Knowledge in Processes - Introduction to Knowledge-Based Systems
Starting Point: Knowledge and Processes

**process knowledge**

knowledge *about* processes:
- workflow
- participants
- ressources

**functional knowledge**

knowledge *in* processes:
- skills
- domain knowledge
- strategies

- expertise
- product knowledge
- market, competitors
- regulations, laws
- lessons learned
Knowledge in Enterprises

- Tacit knowledge in heads of people
- Self-aware knowledge in heads of people
- Documented knowledge in documents/databases
- Formal knowledge in program code/knowledge base

E = mc²

```
function fac (int x) {
  if x > 0 {
    return x * fac(x-1);
  } else return 1;
}
```

People

Organisation

Information Technology
Application of Knowledge

Examples from the Car Rental Company

- Decision-Making
  - Choose between different offers for new cars

- Diagnosis/Problem Solving
  - Find the failure if the engine of the car does not start

- Configuration
  - Select equipment for new cars

- Planning
  - Scheduling of cars so that they are at the branch

- Information Retrieval
  - Find all documents with regulations about international drivers licences
Knowledge Support of Processes

- Structured processes can contain knowledge work
- Support of Knowledge-Intensive Tasks (KIT) by …
  ... Identifying knowledgable people
    - Assign the task to employee with appropriate skills
  ... Intelligent Information Provision
    - Find documentation
  ... Knowledge-Based System (expert system) for
    - Decision making
    - Planning
    - Diagnosis
    - Problem solving
Expert Systems

„An Expert System is an intelligent computer program that uses knowledge and inference procedures to solve problems that are difficult enough to require human expertise for their solutions.“ (Feigenbaum 1982)

The term „knowledge-based systems“ is often used synonym for „expert systems“. It makes clear that the system has an explicit knowledge base.
Problem Solving: Example

Placing a domino on a chess board

- Given a chess board where two opposite corners are missing
- A domino covers two adjacent field
- Is it possible to cover all fields of the board with dominos?
Possible Solution Approaches

- **Solution 1: Exhaustive Search**
  Check all possibilities to put dominos on the board. Stop when all fields are covered or all possibilities failed.

- **Solution 2: Heuristics**
  Prune the search: Try only promising paths which seem to lead to a (good) solution.

- **Solution 3: Knowledge**
Natural Language Understanding

- Problems with natural Language Understanding:
  - There can be multiple expressions for the same statement
    - Word level: Synonyms
    - Sentence level: different formulations
  - A sentence can have multiple
    - „Time flies like an eagle“
    - „John saw the boy in the park with a telescope“
    - „John saw the boy in the park with a statue“

- Understanding natural language requires knowledge about
  - language (**Syntax** – grammar rules)
  - domain (**Semantics** - knowledge about the meaning of terms)
  - conventions about language use (**Pragmatics**)
    - "Do you know the time?“  →  request to tell the time
Example

Same sentence structure but different interpretations:

*John saw the boy in the park with a telescope.*

*John saw the boy in the park with a statue.*

*John saw the boy in the park with a dog.*
Knowledge Engineering

- Knowledge Engineering is the process of
  - building and
  - maintaining
  knowledge-based systems or intelligent agents

- “Knowledge Engineering is an engineering discipline that involves integrating knowledge into computer systems in order to solve complex problems normally requiring a high level of human expertise.”
  

- Sources of knowledge
  - Human experts
  - Documentation
Knowledge Layers

Knowledge can exist on different layers:

- **Cognitive layer:**
  colloquial statement of thoughts; problems are getting modelled, but still not formalised.

- **Representation layer:**
  Formalisation of thoughts in a representation formalism (e.g. production rules, ontologies)

- **Implementation layer:**
  Formalisation has progressed so much, that the sequence is possible on a computer (e.g. realisation of a CLIPS-programme).
Layers of Knowledge-Based Systems

Credit Check
- Balance sheet
- Experiences
- Competitors
...

