Modern Rule-Based Systems

Future Trends in Knowledge Management and Decision Making in Autonomous Systems

Antoni Ligęza

Katedra Automatyki
Wydział Elektrotechniki, Automatyki, Informatyki i InzynieriiBiomedycznej
Akademia Górniczo-Hutnicza
Kraków

Presentation - Kraków 2012

AGH

Table of contents

AGH

Wykaz literatury

1. Stuart J. Russel, Peter Norvig: *Artificial Intelligence. A Modern Approach.* Second Edition. Prentice Hall, New Jersey, 2010.
2. Jay Liebowitz: *The Handbook of Applied Expert Systems.* CRC Press, Boca Raton, 1998.
3. Frank van Harmelen, Vladimir Lifschitz, Bruce Porter (Eds.): *Handbook of Knowledge Representation*. Elsevier B.V., Amsterdam, 2008.
4. Michael Negnevitsky: *Artificial Intelligence. A Guide to Intelligent Systems.* Addison-Wesley, Pearson Education Limited, Harlow, England, 2002.
5. Adrian A. Hopgood: *Intelligent Systems for Engineers and Scientists.* CRC Press, Boca Raton, 2001.
6. Joseph C. Giarratano, Gary D. Riley: *Expert Systems. Principles and Programming.* Fourth Edition, Thomson Course Technology, 2005.

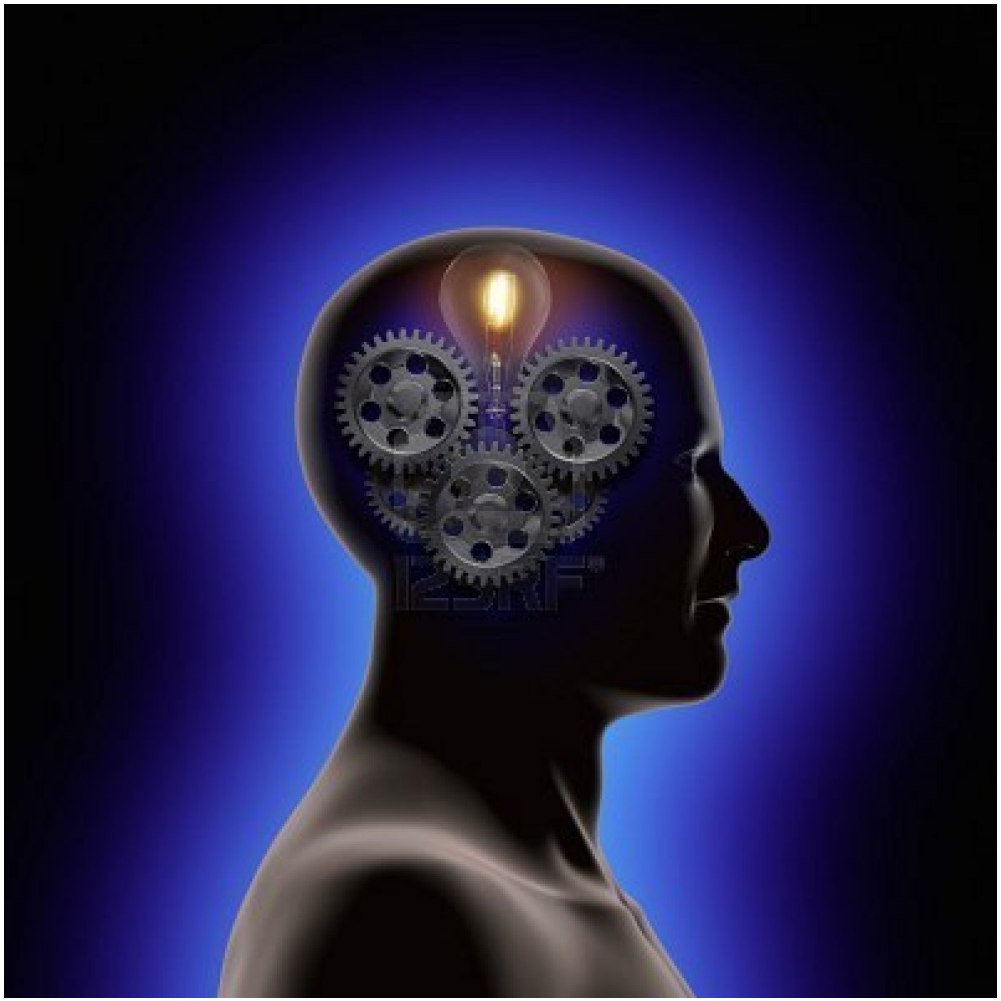
AGH

Wykaz literatury – c.d.

1. Peter Jackson: *Introduction to Expert Systems.* Addison-Wesley, Harlow, England, 1999.
2. Mordechai Ben-Ari: *Mathematical Logic for Computer Science.* Springer-Verlag, London, 2001.
3. Antoni Ligęza: *Logical Foundations for Rule-Based Systems.* Springer-Verlag, berlin, 2006.
4. Michael R. Genesereth, Nils J. Nilsson: *Logical Foundations of Artificial Intelligence*. Morgan Kaufmann Publishers, Inc., Los Altos, California, 1987.
5. Zbigniew Huzar: *Elementy logiki dla informatyków*. Oficyna Wyawnicza Politechniki Wrocławskiej, Wrocław, 2007.
6. Paweł Cichosz: *Systemy uczące się.* WNT, Warszawa, 2000.
7. Jan J. Mulawka: *Systemy ekspertowe.* WNT, Warszawa, 1996.

AGH

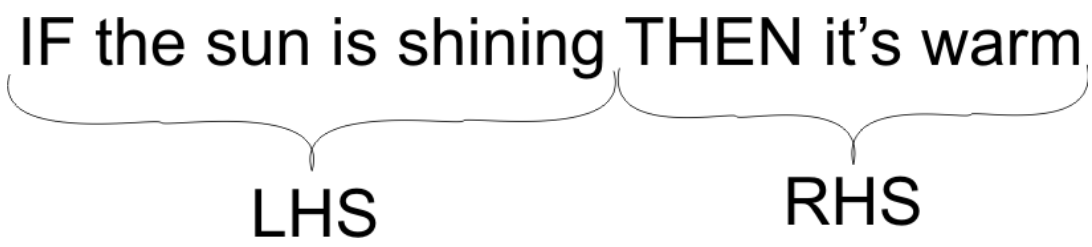
Thinking — What is the Essence of it?



Thinking: Planning, Acting, Monitoring, Reacting, Replanning

Rules and Rule–Based Systems

- Rules – very popular and powerful method for knowledge representation.
  - Usually presented in the **IF...THEN...** form.



- Rule–based systems – a class of expert/control systems.

