "is it the right model?"

Validation techniques

- Internal
  - structured walk-troughs
  - software tools for checking the syntax and find missing parts
- External
  - usually more difficult and/or more comprehensive.
  - main technique: simulation
    - paper-based simulation
    - prototype system

Paper-based simulation

STI · INNSBRUCK

the user says: "the car does not start"	DIAGNOSIS: <u>Complaint</u>: engine-behavior.status = does-not-start	Complaint is received, for which a diagnostic task is started
a possible cause is that the fuel tank is empty	COVER: <u>hypothesis</u>; fuel-tank.status = empty	One of the three possible causes is produced.
in that case we would expect the gas indicator to be low	PREDICT: <u>expected-finding</u>: gas-dial.value = zer	The expected finding provides us with a way of getting supporting evidence for hypothesis

Maintenance

STI · INNSBRUCK

- model development is a cyclic process
- models act as information repositories
  - continuously updated
- but: makes requirements for support tools stronger
  - transformation tools

Domain Documentation Document

- Knowledge model specification
- list of all information sources used.
- list of model components that we considered for reuse.
- scenarios for solving the application problem.
- results of the simulations undertaken during validation
- Elicitation material (appendices)

Summary process

- Knowledge identification
  - familiarization with the application domain
- Knowledge specification
  - detailed knowledge analysis
  - supported by reference models
- Knowledge refinement
  - completing the knowledge model
  - validating the knowledge model
- Feedback loops may be required
  - simulation in third stage may lead to changes in specification
  - Knowledge bases may require looking for additional knowledge sources.
  - general rule: feedback loops occur less frequently, if the application problem is well-understood and similar problems have been tackled

STI · INNSBRUCK

Knowledge elicitation techniques

Elicitation of expertise

- Time-consuming
- Multiple forms
  - e.g. theoretical, how-to-do-it
- Multiple experts
- Heuristic nature
  - distinguish empirical from heuristic
- Managing elicitation efficiently
  - knowledge about when to use particular techniques

Expert types

- Academic
  - Regards domain as having a logical structure
  - Talks a lot
  - Emphasis on generalizations and laws
  - Feels a need to present a consistent “story”: teacher
  - Often remote from day-to-day problem solving
- Practitioner
  - Heavily into day-to-day problem solving
  - Implicit understanding of the domain
  - Emphasis on practical problems and constraints
  - Many heuristics

Human limitations and biases

STI · INNSBRUCK

- Limited memory capacity
- Context may be required for knowledge recollection
- Prior probabilities are typically under-valued
- Limited deduction capabilities

Elicitation techniques

STI · INNSBRUCK

- Interview
- Self report / protocol analysis
- Laddering
- Concept sorting
- Repertory grids

STI · INNSBRUCK

Interview

Interview: Session preparation

- Establish goal of the session
- Consider added value for expert
- Describe for yourself a profile of the expert
- List relevant questions
- Write down opening and closing statement
- Check recording equipment
  - audio recording is usually sufficient
- Make sure expert is aware of session context: goal, duration, follow-up, et cetera

Interview: Start of the session

- Introduce yourself (if required)
- Clarify goal and expectations
- Indicate how the results will be used
- Ask permission for tape recording
- Privacy issues
- Check whether the expert has some questions left
- Create as much as possible a mutual trust

Interview: During the session

STI · INNSBRUCK

- Avoid suggestive questions
- Clarify reason of question
- Phrase questions in terms of probes
  - e.g, “why …”
- Pay attention to non-verbal aspects
- Be aware of personal biases
- Give summaries at intermediate points

Interview: End of the session

STI · INNSBRUCK

- Restate goal of the session
- Ask for additional/qualifying
- Indicate what will be the next steps
- Make appointments for the next meetings
- Process interview results ASAP.
- Organize feedback round with expert
- Distribute session results

Unstructured interview

- No detailed agenda
- Few constraints
- Delivers diverse, incomplete data
- Used in early stages: feasibility study, knowledge identification
- Useful to establish a common basis with expert
  - s/he can talk freely

Structured interview

- Knowledge engineer plans and directs the session
- Takes form of provider-elicitor dialogue
- Delivers more focused expertise data
- Often used for “filling in the gaps” in the knowledge base
  - knowledge refinement phase
- Also useful at end of knowledge identification or start of knowledge specification
- Always create a transcript

Interview structure for domain-knowledge elicitation

STI · INNSBRUCK

- Identify a particular sub-task
  - should be relatively small task, e.g. an inference
- Ask expert to identify “rules” used in this task
- Take each rule, and ask when it is useful and when not
- Use fixed set of probes:
  - “Why would you do that?”
  - “How would you do that?”
  - “When would you do that?”
  - “What alternatives are there for this action?”
  - “What if …?”
  - “Can you tell me more about ..?”

Interview pitfalls

STI · INNSBRUCK

- Experts can only produce what they can verbalize
- Experts seek to justify actions in any way they can
  - "spurious justification"
- Therefore: supplement with techniques that observe expertise "in action"
  - e.g. self report

Self report

Self report

STI · INNSBRUCK

- Expert performs a task while providing a running commentary
  - expert is “thinking aloud”
- Session protocol is always transcribed
  - input for protocol analysis
- Variations:
  - *shadowing:* one expert performs, a second expert gives a running commentary
  - *retrospection*: provide a commentary after the problem-solving session
- Theoretical basis: cognitive psychology

Requirements for self-report session

- Knowledge engineer must be sufficiently acquainted with the domain
- Task selection is crucial
  - only a few problems can be tackled
  - selection typically guided by available scenario's and templates
- Expert should not feel embarrassed
  - consider need for training session

Analyzing the self-report protocol

STI · INNSBRUCK

- Use a reference model as a coding scheme for text fragments
  - Task template
- Look out for “when”-knowledge
  - Task-control knowledge
- Annotations and mark-ups can be used for domain-knowledge acquisition
- Consider need for tool support

Self report guidelines and pitfalls

- Present problems in a realistic way
- Transcribe sessions as soon as possible
- Avoid long sessions (maximum = 20 minutes)
- Presence of knowledge engineer is important
- Be aware of scope limitations
- Verbalization may hamper performance
- Knowledge engineer may lack background knowledge to notice distinctions

Use of self reports

- Knowledge specification stage
- Validation of the selection of a particular reference model
- Refining / customizing a task template for a specific application
- If no adequate task template model is available: use for bottom-up reasoning model construction
  - but: time-consuming