Production Rules
IF return > 5%
THEN ...
IF assets < liabilities
THEN ...

Shell CLIPS 6.1:
(defrule no_credit
  (firm (name ?name))
  => (printout t " no credit")
)

Cognitive Layer
Formulation of thoughts
- No formally modelled

Representation Layer
Selection of a representation formalism

Implementation Layer
Selection of a software tool

Prof. Dr. Knut Hinkelmann
Classification of Knowledge

- Derivation
  - explicit
  - implicit

- Reliability
  - precise
  - insecure
  - incomplete
  - vague

- Type
  - declarative
  - procedural

- Representation
  - Meta-Rules
  - Rules
  - Facts
Knowledge in Enterprises

knowledge evolution

<table>
<thead>
<tr>
<th>implicit knowledge</th>
<th>explicit knowledge</th>
</tr>
</thead>
<tbody>
<tr>
<td>tacit knowledge in heads of people</td>
<td>self-aware knowledge in heads of people</td>
</tr>
<tr>
<td>documented knowledge in documents/databases</td>
<td>formal knowledge</td>
</tr>
<tr>
<td>program code knowledge bases</td>
<td></td>
</tr>
</tbody>
</table>

people
organisation
information technology
Reliability of Knowledge

- **Exact knowledge:**
  - „It is raining."

- **Uncertain knowledge:**
  - „I believe it will not rain tomorrow.“

- **Incomplete knowledge (knowledge not complete, but strongly delimited):**
  - „The temperature ist between 10 and 15 degree Celsius“

- **Vague knowledge (interpretation-dependent knowledge):**
  - „The weather is good."
Knowledge-Based Systems

Knowledge Base

- **Facts**
  - Domain knowledge

- **Rules**
  - Problem solving knowledge

Inference Component

- Meta-Rules
- Inference Rules
Types of Knowledge

- Facts: statements about reality
- Rules: General proposition about relations or procedure that are valid under specific conditions (e.g. in an „if ... then“-form“)

Examples:
- Fact:
  - Socrates is human
- Rule:
  - All humans are mortal
Derivation

- Explicit knowledge:
  - knowledge which is filled away in the knowledge base (static knowledge)

- Implicit knowledge:
  - not explicitly stated in the knowledge base
  - is determined from facts by application of rules

Derivation = Inference = Reasoning

- New knowledge is generated from existing one: Making implicit knowledge explicit

Sokrates is human. All humans are mortal.  
\[ \text{Sokrates is mortal.} \]
Meta Rules

Meta Rules …
- implement the Inference
- control the application of rules
- are part of the Inference Engine

Meta Rules can be general, e.g.
- If all conditions of a rule are satisfied then add the conclusion to the knowledge base
- If more than one rule can be applied use the most specific one
- First check whether the

… or domain specific
- For underwriting in health insurance, first apply the rules that deal with the health conditions and then check for the credibility of the applicant
Structural Knowledge

- **Facts:**
  - **Structural knowledge:** Entities and structural relations between them
    - *Classification:* Object A is of class $C_A$
    - *Taxonomy (hierarchy):* Class $C_A$ is more general than Class $C_B$
    - *Aggregation:* Object A consists of parts $A_1$, $A_2$, ...
  - **Relational knowledge:** non-structural Relations and properties

- **Rules:**
  - Knowledge about ways to solve a problem
  - Conditional statements
Structural vs. Operational rules

- Rules can represent structural and operational knowledge
- Remember structural and operational rules in SBVR
  - Structural: True by definition
    - Every Employee is a person
  - Operational: govern what the business does
    - Every Employee who is a manager gets a bonus
Example of a Declarative Knowledge Base

Employee(john_smith)
Employee(mary_baker)
Employee(mark_miller)
Manager(mary_baker)

Employee(X) ⇒ Person(X)
Employee(X) AND Manager(X) ⇒ Bonus(X)