Rule 1 : IF s is NB THEN u_f is bigger

Rule 2 : IF s is NM THEN u_f is big

Rule 3 : IF s is Z THEN u_f is medium

Rule 4 : IF s is PM THEN u_f is small

Rule 5 : IF s is PB THEN u_f is smaller

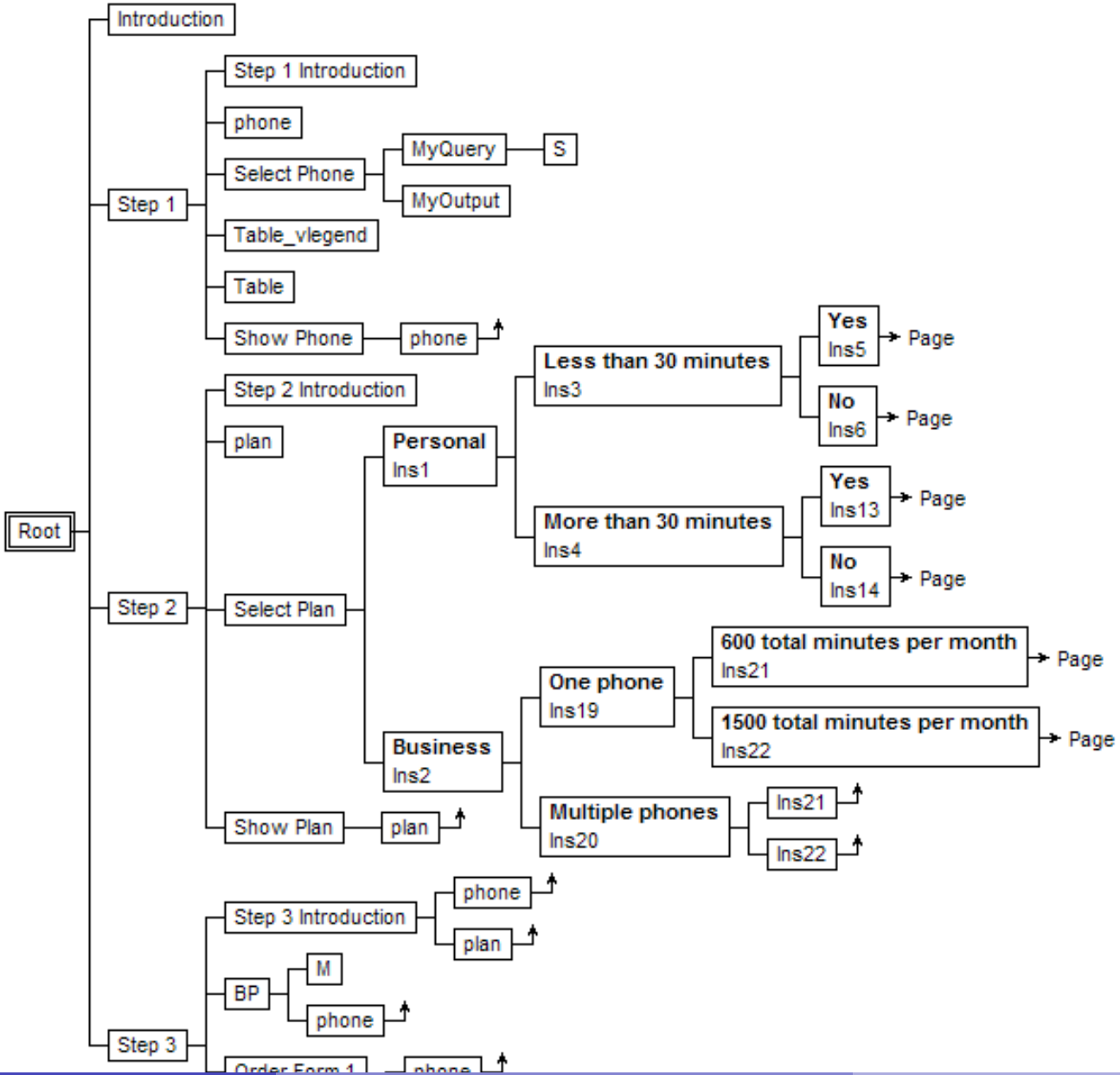
AGH

Rules: Decision Tables

Rules void **defineGreeting**(App app, int hour)		
C1	C2	A1
min <= hour	hour <= max	app.greeting = greeting;
int min	int max	String greeting
Hour From	**Hour To**	**Set Greeting**
0	11	Good Morning
12	17	Good Afternoon
18	22	Good Evening
23	24	Good Night

Rules void **defineSalutation**(App app, Customer c)			
Gender	**Marital Status**	**Age Less Than**	**Set Salutation**
Female	Married		Mrs.
Female	Single		Ms.
Male			Mr.
Male	Single	10	Little

Rules: Decision Trees



An Idea of Rule-Based System

Fact Base

$$\begin{aligned} fact_1 &: \#q_1 \\ fact_2 &: \#q_2 \\ &\vdots \\ fact_k &: \#q_k \end{aligned}$$

where: $\#$ stays for negation ($\neg$) or nothing; $\#p$ is a literal.

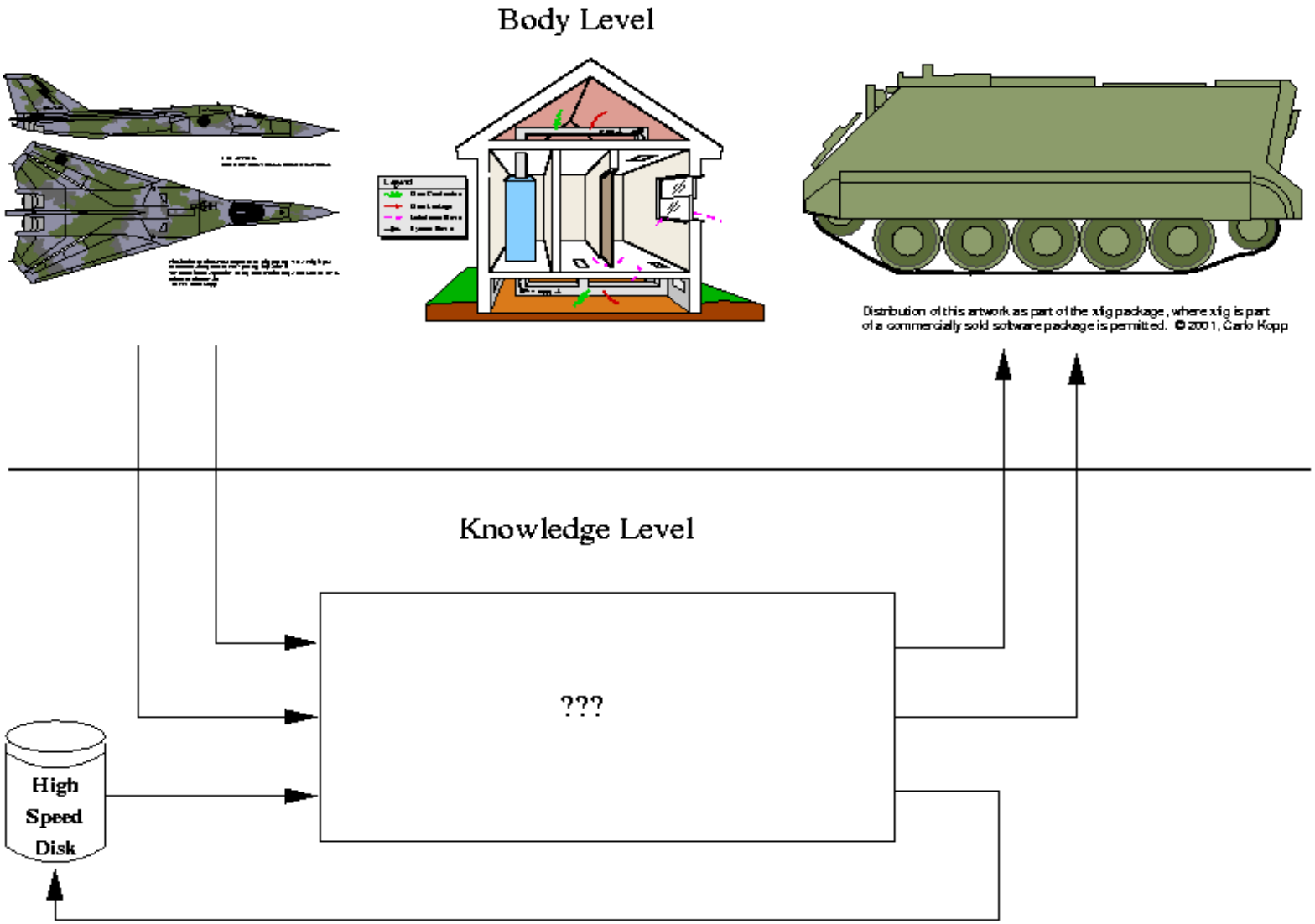
Rule-Base

$$\begin{aligned} rule_1 &: \#p_1 \wedge \#p_2 \wedge \ldots \wedge \#p_n \longrightarrow \#h_1 \\ rule_2 &: \#p_1 \wedge \#p_2 \wedge \ldots \wedge \#p_n \longrightarrow \#h_2 \\ &\vdots \\ rule_m &: \#p_1 \wedge \#p_2 \wedge \ldots \wedge \#p_n \longrightarrow \#h_m \end{aligned}$$

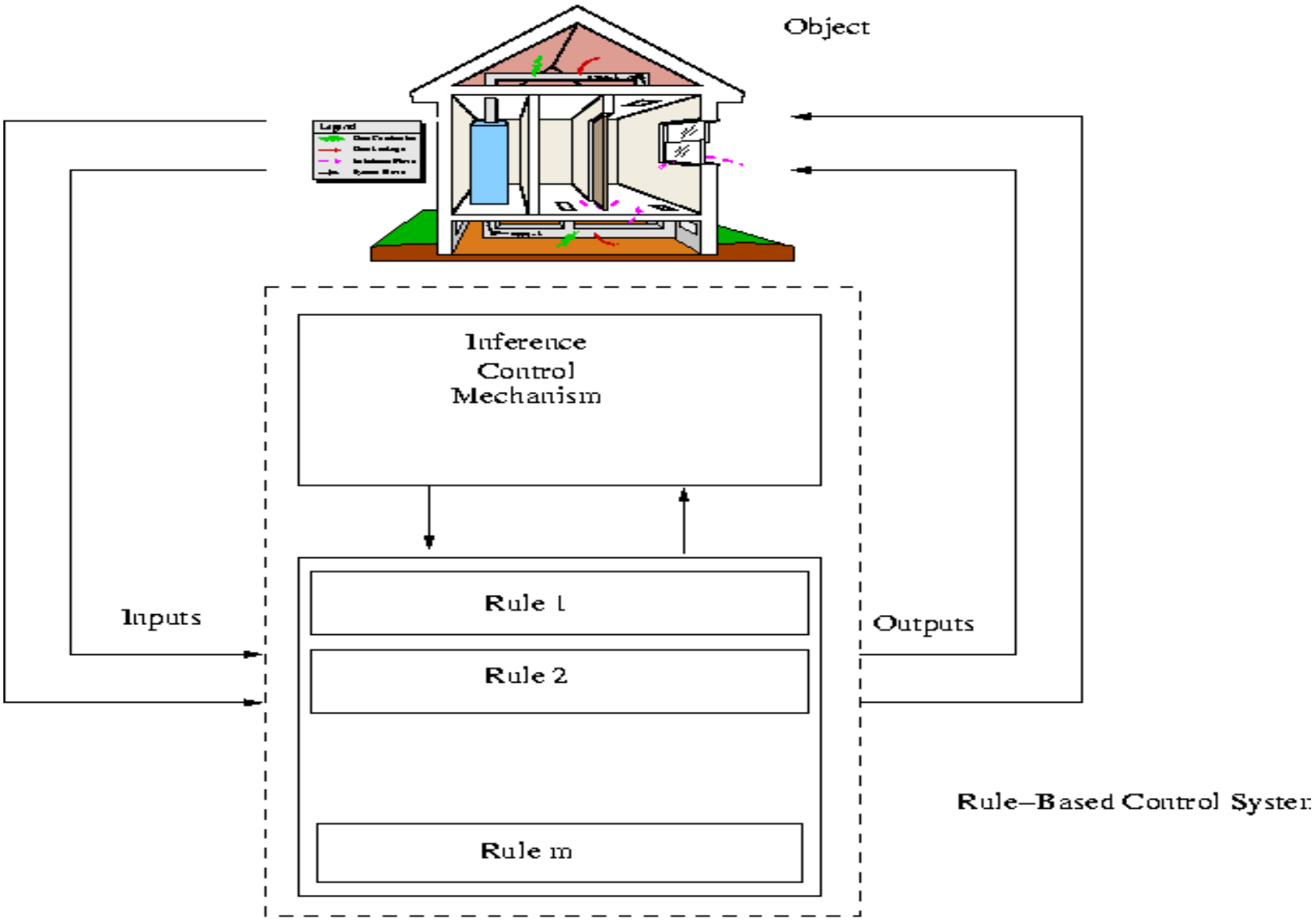
where: $\#$ stays for negation ($\neg$) or nothing; $\#p$ is a literal.