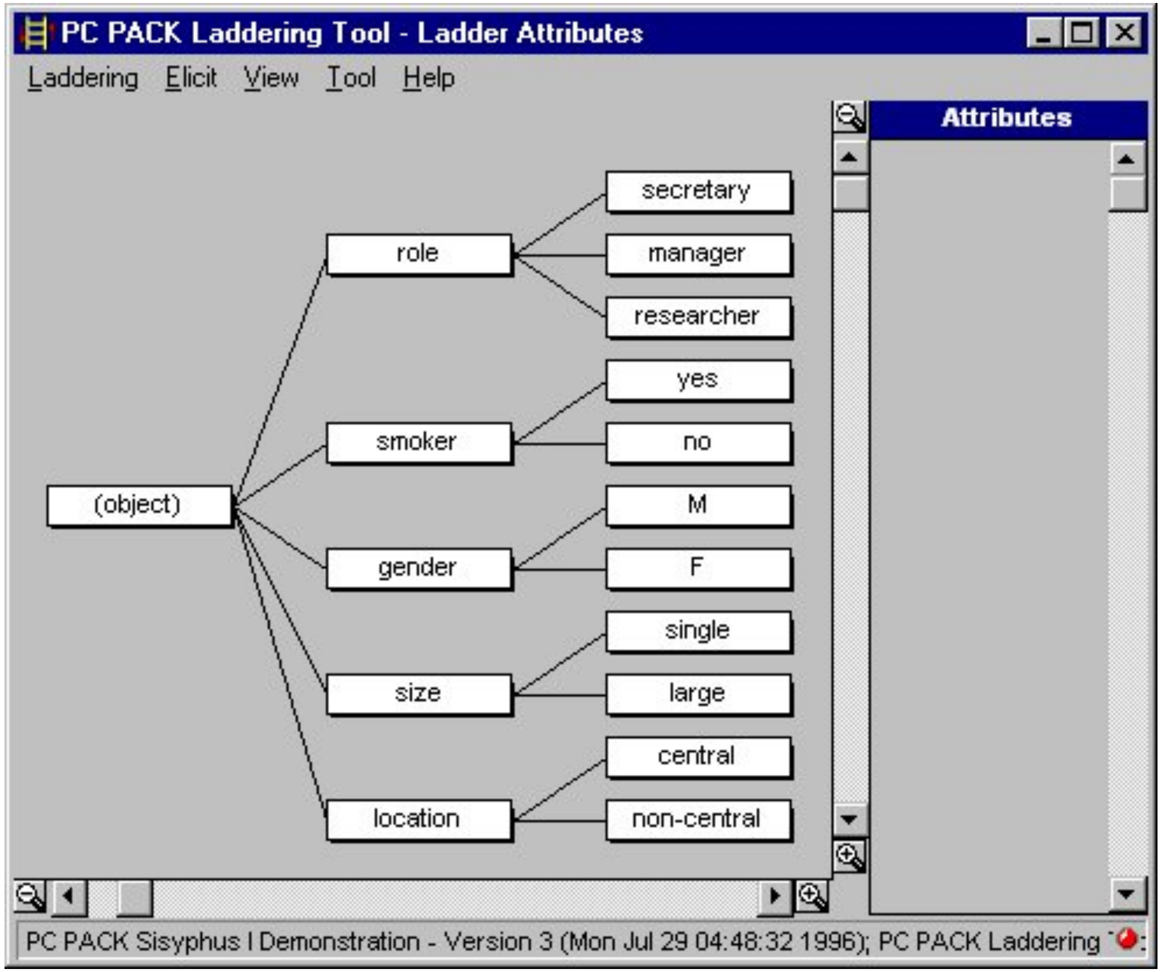
Laddering

Laddering

STI · INNSBRUCK

- Organizing entities in a hierarchy
- Hierarchies are meant as pre-formal structures
- Nodes can be of any type
  - class, process, relation, ….
- Useful for the initial phases of domain-knowledge structuring
  - in particular knowledge identification
- Can be done by expert
  - tool support

Example ladder

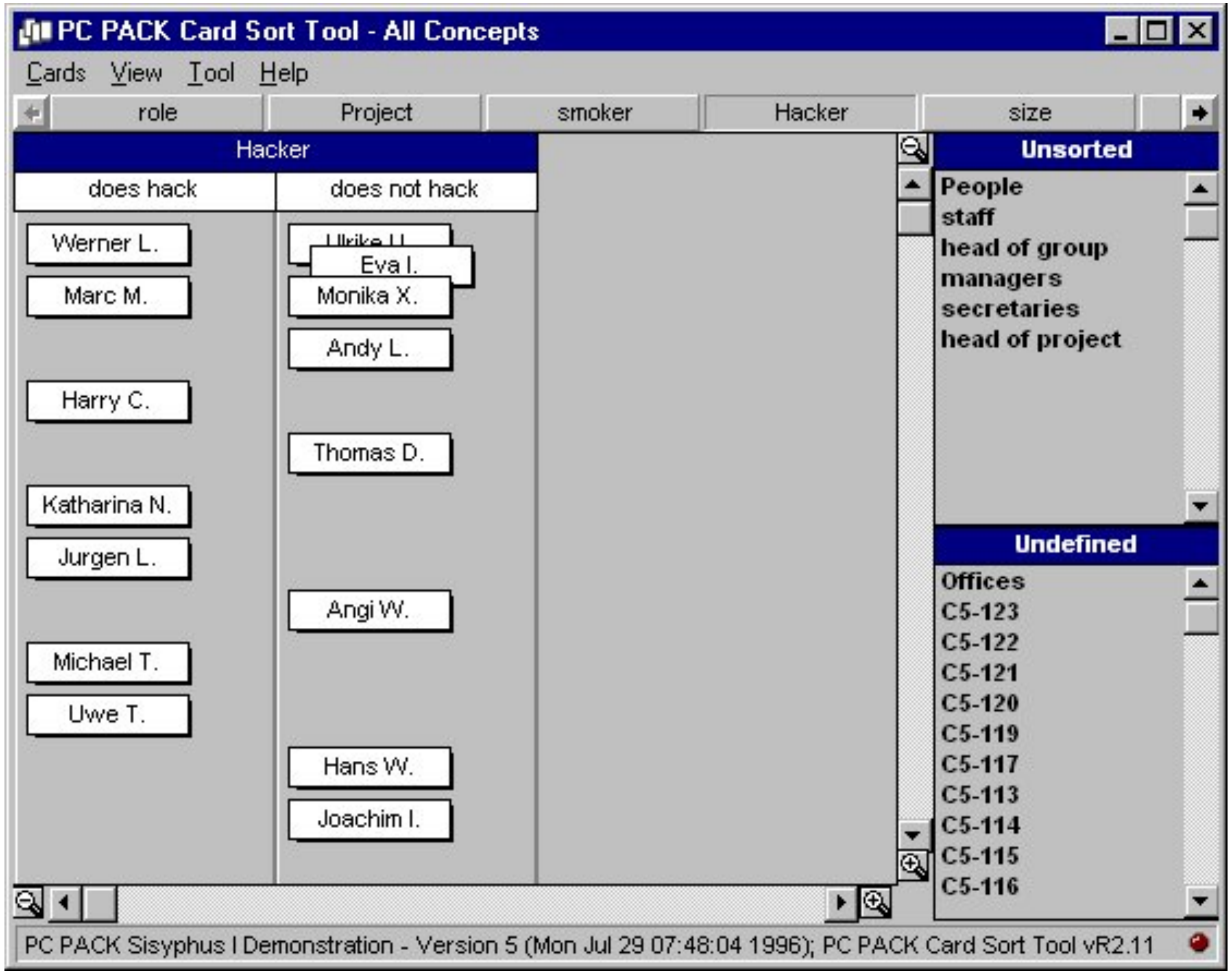


Concept sorting

Concept sorting

- Technique:
  - present expert with shuffled set of cards with concept names
  - expert is asked to sort cards in piles
- Helps to find relations among a set of concepts
- Useful in case of subtle dependencies
- Simple to apply
- Complementary to repertory grids
  - concept sort: nominal categories
  - repertory grid: ordinal categories