Predicate Logic:
∀x Employee(x) ⇒ Person(x)
∀x Employee(x) ∧ Manager(x) ⇒ Bonus(x)
Example of a Declarative Knowledge Base

Father(peter,mary)
Father(peter,john)
Mother(mary,mark)
Mother(jane,mary)

Father(X,Y) AND Father(Y,Z) → Grandfather(X,Z)
Father(X,Y) AND Mother(Y,Z) → Grandfather(X,Z)
Mother(X,Y) AND Father(Y,Z) → Grandmother(X,Z)
Mother(X,Y) AND Mother(Y,Z) → Grandmother(X,Z)
Father(X,Y) AND Father(X,Z) → Sibling(Y,Z)
Mother(X,Y) AND Mother(X,Z) → Sibling(Y,Z)

The rules can be used to
• Derive all grandparent and sibling relationships (forward chaining)
• Answer questions about relationships (backward chaining)
Forward Chaining vs. Backward Chaining

- **Forward Chaining**
  - Start with the facts and derive all the knowledge that can be derived from that by applying rules
  - Forward Chaining corresponds to Modus Ponens

- **Backward Chaining**
  - Answering Questions
    - Is Peter the Grandfather of Marc?
    - Who are the siblings of Mary?
  - Start with the conclusion of the rules and check conditions
  - Backward Chaining corresponds to Modus Tolens
Declarative vs. Procedural Knowledge

- **Declarative knowledge**: The representation of knowledge is independent of an inference engine.

- **Procedural knowledge**: The representation of knowledge determines its use, e.g. representing actions, updating knowledge.
  
  \[
  \text{if } \text{a car reaches the a traffic light} \\
  \quad \text{and } \text{the traffic light has switched to red} \\
  \quad \text{then hold at the stop line}
  \]

  \[
  \text{if } \text{account balance is X} \\
  \quad \text{and } \text{deposit is Y} \\
  \quad \text{then account balance is X + Y}
  \]
Paradigms of Knowledge Processing

- Symbolic Systems:
  - Logic Systems:
    - Representations: logical formulas
    - Derivation of knowledge: Inference (Deduction)
  - Non-Logic Systems:
    - Representations: condition-action rules
    - Derivation of knowledge: Inference
  - Fuzzy Systems:
    - Representation: linguistic formulated knowledge
    - Derivation of knowledge: Approximate conclusion

- Subsymbolic Systems:
  - Neural Networks
    - Representation: units, weights between units
    - Derivation of knowledge: Connotation
Example: Application for Health Insurance

- Medical Underwriting for Life Insurance is a knowledge-intensive tasks in a structured process of dealing with applications for health insurance.
Automated Underwriting

- Supporting Underwriting has been an application for knowledge based systems for many years
- In the following we look at an example to automate underwriting of Long Term Car and Life Insurance applications
- The system has been in production since December 2002
- In 2005 it completely automated 19.2% of the LTC applications
Example: Underwriting of Insurance Applications

Manual Underwriting Process

- Underwriter Manual
- Knowledge and Experience
- Request Additional Information
- Rate Class Decision (A, B, C, ...)
- Senior Underwriter
- Underwriter
Manual Underwriting Process

- The LTC underwriting process begins when a paper application (APP)
- The APP is scanned into an electronic data warehouse.
- Underwriters located throughout the country view these scanned documents online, and then rate the risk of insuring each person.
- If the underwriter has any concerns, he can request additional information from the applicant via a Phone Health Interview (PHI) and/or a Face-to-Face (F2F) interview, resulting in the submission of additional paper forms.
- An underwriter can also request a copy of the applicant’s medical history from their primary physician (Attending Physician Summary APS).
- An underwriter can make a decision at any point they feel they have sufficient information.
- If they have any questions or concerns, they can refer cases to a senior underwriter.
Decision Making in Underwriting