AGH

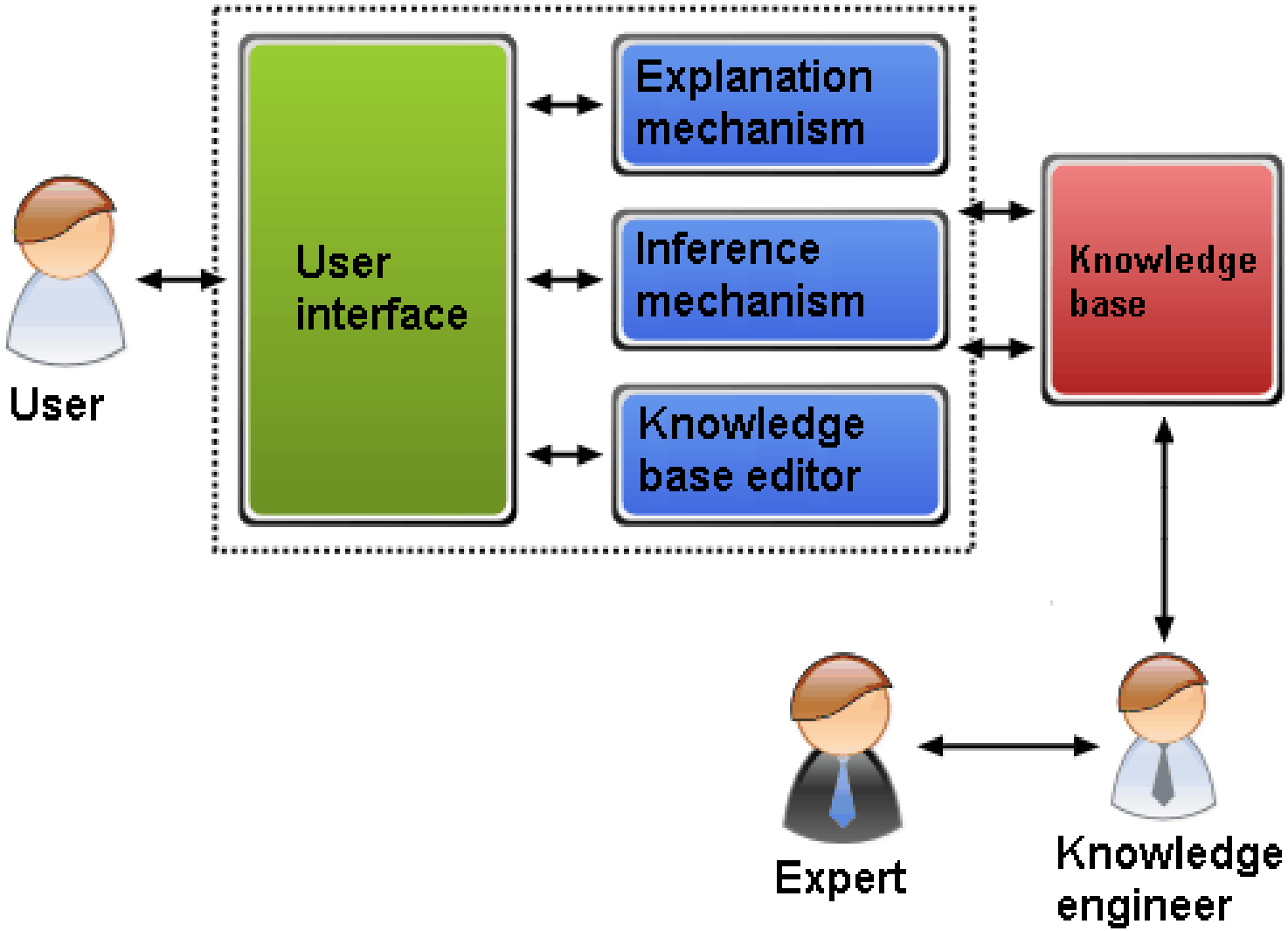
Structure: Basic components of an Expert System Shell



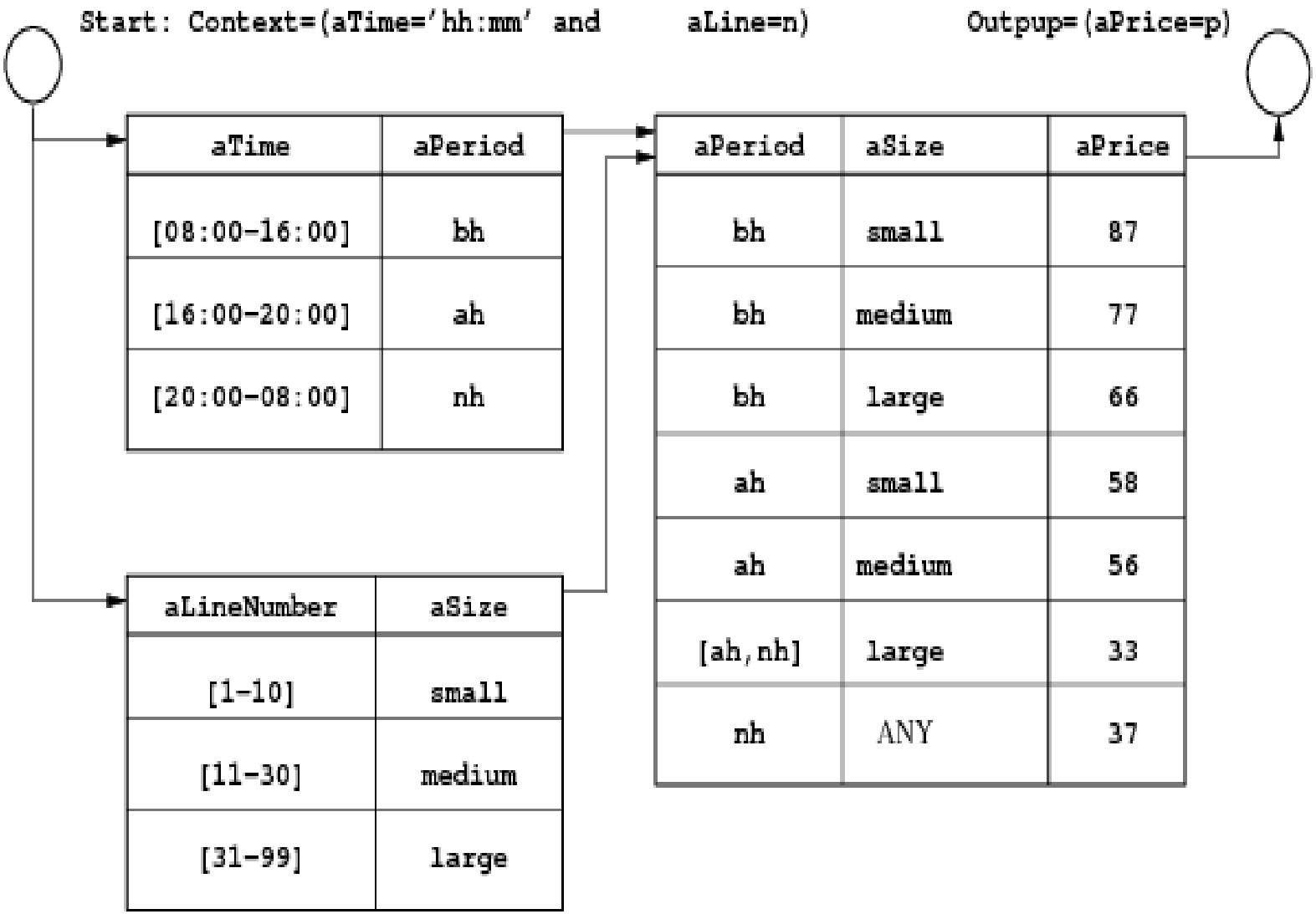
Structure: Basic components of an Expert System Shell



Structure: Basic components of an Expert System Shell



Example structure and rules



aTime	aPeriod
[08:00-16:00]	bh
[16:00-20:00]	ah
[20:00-08:00]	nh

aLineNumber	aSize
[1-10]	small
[11-30]	medium
[31-99]	large

aPeriod	aSize	aPrice
bh	small	87
bh	medium	77
bh	large	66
ah	small	58
ah	medium	56
[ah,nh]	large	33
nh	ANY	37

Rule-Based Systems — Concepts, Practice, Tools

Selected Application Areas

- society: co-exitence, ethics, code of honor,
- law: orders and prohibitions, taxes,
- organizations: army, banks, companies,
- education and universities,
- economy and business: (BI – Business Intelligence, BR – Business Rules),
- medicine: medical procedures,
- mathematics, physics, chemistry, logics, biology,
- technology, management, industry:
  - monitoring,
  - control,
  - diagnostics,
  - service, repairs,
  - decision support,
  - design and analysis,
  - verification, testing, validation,
  - optimization, adaptation, tuning,
  - configuration.