Card sort tool



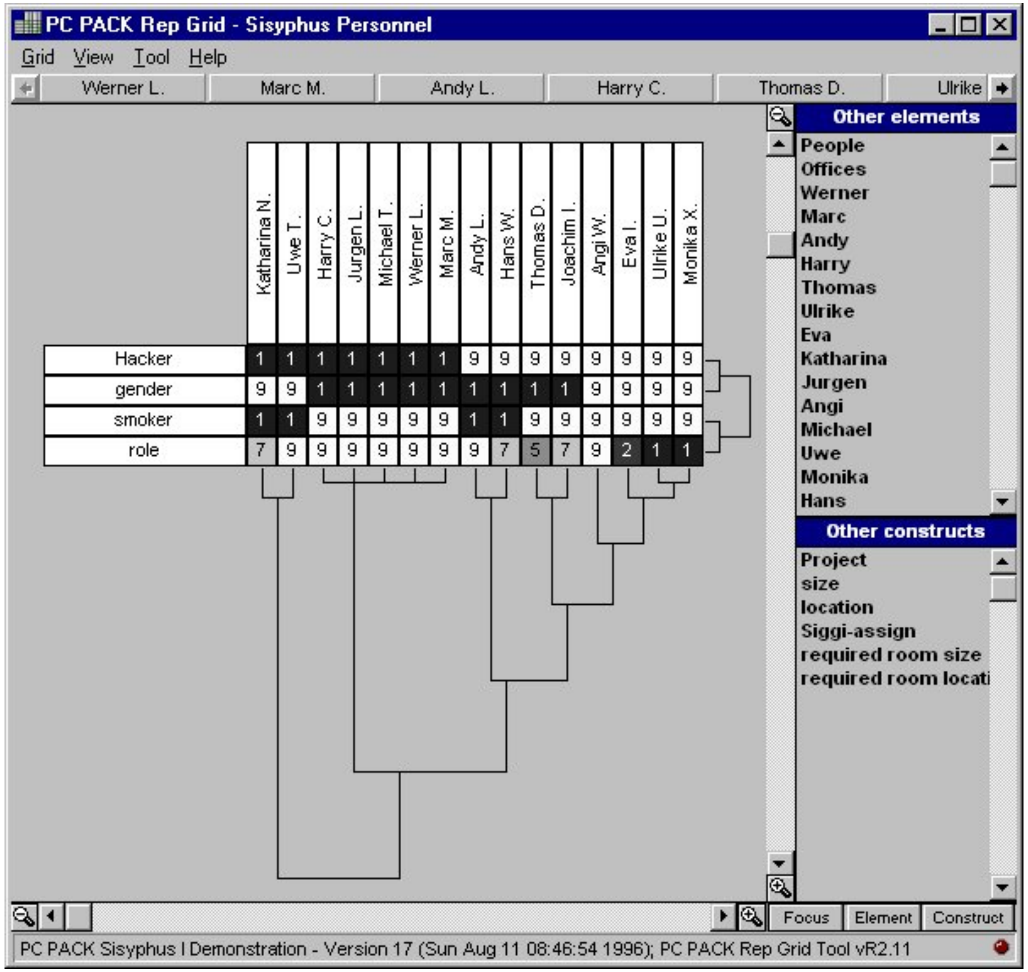
Repertory grids

Repertory grid

STI · INNSBRUCK

- Based on personal construct theory (Kelly, 1955)
- Subject: discriminate between triads of concepts
  - Mercury and Venus versus Jupiter
- Subject is asked for discriminating feature
  - E.g. “planet size”
- Re-iterate until no new features are found
- Rate all concepts with respect to all features
- Matrix is analyzed with cluster analysis
- Result: suggestions for concept relations
- Tool support is required

Example grid



When to use which technique?

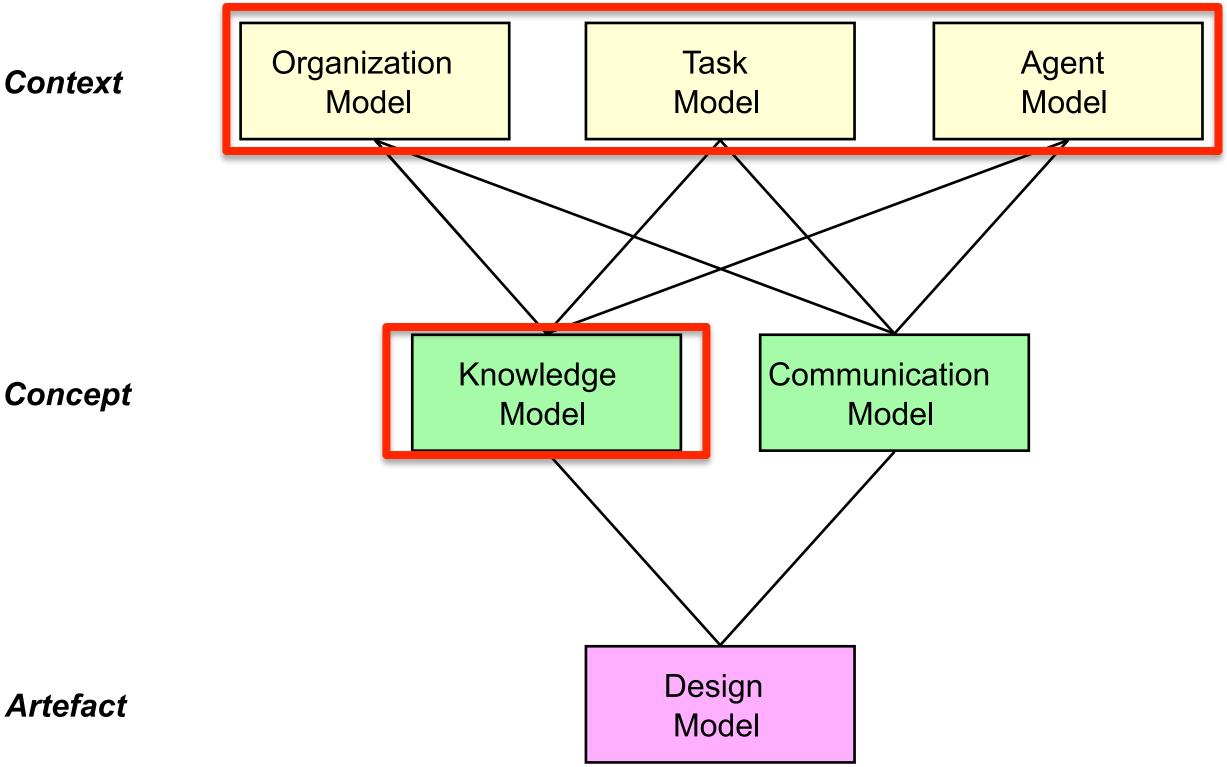
- Knowledge identification
  - Unstructured interview, laddering
- Knowledge specification
  - Domain schema: concept sorting, repertory grid
  - Template selection: self report
  - Task & inference knowledge: self report
- Knowledge refinement
  - Structured interview

EXAMPLE

Housing application

STI · INNSBRUCK

- An application for assigning houses to potential renters
- We now sketch the organization, task and agent model and build the knowledge model on top.