- Underwriters make decisions following guidelines specified in an underwriter manual.
- They also rely upon extensive medical knowledge and personal experience when underwriting cases.
- Problems:
  - The reliance upon their own experience and judgment causes inconsistency across the underwriters, resulting in inaccurate rate classifications.
Automated Underwriting Process

- Medical Summarizers view applications and fill web-based forms
- FLRE (Fuzzy Logic Rule Engine) = Digital Underwriter
  - Codification of underwriter rules
- Three decisions
  - Rate class of the application
  - Whether or not to order additional information
  - Whether or not to send the case to a human underwriter for review
### Part of an APP Summarization Form

<table>
<thead>
<tr>
<th>Yes</th>
<th>No</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>1. Are you covered by Medicaid (not Medicare)?</td>
<td>Yes</td>
</tr>
<tr>
<td>2. Do you use a Walker or Wheel chair; Oxygen; Respirator; or Kidney Dialysis; or need assistance or supervision by another person in performing any of the following: Moving in/out of bed or chair; Bathing; Dressing; Eating; Toileting; Bowel/Bladder control; Walking?</td>
<td>Yes</td>
</tr>
<tr>
<td>3. Have you had, do you currently have, or have you ever been medically diagnosed as having any of the following:</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Acquired Immune Deficiency Syndrome (AIDS)</td>
</tr>
<tr>
<td></td>
<td>Emphysema/COPD in combination with any of the following:</td>
</tr>
<tr>
<td></td>
<td>current smoking, Congestive Heart Failure (CHF), Asthma, or Chronic Bronchitis</td>
</tr>
<tr>
<td></td>
<td>AIDS Related Complex (ARC)</td>
</tr>
<tr>
<td></td>
<td>Frequent or persistent Forgetfulness</td>
</tr>
<tr>
<td></td>
<td>ALS (Lou Gehrig's Disease)</td>
</tr>
<tr>
<td></td>
<td>Memory Loss</td>
</tr>
<tr>
<td></td>
<td>Alzheimer's Disease</td>
</tr>
<tr>
<td></td>
<td>Metastatic Cancer (Spread from original site/location)</td>
</tr>
<tr>
<td></td>
<td>Congestive Heart Failure (CHF) in combination with any of the following: Heart Attack or Angina;</td>
</tr>
<tr>
<td></td>
<td>Multiple Sclerosis (MS)</td>
</tr>
<tr>
<td></td>
<td>Transient Ischemic Attack (TIA) within the past 5 years</td>
</tr>
<tr>
<td></td>
<td>TIA in combination with Diabetes or Heart Surgery</td>
</tr>
</tbody>
</table>
Challenges for Automating Underwriting

- Use of personal knowledge and experience to make decisions made automation a difficult problem
  - Knowledge elicitation
- Input to the process (application, attending physician summary APP, summaries) contained free text
  - Natural Language Understanding
Relative Frequency of Impairments

- Hypertension: 35.97%
- Osteoarthritis: 17.70%
- Diabetes: 8.26%
- Depression: 7.33%
- Bone and Joint Disorder: 6.02%
- Gastro-Intestinal Disorder: 5.83%
- Dysrhythmia: 5.39%
- Osteoporosis: 5.07%
- Prostate Disorder: 4.92%
- Memory Loss/Forget/Confusion: 3.51%
Underwriter Assist Screen

### Applicant Information

<table>
<thead>
<tr>
<th>Policy Number</th>
<th>PI/SP:</th>
<th>SP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name:</td>
<td>Age:</td>
<td>47</td>
</tr>
</tbody>
</table>

### Application Type:

- Preferred

### Employment Status:

- Does Not Work
- Smoking Status: Non-Smoker

### App Height:

- 5 ft. 10 in.