Knowledge Engineering: Status of Rule-Based Systems

Concepts and Foundations: Matured Solutions

- programming languages (e.g.Lisp, Prolog, procedural ones),
- knowledge representation formalisms (attribute logic, FOL),
- formal logic,
- inference,
- inference control.

AGH

Knowledge Engineering: Status of Rule-Based Systems

Concepts and Foundations: Matured Solutions

- programming languages (e.g.Lisp, Prolog, procedural ones),
- knowledge representation formalisms (attribute logic, FOL),
- formal logic,
- inference,
- inference control.

Rule-Based Systems: State-of-the-Art

- matured technology,
- numerous tools — shells,
- numerous successful applications,
- stable *market,*
- *textbook, courses, seminars; research — reincarnation*
- *embedded solutions, hidden components,*
- *negative lessons learned.*

Knowledge Engineering: Status of Rule-Based Systems

Rule-Based Systems: Position Statement

- rules are some most successful knowledge representation formalism,
- number of engineering and business applications (hidden or explicit),
- expressive power: high-level declarative knowledge representation,
- *logical independence of applications* — rules are 'data',
- logical model (in background),
- often combined with other tools,
- visual edition tools (sometimes).

AGH

Key Factors for Rule-Based System Success

9 components

- Knowledge representation,
- Inference rules (legal mooves),
- Inference control (arrive at the goal; avoid combinatorial explosion),
- Knowledge acqusition:
  - Design,
  - Implementation,
  - Input of Knowledge,
- User interface (picture!), communiaction, explanations,
- Verification and Validation,
- Modification, extention, adaptataion, learning,
- Abstraction, generalization,
- Automated approach.

AGH

Knowledge Representation Techniques

Variety of KR Tools

- numeric (numbers, vectors, matrices, functions, equations),
- qualitative (intervals, symbolic, $\{-, 0, +\}$,
- algebraic (sets, relations, structures),
- logical formalisms (facts, formulas, rules),
  - propositionl logic,
  - attribute logic,
  - Datalog, Prolog, FOL,
  - Description Logic,
- **rule-based systems, rules**,
- graphs, semantic networks,
- frames, structural (objects),
- pictures (diagrams, schemes, blocks),
- combined (e.g. XTT).

AGH

Inference Techniques

Patterns of inference

- Abstraction (generalization),
- Specjalization,
- Pattern Matching,
- Case-Based Reasoning, analogy,
- Logical inference:
  - deduction,
  - abduction,
  - induction,
- **Rule-Based Inference (forward, backward, top-down)**,
- Search algorithms,
- Problem reduction (AND-OR graph search),
- Constraint Satisfaction Techniques,
- Consistency-Based Reasoning,
- Graph Transformations, Graph Grammars,
- Numerical Procedures (e.g. optimization),

Inference Control Techniques

Efficient Inference Factors

- KR formalism; search-space selection,
- systematic, blind inference/search,
- heuristic inference/search,
- rule selection, conflict resolution,
- meta-rules, inference control rules,
- problem decomposition, structuring, ordering,
- inference planning,
- using constraints,
- problem reduction,
- constraint relaxation,
- mini-max strategies,
- elimination of cases (tabu search).

AGH

Example of Jess/Clips

```
(bind ?*workdays* (create$ Monday Tuesday Wednesday Thursday Friday))

(bind ?*weekend-days* (create$ Saturday Sunday))

; Rule 1

(defrule workday-if-M-F
 (declare (salience 100))
 (day ?d&:(member$ ?d ?*workdays*))
 ?dow <- (day ?d)
 =>
 (assert (workday))
 (retract ?dow)
 (bind ?*WEEKDAY* ?d)
 (bind ?*WEEKEND* False)
)

(defrule workday-if-M-F2
 (declare (salience 100))
 (day ?d&:(member$ ?d ?*workdays*))
 ?weekend <- (weekend)
 ?dow <- (day ?d)
 =>
 (assert (workday))
 (retract ?weekend)
 (retract ?dow)

; Rule 2

(defrule wekeend-if-S-S
 (declare (salience 100))
 (day ?d&:(member$ ?d ?*weekend-days
 ?dow <- (day ?d)
 =>
 (assert (weekend))
 (retract ?dow)
 (bind ?*WEEKDAY* ?d)
 (bind ?*WEEKEND* True)
)

(defrule wekeend-if-S-S2
 (declare (salience 100))
 (day ?d&:(member$ ?d ?*weekend-days
 ?workday <- (workday)
 ?dow <- (day ?d)
 =>
 (assert (weekend))
 (retract ?workday)
 (retract ?dow)
 (bind ?*WEEKDAY* ?d)
 (bind ?*WEEKEND* True)
```

Example of Drools Code

```
rule "Rule 1"
when
//$wd : WorkDays()
    //$ttd : ThermostatTimeData( day memberOf $wd.daylist )
    $ttd: ThermostatTimeData ( day == "Monday" || day == "Tuesday" || day
then
//insertLogical( new WorkDay() );
modify ( $ttd ) {
 setDaytype("workday")
}
end

rule "Rule 2"
 when
$ttd: ThermostatTimeData ( day == "Saturday" || day == "Sunday", daytype =
 then
modify ( $ttd ) {
setDaytype("weekend")
}
end
```

Example decision table

EU-Rent discounts for car rental

Car Group	Compact						Mid-size, Full-size												Lux, SUV, Van						Other	
Rental period	D, W, M				Other		D				W				M				Other		D		W		M		-	
New club member	Y		N		-		Y		N		Y		N		Y		N		-		-		-		-		-	
>3 days in advance	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N
10%	x		x		x		x		x		x		x		x		x		x		x		x		x		x	
€ 50											x	x	x	x								x	x					
Max (10%, € 50)											x		x										x					
2-group upgrade	x	x					x	x			x	x			x	x												
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28

Decision tables are used by several commercial rule engines

Example from "Decision Tables and Business Rules", EBRC 2004 Tutorial, Jan Vanthienen

Attribute Logic: ALSV(FD)

A_i — an attribute V_i — a subset of D_i; $d \in D_i$ — a single element.

Simple Attributes

The legal atomic formulae of ALSV for simple attributes are:

- $A_i = d$,
- $A_i \neq d$,
- $A_i \in V_i$,
- $A_i \notin V_i$.

Generalized Attributes

The legal atomic formulae of ALSV for generalized attributes are:

- $A_i = V_i$,
- $A_i \neq V_i$,
- $A_i \subseteq V_i$,
- $A_i \supseteq V_i$,
- $A \sim V$,
- $A_i \not\sim V_i$.

XTT Table

Rule	A_1	A_2	...	A_n	H
1	$\propto_{11}$ t_{11}	$\propto_{12}$ t_{12}	...	$\propto_{1n}$ t_{1n}	h_1
2	$\propto_{21}$ t_{21}	$\propto_{22}$ t_{22}	...	$\propto_{2n}$ t_{2n}	h_2
⋮	⋮	⋮	⋱	⋮	⋮
m	$\propto_{m1}$ t_{m1}	$\propto_{m2}$ t_{m2}	...	$\propto_{mn}$ t_{mn}	h_m

AGH

Example: Thermostat Table

Info	Prec		Retract	Assert	Decision	Ctrl	
I	*aSE*	*aOP*			*aTHS*	*N*	*E*
11	*spr*	*dbh*			20	1.1	4.12
12	*spr*	*ndbh*			15	1.1	4.13
13	*sum*	*dbh*			24	1.1	4.14
14	*sum*	*ndbh*			17	1.1	4.15
15	*aut*	*dbh*			20	1.1	4.16
16	*aut*	*ndbh*			16	1.1	4.17
17	*win*	*dbh*			18	1.1	4.18
18	*win*	*ndbh*			14	1.1	1.1

rules 11 and 15 can be glued to a single rule, in this case the preconditions would read $aSE \in \{spr, sum\} \wedge aOP = dbh$