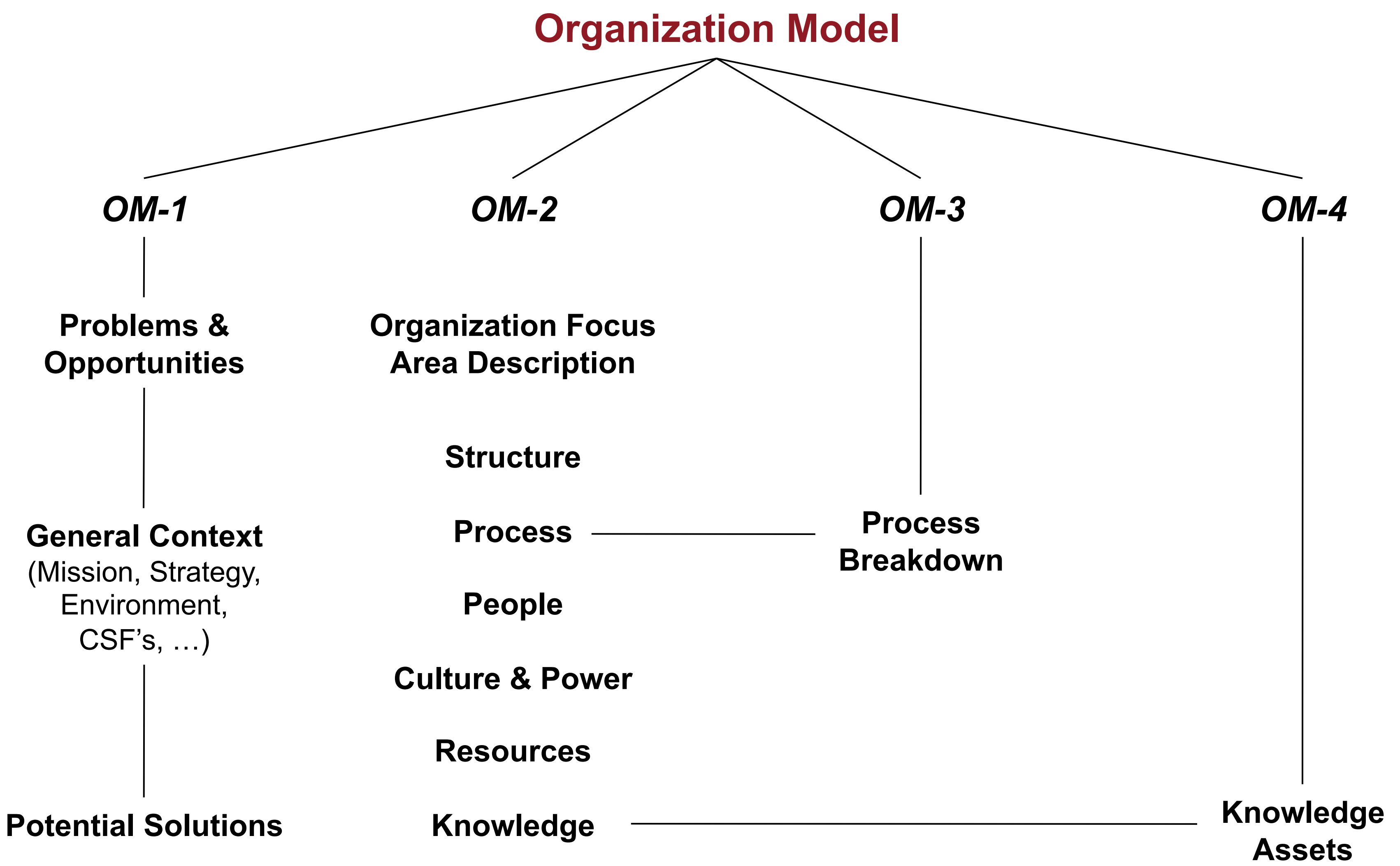
Problem description

STI · INNSBRUCK

- Local government institution is responsible for assignment of rental houses to applicants
- Transparent assignment procedure
  - two-weekly magazine with house offers
  - publication of results
- Partially automated process
- Existing databases of applicants and residences

Organization models

STI · INNSBRUCK



Organization Model	**Problems and Opportunities Worksheet Organization model 1**
Problems and opportunities	assessment takes too much time not sufficient time for urgent cases
Organizational context	**Mission**: transparency of procedure, clear applicant responsibility **External actors**: local council, public opinion, national regulations, … **Strategy**: broaden scope of market
Solutions	1. Automated assessment system & 2. Training program for assessors to be come urgency handlers

Organization model 2

Organization model	Variant aspects: Worksheet Organization model 2
Resources	**Existing database** of applicants and residences **Priority calculator** for computing a priority list of applicants for a residence.
Knowledge	**Assessment criteria**: knowledge for judging correctness of individual applications **Assignment rules**: knowledge used for selecting an applicant for a particular house. **Urgency rules**: special rules and regulations for urgent cases (e.g., handicapped people).
Culture & power	Hierarchical organization Employees view the future with some trepidation Management style: history as civil servant

Organization model 3

STI · INNSBRUCK

Task	Performed by	Where	Knowledge asset (s)	KI?	Signifi-cance
1. Magazine production	Magazine editor	Public service	-	No	3
2. Data entry applications	Data typist / automated telephone	Residence assignment	-	No	2
3. Application assessment	Assigner	Residence assignment	Assessment criteria	Yes	5
4. Residence assignment	Assigner	Residence Assignment	Assignment & urgency rules	Yes	5

Organization model 4

STI · INNSBRUCK

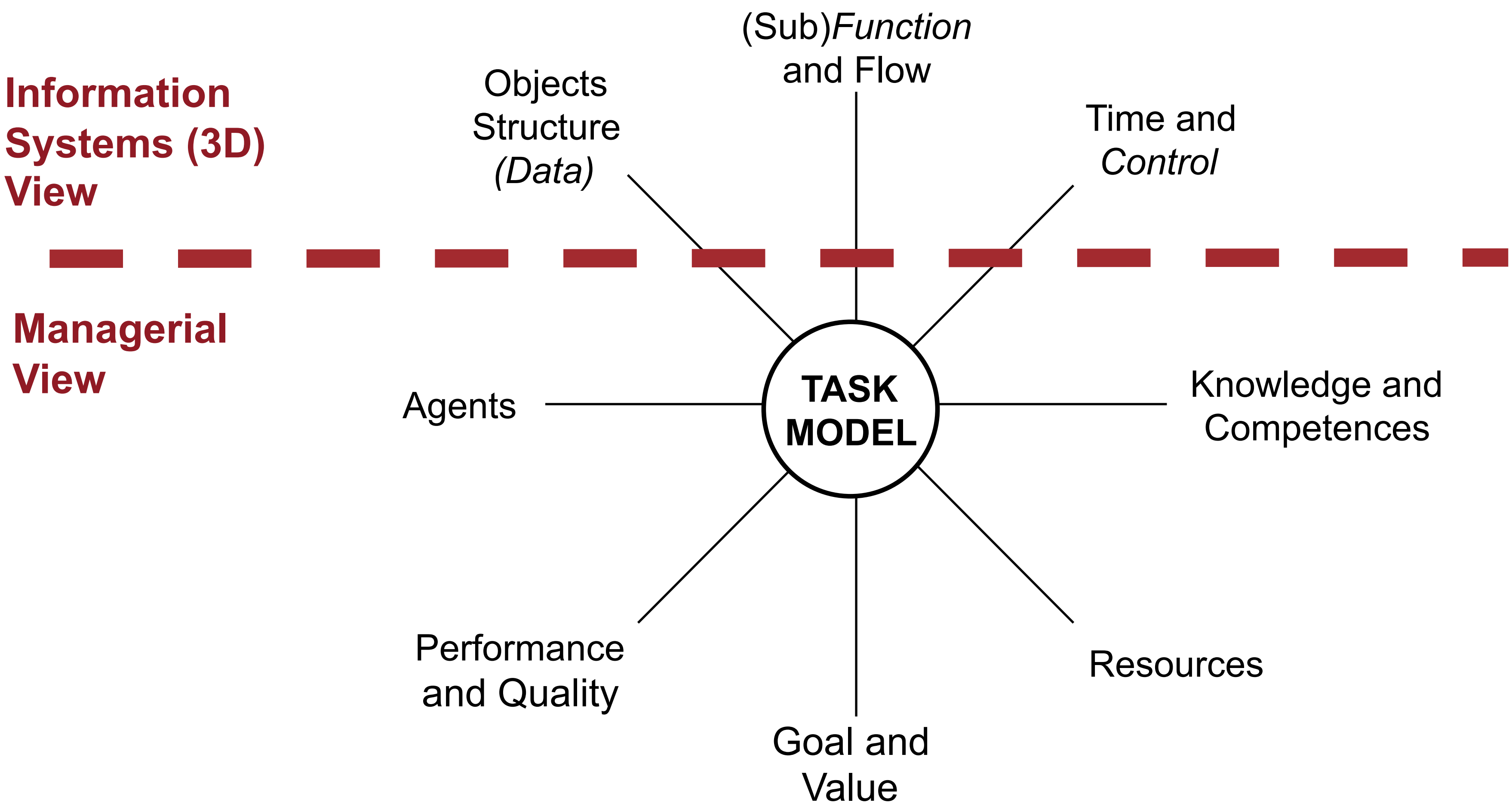
- Knowledge asset:
  - “general residence-application norms”
- right form?
  - no, should be also in electronic form
- right place, time, quality?
  - yes