### PHI Height:

- NA

### MRR Height:

- NA

### Engine Results Summary

<table>
<thead>
<tr>
<th>Date/Time</th>
<th>Engine</th>
<th>Recommendation</th>
<th>Routing</th>
<th>Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>03/22/2004 12:20:54</td>
<td>APP</td>
<td>PREFERRED</td>
<td>UW</td>
<td>NA</td>
</tr>
<tr>
<td>0</td>
<td>PHI</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>0</td>
<td>HTN</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>0</td>
<td>DM</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>03/23/2004 05:23:05</td>
<td>OA</td>
<td>STANDARD</td>
<td>UW</td>
<td>NA</td>
</tr>
<tr>
<td>0</td>
<td>OP</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>0</td>
<td>GENERAL</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>03/23/2004 10:12:02</td>
<td>Finals</td>
<td>STANDARD</td>
<td>UW</td>
<td>NA</td>
</tr>
</tbody>
</table>

Recommendations made by different engines (one line per FLRE instance)
### Underwriter Assist Screen

Details about rules that caused the rate class recommendation

#### APP

<table>
<thead>
<tr>
<th>Underwriting Reason</th>
<th>Value</th>
<th>English Rule</th>
<th>Guideline</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prescription_1</td>
<td>NA</td>
<td>Applicant takes a prescription</td>
<td>PDF</td>
<td>pg 2</td>
</tr>
<tr>
<td>Speciality_Not_Stated</td>
<td>NA</td>
<td>Speciality not stated</td>
<td>PDF</td>
<td>pg 10</td>
</tr>
<tr>
<td>Other_Dr_Reason_Visit_1</td>
<td>NA</td>
<td>Unknown reason for a doctor’s visit</td>
<td>PDF</td>
<td>pg 1</td>
</tr>
</tbody>
</table>

#### Osteoarthritis

<table>
<thead>
<tr>
<th>Rate Class Reason</th>
<th>Value</th>
<th>English Rule</th>
<th>Guideline</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prescription_Use</td>
<td>NA</td>
<td>Applicant takes a non-narcotic prescription for OA</td>
<td>PDF</td>
<td>pg 7</td>
</tr>
<tr>
<td>Joint_Replacement_Discussed</td>
<td>NA</td>
<td>Doctor discussed joint replacement surgery with applicant</td>
<td>PDF</td>
<td>pg 7</td>
</tr>
<tr>
<td>COX2_Use</td>
<td>NA</td>
<td>Applicant takes a COX2 inhibitor</td>
<td>PDF</td>
<td>pg 8</td>
</tr>
</tbody>
</table>
Fuzzy Logic Rule for Classification

- Fuzzy logic rules are used to encode underwriting standards.
- Fuzzy logic is a superset of conventional Boolean logic for vague knowledge:
  - Instead of having only two values (true/false) it can express intermediate truth values.
- Each rule represents fuzzy constraints at the boundaries between different rate classes for each input, such as cholesterol, blood pressure, or body-mass index.

Below 250 Cholesterol is good (1)

Cholesterol higher than 270 is bad (0)
Objective: determine if the text entered by the summarizers is benign
  - If text entries could be interpreted and classified as benign, the level of automation could be increased.

A grammar was constructed for benign text and lists were created for:
  - Noise words and in-phrase characters (Noise)
  - Phrase separators (Separator)
  - Benign words or synonyms (Benign)
  - Dates in various formats (Date)

Statistical Approach: Learn benign words and phrases

The current grammar for benign text is:
  - **BenignText:**
    - BenignPhrase [Separator [BenignPhrase]]*
  - **BenignPhrase:**
    - [Noise]* [Benign [Noise]* [Date [Noise]*]
Are Machines Able to Think? – The Turing-Test

- Are Machines able to think?
- In order to find an answer to this question, the English computer pioneer A. Turing developed 1950 the so-called Turing-Test
- Test arrangement:
  - Room A: interviewer
  - Room B: Computer and Human
- The interviewer asks questions from different fields aiming to discover whether the computer or the human has provided the answer.
- The computer has passed the Turing-Test, if the interviewer cannot say who answers the questions, the computer or the human.