XTT Prolog

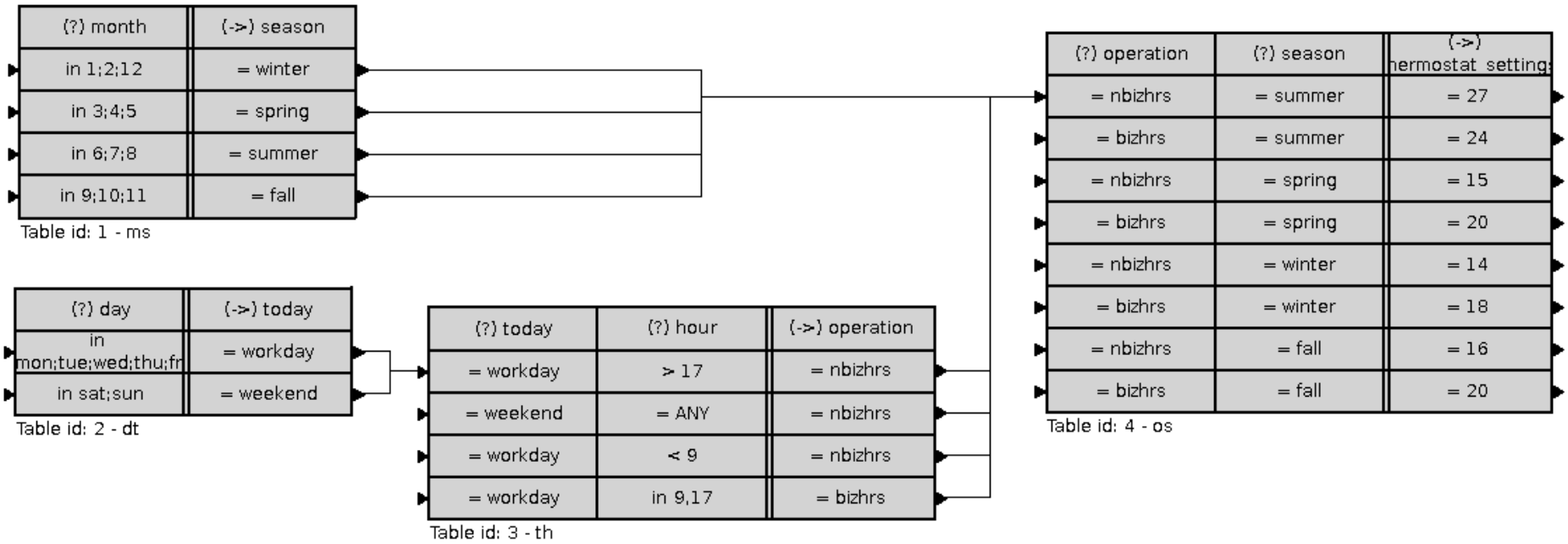
```
rule(4,11,[f(aSE,atomic,spr),f(aOP,atomic,dbh)],[],[],[f(aTHS,atom
rule(4,12,[f(aSE,atomic,spr),f(aOP,atomic,ndbh)],[],[],[f(aTHS,atc
rule(4,13,[f(aSE,atomic,sum),f(aOP,atomic,dbh)],[],[],[f(aTHS,atom
rule(4,14,[f(aSE,atomic,sum),f(aOP,atomic,ndbh)],[],[],[f(aTHS,atc
rule(4,15,[f(aSE,atomic,aut),f(aOP,atomic,dbh)],[],[],[f(aTHS,atom
rule(4,16,[f(aSE,atomic,aut),f(aOP,atomic,ndbh)],[],[],[f(aTHS,atc
rule(4,17,[f(aSE,atomic,win),f(aOP,atomic,dbh)],[],[],[f(aTHS,atom
rule(4,18,[f(aSE,atomic,win),f(aOP,atomic,ndbh)],[],[],[f(aTHS,atc
```

AGH

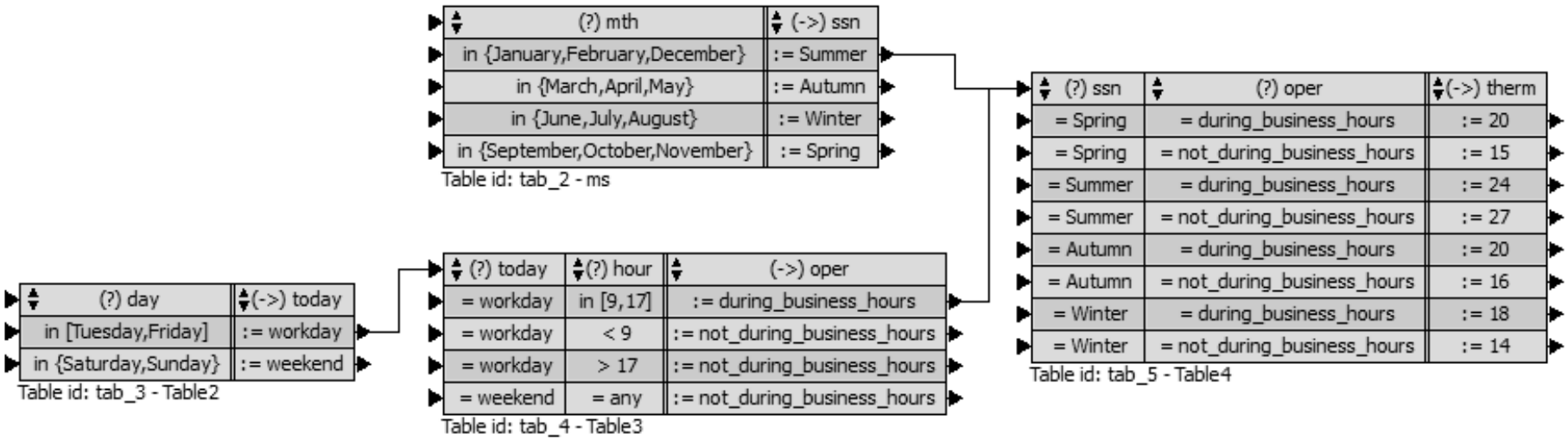
XTT Prolog: HMR

```
xschm th: [today,hour] ==> [operation].

xrule th/1:
  [today eq workday,
   hour gt 17]
  ==>
  [operation set not_bizhours].

xrule th/4:
  [today eq workday,
   hour  in [9 to 17]]
  ==>
  [operation set bizhours].
```

XTT Example


(?) month
(->) season
in 1;2;12
= winter
in 3;4;5
= spring
in 6;7;8
= summer
in 9;10;11
= fall
Table id: 1 - ms
(?) day
(->) today
in mon;tue;wed;thu;fr
= workday
in sat;sun
= weekend
Table id: 2 - dt
(?) today
(?) hour
(->) operation
= workday
> 17
= nbizhrs
= weekend
= ANY
= nbizhrs
= workday
< 9
= nbizhrs
= workday
in 9,17
= bizhrs
Table id: 3 - th
(?) operation
(?) season
(->) hermostat setting
= nbizhrs
= summer
= 27
= bizhrs
= summer
= 24
= nbizhrs
= spring
= 15
= bizhrs
= spring
= 20
= nbizhrs
= winter
= 14
= bizhrs
= winter
= 18
= nbizhrs
= fall
= 16
= bizhrs
= fall
= 20
Table id: 4 - os

XTT Example



AGH

XTT Prolog: Forms of Rules

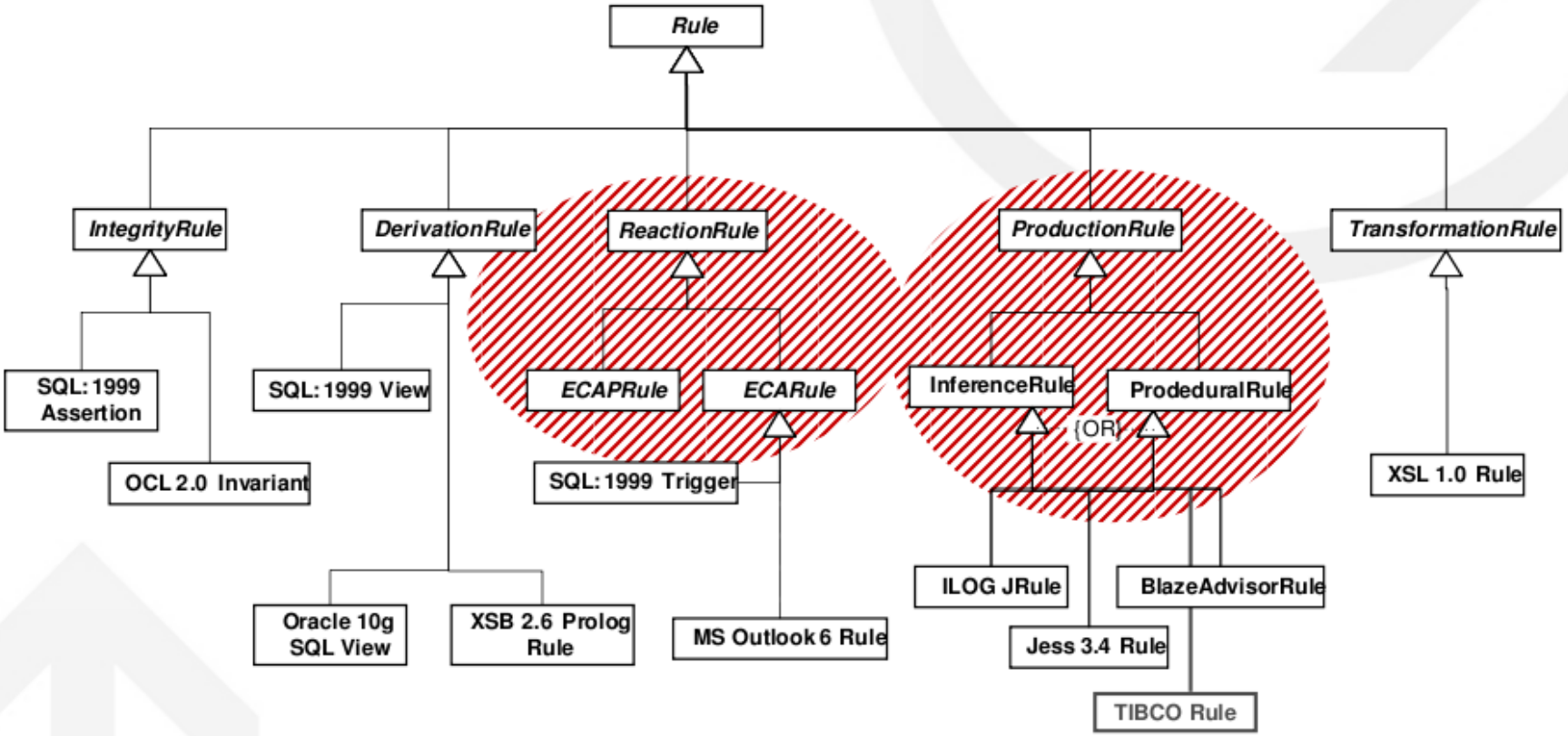
```
% rule(<table_number>,
%      <rule_number>,
%      [<precondition_list>],
%      [<retract_list>],
%      [<assert_list>],
%      [<decision_list>],
%      <next_table>,
%      <next_rule in next_table>,
%     ).

rule(1,1,[f(aDD,set,sWD)],[f(aTD,set,_)],[f(aTD,atomic,wd)],[],2,3
rule(1,2,[f(aDD,set,sWK)],[f(aTD,set,_)],[f(aTD,atomic,wk)],[],2,6

rule(1,1,[f(aDD,set,sWD)],[f(aTD,set,_)],[f(aTD,atomic,wd)],[],2,3
rule(1,2,[f(aDD,set,sWK)],[f(aTD,set,_)],[f(aTD,atomic,wk)],[],2,6
```