Task model

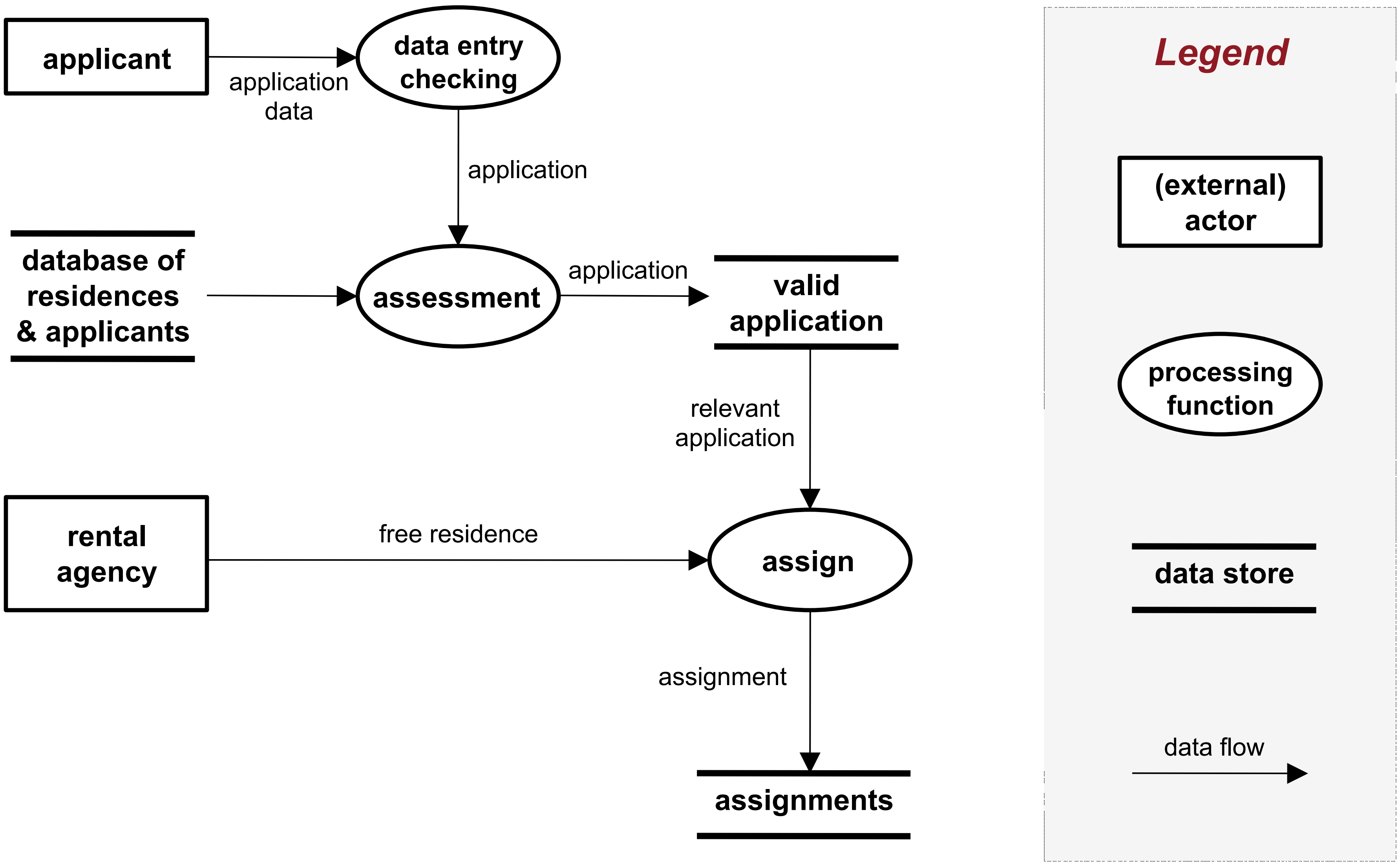
STI · INNSBRUCK

- Task = subpart of a business process
- goal-oriented value-adding activity
- handles inputs and delivers desired outputs
  - in a structured and controlled way
- consumes resources;
- requires (and provides) knowledge/skills
- adheres to quality and performance criteria
- carried out by responsible and accountable agents