Possible Complex Forms of Rules

rule(*n*)		*name(parameters)*
resources		*resource(s)*
if		*context_formula*
	and	
if not		*excluding_condition(s)*
	and	
if		*precondition(s)*
	and	
if not		*detailed_excluding_condition(s)*
then		
	do	*action(s)*
	retract	*delete_result(s)*
	assert	*add_result(s)*
	output	*message(s)*
next		*rule(s)*
else		*rule(s)*

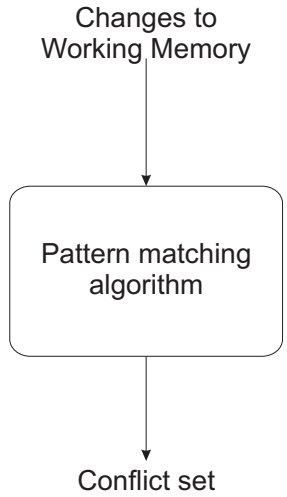
Types of Rules

Rule types for CEP



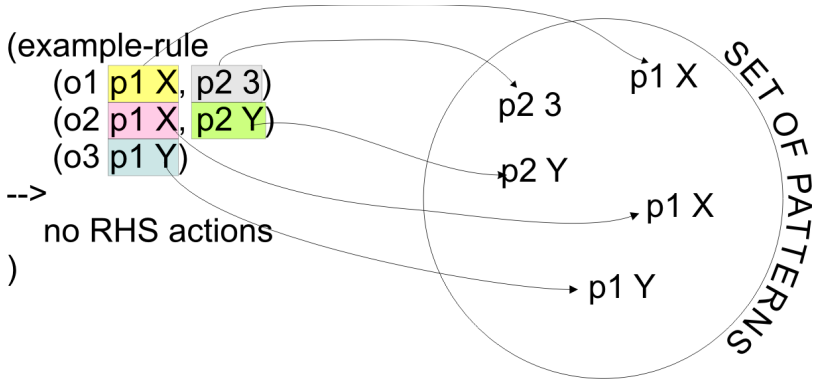
Rule Inference Algorithm

- An inference algorithm performs three steps:
  1. Pattern Matching.
  2. Conflict Set Resolution.
  3. Action Execution.
- *Pattern Matching* is a bottleneck of the inference process.
- The naive algorithm is far too slow.
- More efficient algorithms: RETE, TREAT, GATOR.



Rule Inference Algorithm – concepts

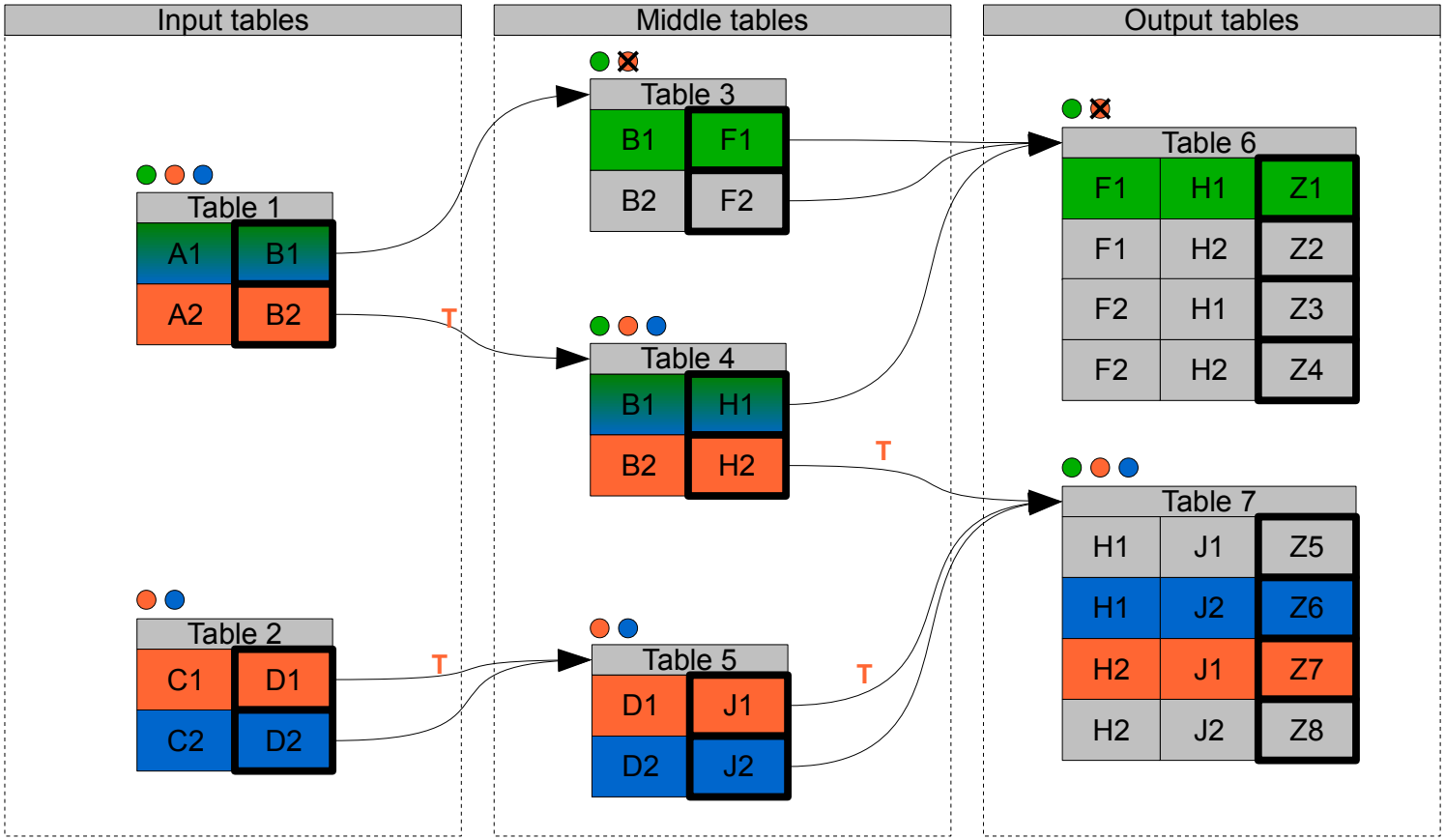
- **temporal redundancy** – *most* of the rules have RHS influencing *a few* facts only, and only *a few* rules are affected by those changes,
- **structural similarity** – many rules have a similar pattern in their LHP part.
- Facts are stored in the *Working Memory*.
- LHS consists of *patterns*:



- *Network* – a tree-like structure consisting of patterns.
- *Working element* – an object with attribute/value pairs describing it.

Object 1
property1 := 2 property2 := 12 property3 := 7 property4 := 11

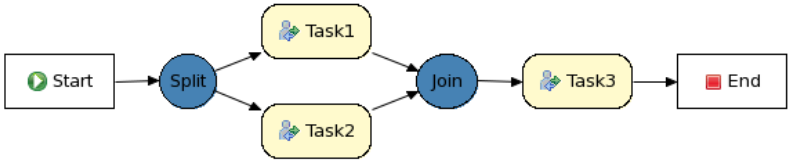
HeKatE XTT2: Inference Modes



- DDI with assumption that fact **A1** is in knowledge base and **Table 1** is a start table
- TDI with assumption that facts **A2** and **C1** are in knowledge base and **Table 7** and **Table 8** are goal tables
- GDI with assumption that facts **A1** and **C2** are in knowledge base, and **Table 7** is a goal table
- T Token sent from one table to another

Drools Flow

- Provides visualisation and a graphical user interface (GUI):



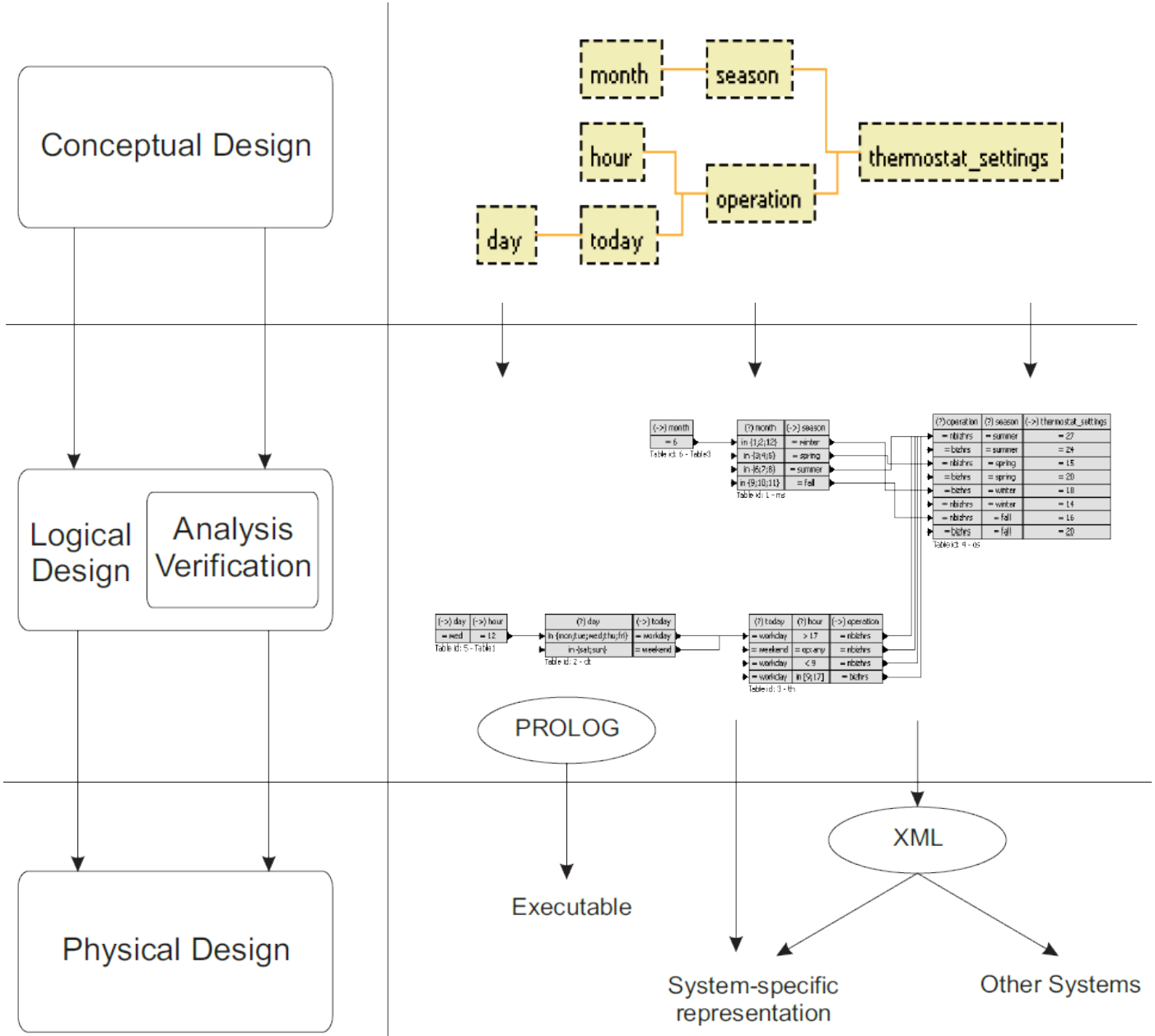
- Rules are stored in one global knowledge base.
- Rules can be grouped into *ruleflow-groups*:

```
rule Żule1"
ruleflow-group "Task1"
when
...
then
...
end
```

- Only rules from the current *ruleflow-group* are evaluated and fired.
- *Ruleflow-groups* determine the order of the rules evaluation and execution.

AGH

HeKatE XTT2: Complete Design Process



User Interface: Decision Tables

	Image	Price	Size	Dimensions	Weight	Battery Life	Ta
Objective	0	1	1	0	1	0	0
Report	1	1	0	1	0	0	1
Direction	1	-1	-1	-1	-1	1	1
Mitsubishi G75	<input type="im	59,95	14,63	5.5x1.9x1.4	7,9	3,24	5
Motorola g520	<input type="im	79,95	12,1	5.5x2.0x1.1	6	1,5	3,25
Ericsson 688	<input type="im	99,95	9,69	5.1x1.9x1.0	6,5	1,74	4
Nokia 5190	<input type="im	159,95	11,86	5.2x1.9x1.2	6	3,62	5
Motorola 6000	<input type="im	199,95	11,5	5.0x2.3x0.9	7,2	1,5	3,73
Ericsson 788	<input type="im	349,95	7,01	4.1x1.9x0.9	4,7	1,44	3,33
Motorola StarTAC 850	<input type="im	699,95	8,07	3.9x2.3x0.9	3,5	1,24	2,75
Bosch World 718	<input type="im	389,95	8,8	5.5x2.0 x 0.8	6,7	1,73	3,33

AGH

User Interface: Decision Tables

1.	**Do you or have you had a serious, disabling, or life-threatening condition such as stroke, heart, liver or kidney failure, cancer, etc.?**	○ Yes ◉ No ○ Unknown/not applicable
2.	**Do you have recurring unexplained episodes of any of the following symptoms?**	◉ A. Shortness of breath, palpitations, dizziness, or trembling ○ B. Sweating, nausea, choking sensations, or panic attacks ○ C. Numbness, tingling sensations, or unexplained symptoms such as disabling "aches and pains." ○ D. More than one of the above ○ E. None of the above ○ F. Unknown/not applicable
3.	**Have you undergone tattooing or body piercing (other than earrings)?**	○ Yes ◉ No ○ Unknown/not applicable
4.	Have you seen a psychiatrist, psychologist, or mental health worker in the past six months?	○ Yes ◉ No ○ Unknown/not applicable
5.	**Are your symptoms a diagnostic puzzle?**	○ Yes ○ No ◉ Unknown/not applicable
6.	Have you ever been treated or hospitalized for depression or a psychiatric condition?	○ Yes ◉ No ○ Unknown/not applicable

User Interface: Decision Tables

Depression Results

Condition/Disease	Estimated probability
Click on any disease for a description.	
Reactive Depression	76%
Major depression	12%
No significant depression	12%

What do these probabilities mean?

Millions of people visit the Internet daily in search of information about their complaints and conditions. *EasyDiagnosis* offers a novel interactive resource t assist health site visitors **bypass vast amounts of irrelevant medical news and information** offered by traditional medical web sites.

SUBSCRIBE **Why subscribe?**

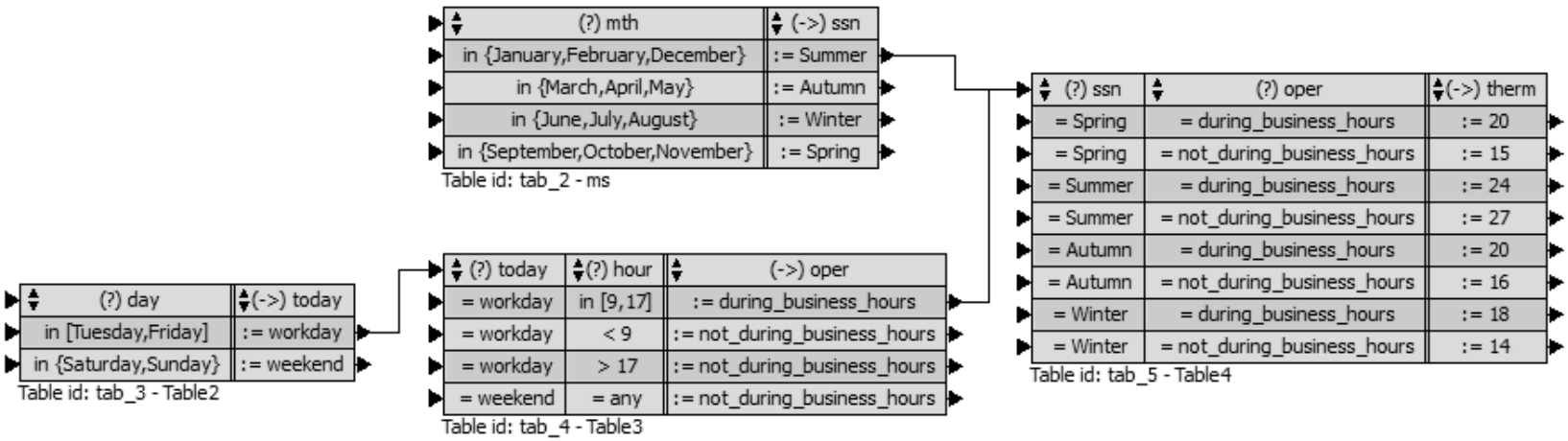
Links

Drools5

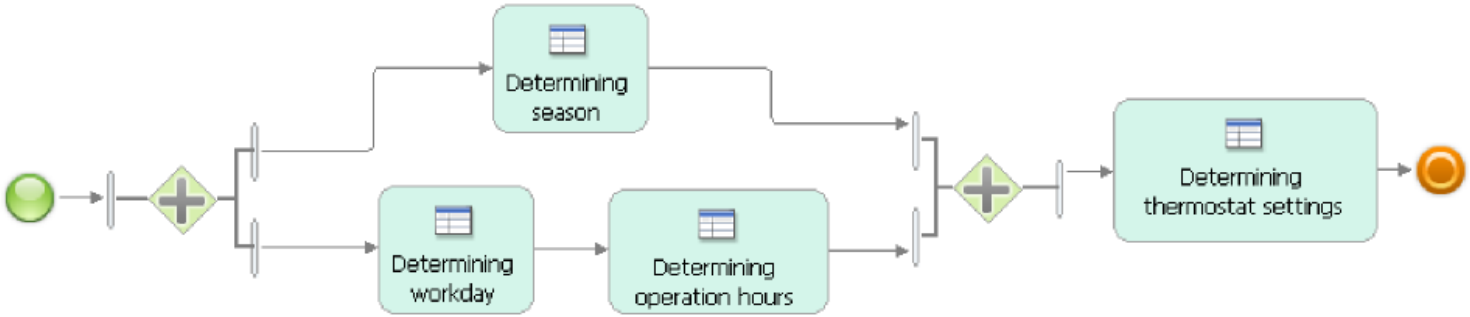
- More than a classic expert system shell – provides a platform for integration of processes and rules.
- Consists of four modules:
  - Drools Guvnor – knowledge base repository.
  - Drools Expert – rule engine.
  - Drools Flow – workflow modelling.
  - Drools Fusion (event processing/temporal reasoning).
- Only provides forward chaining.
- Inference engine uses a RETE-based algorithm.
- Knowledge represented as rules in Drools5 format:

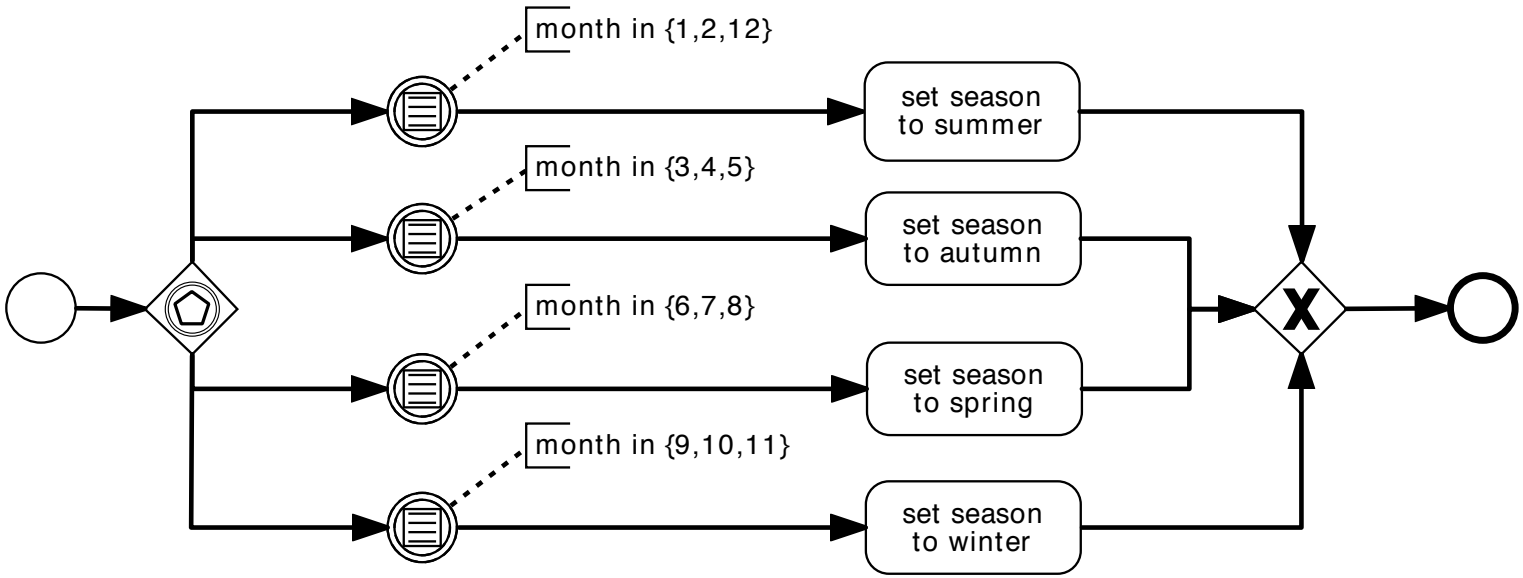
```
rule ŻuleName"when
  // conditions
then
  // actions

end
```

XTT Example

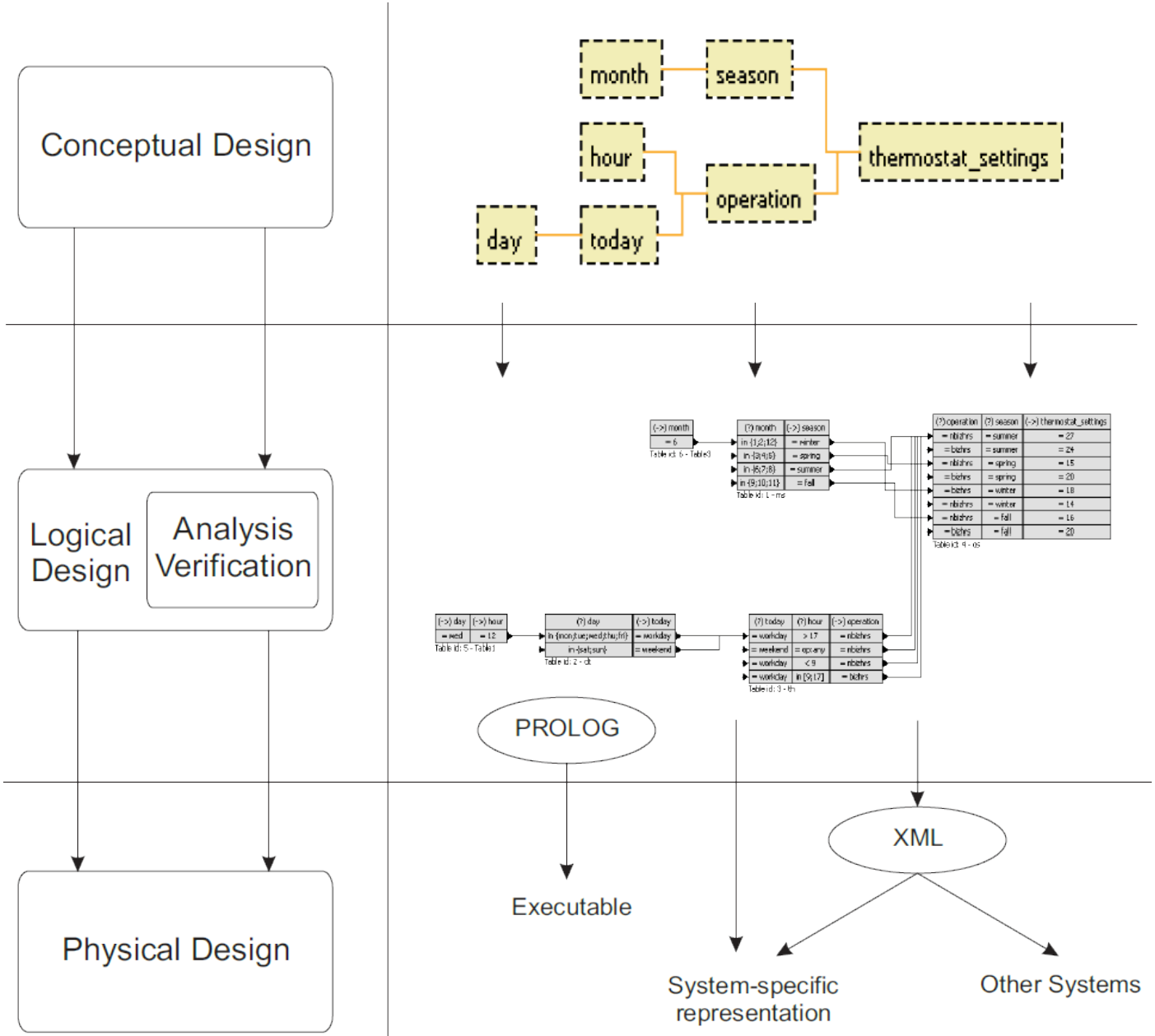

(?) mth
(->) ssn
in {January,February,December}
:= Summer
in {March,April,May}
:= Autumn
in {June,July,August}
:= Winter
in {September,October,November}
:= Spring
Table id: tab_2 - ms
(?) day
(->) today
in [Tuesday,Friday]
:= workday
in {Saturday,Sunday}
:= weekend
Table id: tab_3 - Table2
(?) today
(?) hour
(->) oper
= workday
in [9,17]
:= during_business_hours
= workday
< 9
:= not_during_business_hours
= workday
> 17
:= not_during_business_hours
= weekend
= any
:= not_during_business_hours
Table id: tab_4 - Table3
(?) ssn
(?) oper
(->) therm
= Spring
= during_business_hours
:= 20
= Spring
= not_during_business_hours
:= 15
= Summer
= during_business_hours
:= 24
= Summer
= not_during_business_hours
:= 27
= Autumn
= during_business_hours
:= 20
= Autumn
= not_during_business_hours
:= 16
= Winter
= during_business_hours
:= 18
= Winter
= not_during_business_hours
:= 14
Table id: tab_5 - Table4

BPMN: Modeling with Rules





HeKatE XTT2: Complete Design Process



Rule-Based Systems

Examples

- OPS5, OPS 83,
- CLIPS, JESS,
- Drools,
- G2 (Gensym),
- Sphinx/PC-Shell,
- BizTalk Rules Engine,
- XpertRules, ILOG JRULES, Soar.

Active Areas

- Datalog, Prolog, Erlang,...
- Constraints: CLP, ECLIPSE-CLP, CHOCO,
- Answer Set Programming,
- RuleML initiative,
- Semantic Web Stack: Datalog,
- Business Rules.

The End

Many Thanks for your attention!