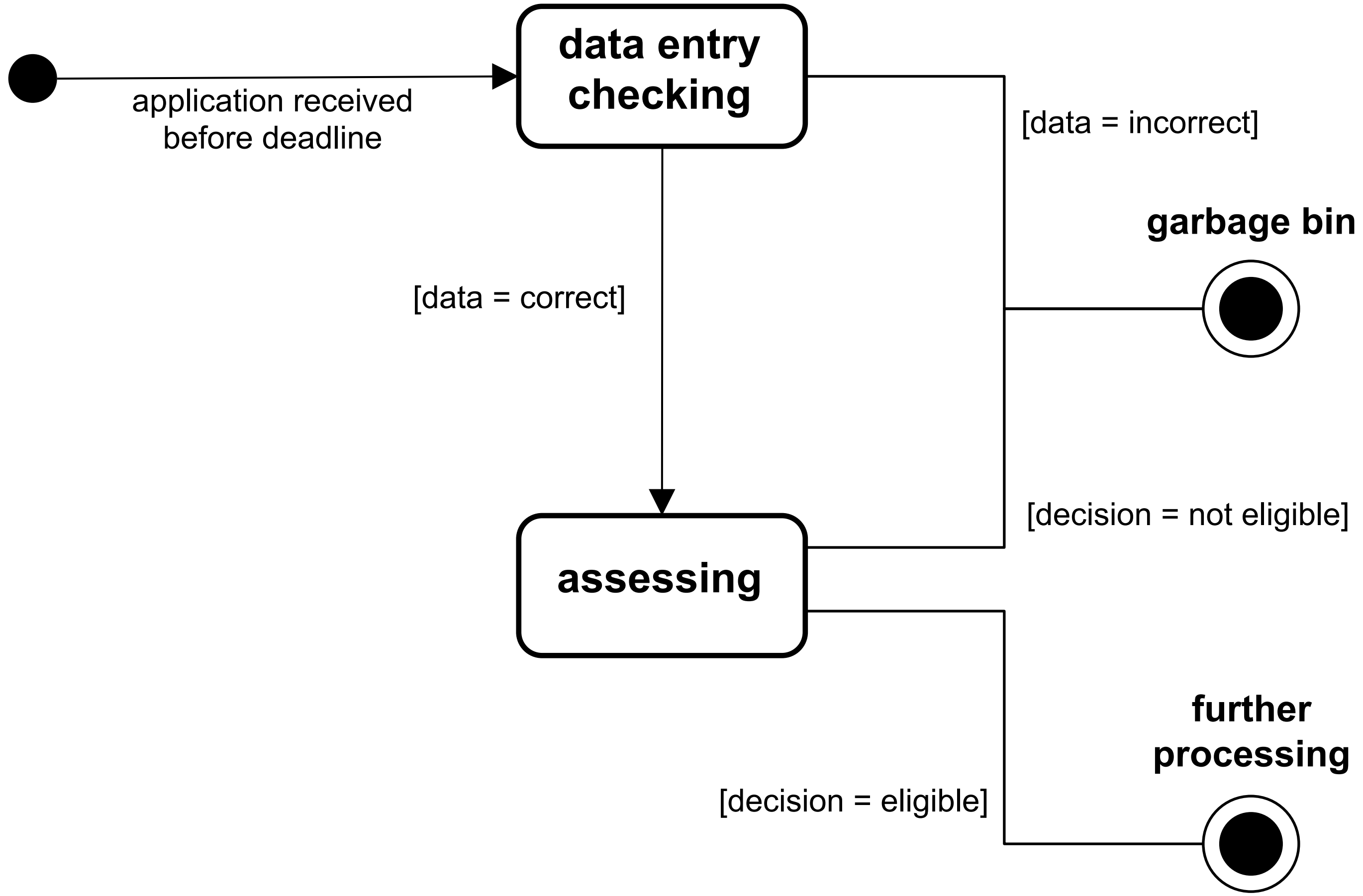
Task model


Information Systems (3D) View
Managerial View
Objects Structure *(Data)*
(Sub)*Function* and Flow
Time and *Control*
Agents
TASK MODEL
Knowledge and Competences
Performance and Quality
Goal and Value
Resources

Task model: data flow



Task model: control flow

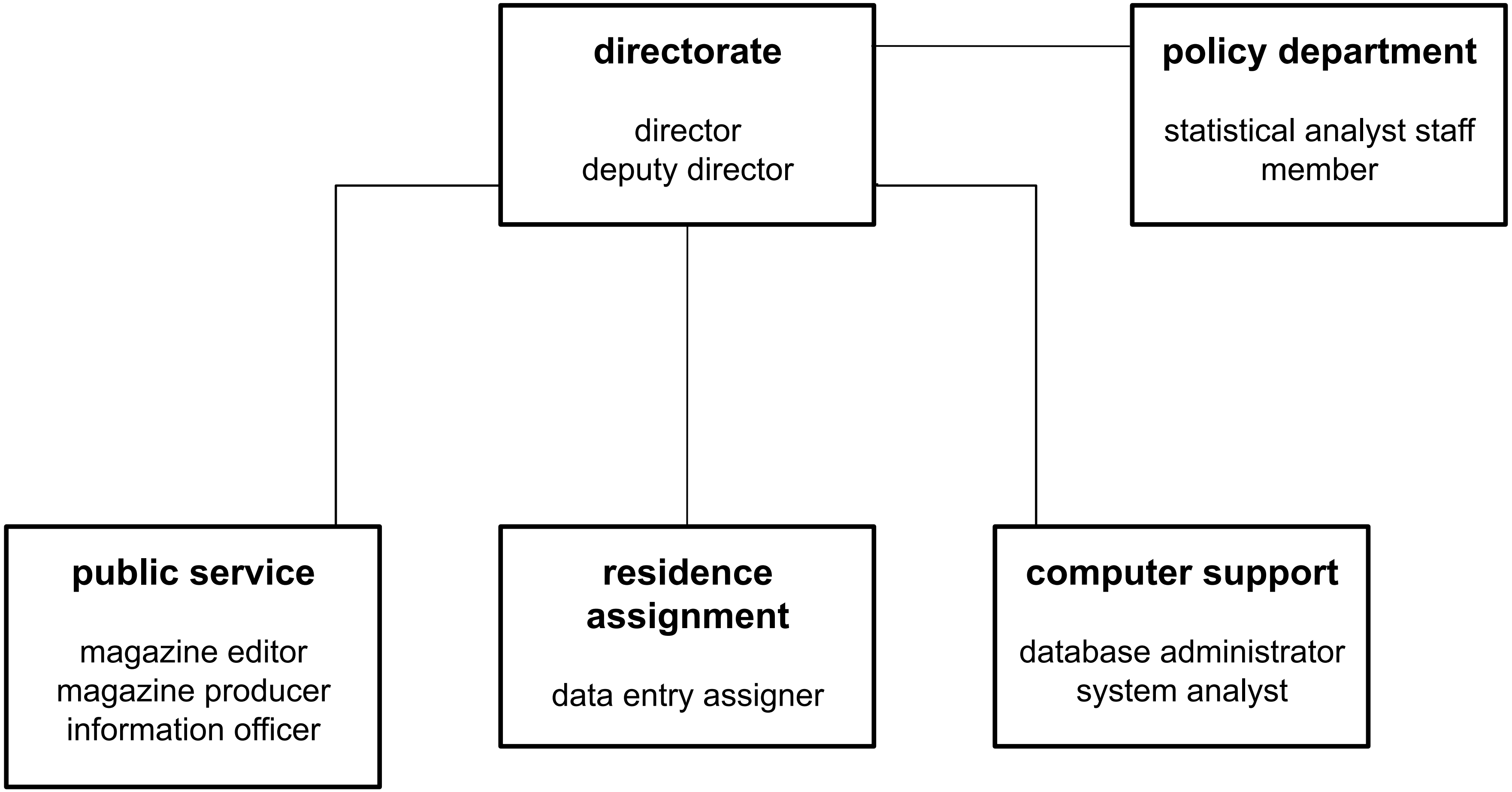


Agent model

STI · INNSBRUCK

- OM and TM => process/task perspective
- AM: perspective of individual agents
  - staff, software systems
- large part: rearrangement of information already in other worksheets
  - just a single worksheet
- agent view useful for judging impact
  - See attitude matrix
- important input for communication model

Agent model


directorate
director
deputy director
policy department
statistical analyst staff member
public service
magazine editor
magazine producer
information officer
residence assignment
data entry assigner
computer support
database administrator
system analyst

Agent model

Name	Assigner
Organization	Residence-assignment department
Involved In	3. Application assessment 4. Residence assignment
Communicates with	Database Priority calculator
Knowledge	Assessment criteria Assignment rules Urgency rules
Other competencies	Ability to handle problematic non-standard cases
Responsibilities & constraints	Make sure that people are treated equally (no favors). This has been a problem in the past

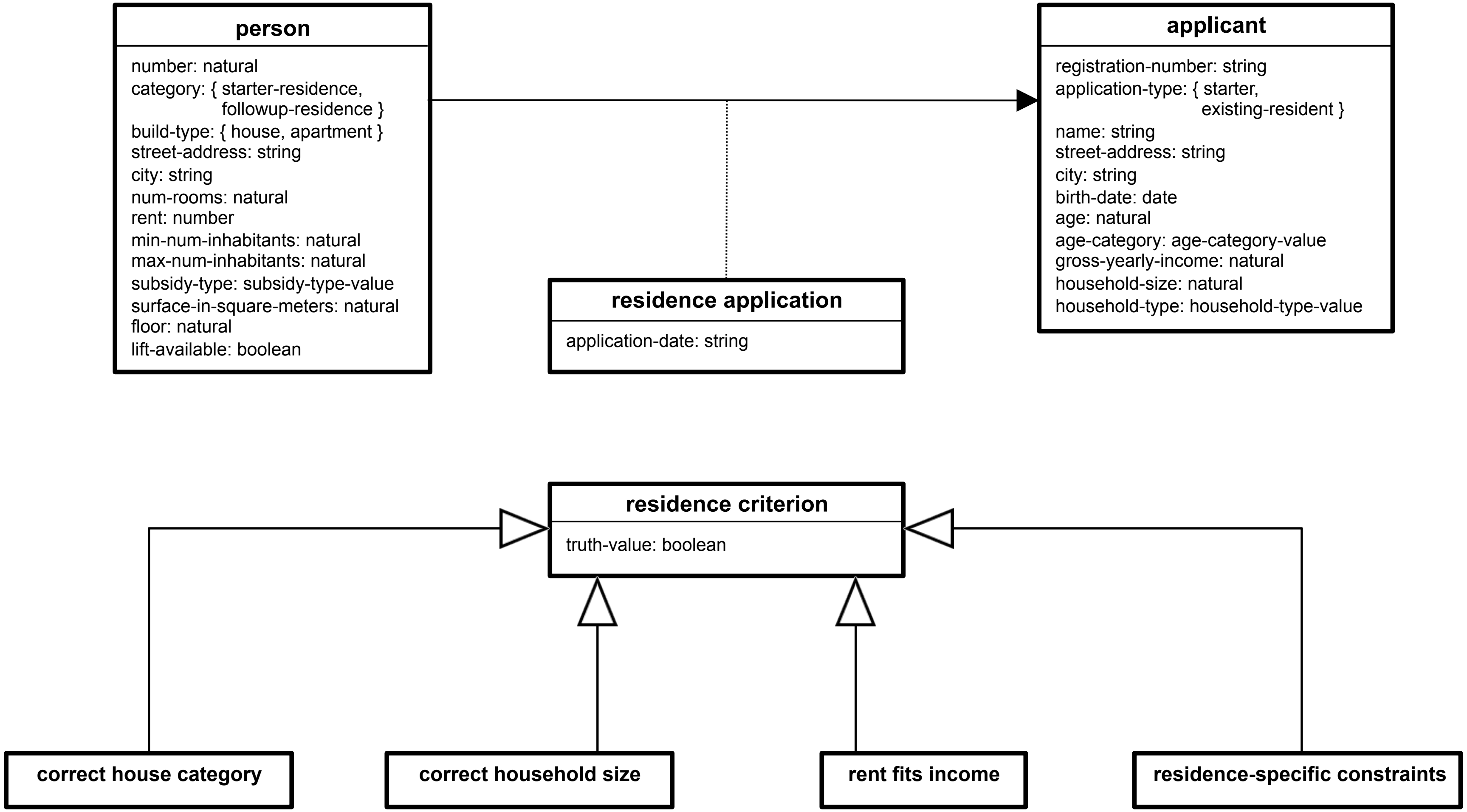
Knowledge model

STI · INNSBRUCK

- Reading the two-weekly magazine in detail
  - organizational goal of transparent procedure makes life easy
- Reading the original report of the local government for setting up the house assignment procedure
  - identification of detailed information about handling urgent cases
- Short interviews/conversations
  - staff member of organization
  - two applicants (the “customers”)
- **Now we look into**
  - **Domain model**
  - **Inference structure**
  - **Task layer**

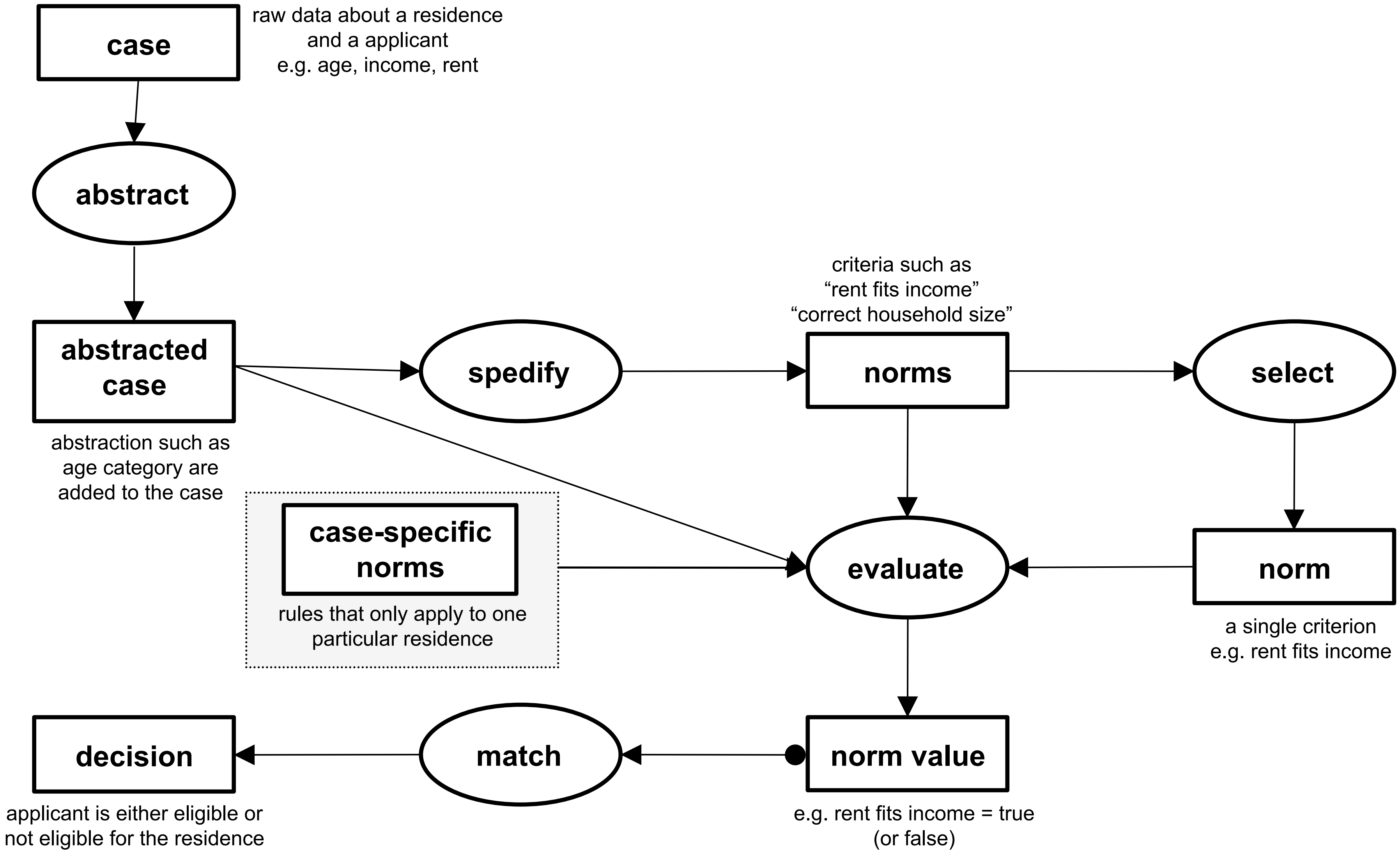
Domain model

STI · INNSBRUCK



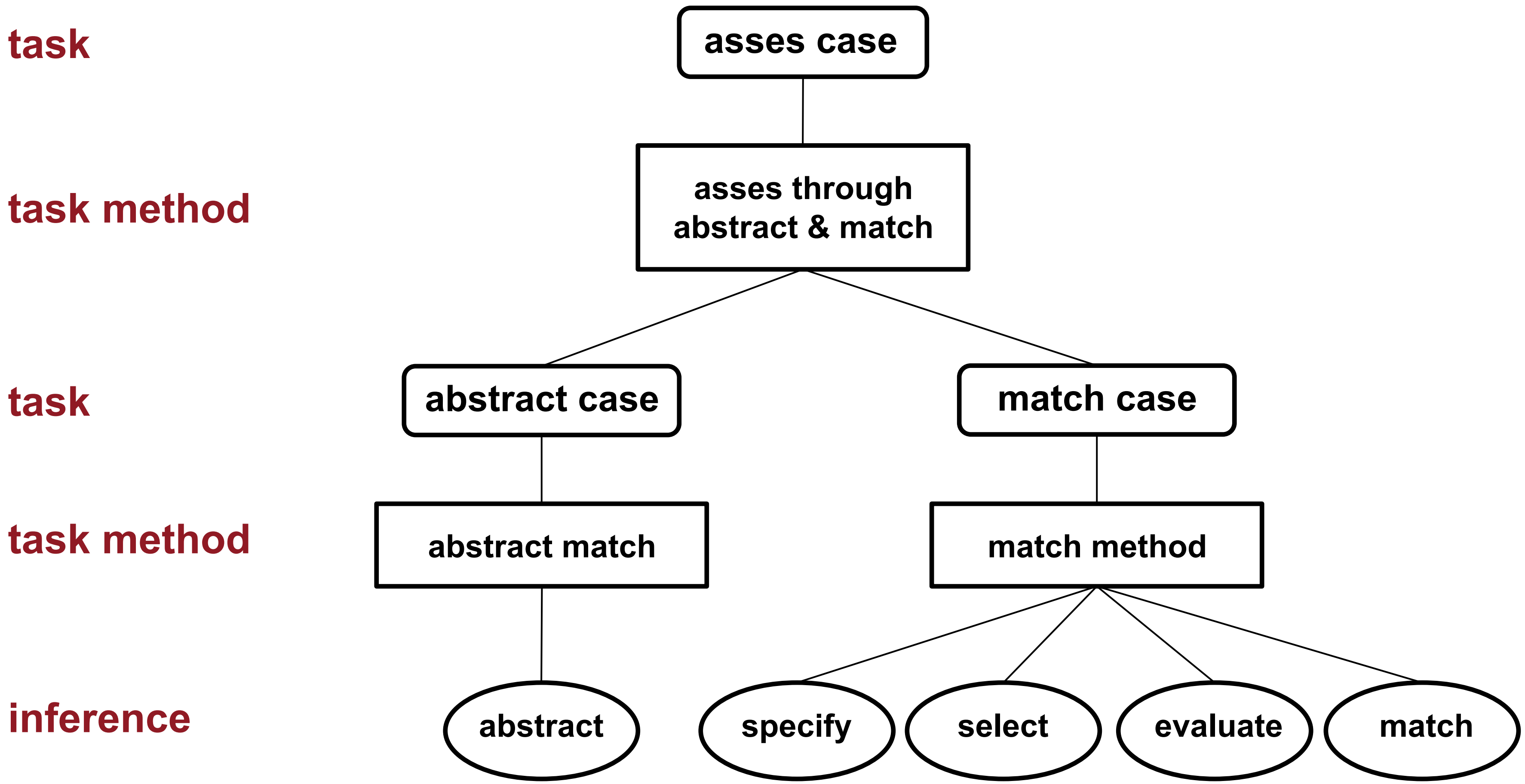
Inference structure

STI · INNSBRUCK



Task layer

STI · INNSBRUCK



SUMMARY

Summary

- Knowledge model components
  - Knowledge model:
    - specialized tool for specification of knowledge-intensive tasks
    - abstracts from communication aspects
    - real-world oriented
    - reuse is central theme
  - Task knowledge
    - goal-oriented
    - functional decomposition
  - Domain knowledge
    - relevant domain knowledge and information
    - static
  - Inference knowledge
    - basic reasoning steps that can be made in the domain knowledge and are applied by tasks
- Template knowledge models
  - Knowledge models partially reused in new applications
  - Type of task = main guide for reuse
  - Catalog of task templates
  - reusable combination of model elements
    - (provisional) inference structure
    - typical control structure
    - typical domain schema from task point-of-view
  - specific for a task type
  - supports top-down knowledge modeling

Summary (cont'd)

	STAGES	TYPICAL ACTIVITIES
Knowledge model construction	**knowledge identification**	▪ **domain familiarization** (information sources, glossary, scenarios) ▪ **list potential model components for reuse** (task- and domain-related components)
	↓	
	knowledge specification	▪ **choose task template** (provides initial task decomposition) ▪ **construct initial domain conceptualization** (main domain information types) ▪ **complete knowledge-model specification** (knowledge model with partial knowledge bases)
	↓	
	knowledge refinement	▪ **validate knowledge model** (paper simulation, prototype of reasoning system) ▪ **knowledge-base refinement** (complete knowledge bases)

Summary (cont'd)

STI · INNSBRUCK

- Knowledge elicitation techniques
  - Interview
  - Self report / protocol analysis
  - Laddering
  - Concept sorting
  - Repertory grids
  - Automated learning techniques
    - Induction

REFERENCES

References

- Mandatory reading:
  - Guus Schreiber, Hans Akkermans, Anjo Anjewierden, Robert de Hoog, Nigel Shadbolt, Walter Van de Velde and Bob Wielinga. *Knowledge Engineering and Management: The CommonKADS Methodology*, MIT Press, ISBN 0262193000. 2000.
    - Chapters 1, 2, 4, 6-8

References

- Further reading:
  - Guus Schreiber, Hans Akkermans, Anjo Anjewierden, Robert de Hoog, Nigel Shadbolt, Walter Van de Velde and Bob Wielinga. *Knowledge Engineering and Management: The CommonKADS Methodology*, MIT Press, ISBN 0262193000. 2000.
  - http://www.commonkads.uva.nl

References

- Wikipedia links:
  - http://en.wikipedia.org/wiki/Knowledge_engineering
  - http://en.wikipedia.org/wiki/Knowledge-based_systems
  - http://en.wikipedia.org/wiki/Knowledge_modeling
  - http://en.wikipedia.org/wiki/Expert_system

Next Lecture

#	Title
1	Introduction
2	Propositional Logic
3	Predicate Logic
4	Reasoning
5	Search Methods
6	CommonKADS
7	**Problem-Solving Methods**
8	Planning
9	Software Agents
10	Rule Learning
11	Inductive Logic Programming
12	Formal Concept Analysis
13	Neural Networks
14	Semantic Web and Services

Questions?