

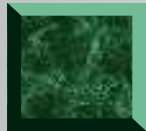
Managing Knowledge in Small and Medium Sized Companies



1. Definitions, appliance domains, and practical usage



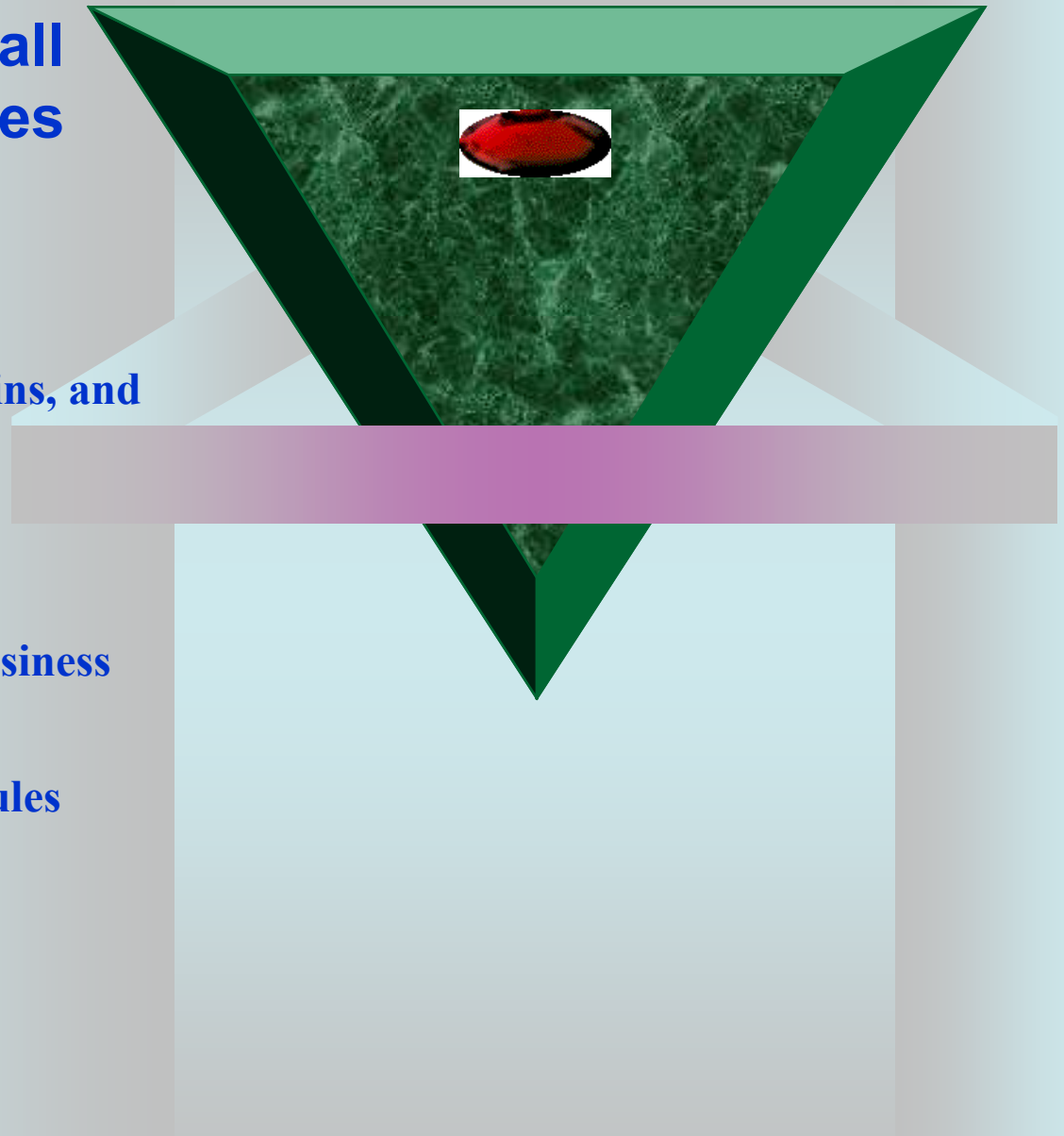
2. Formalizing Rules



3. Designing and Managing Business Rules with ARulesXL



4. Importing and Exporting Rules



1. Definitions, appliance domains, and practical usage

=>Data, Information, Knowledge, and Wisdom

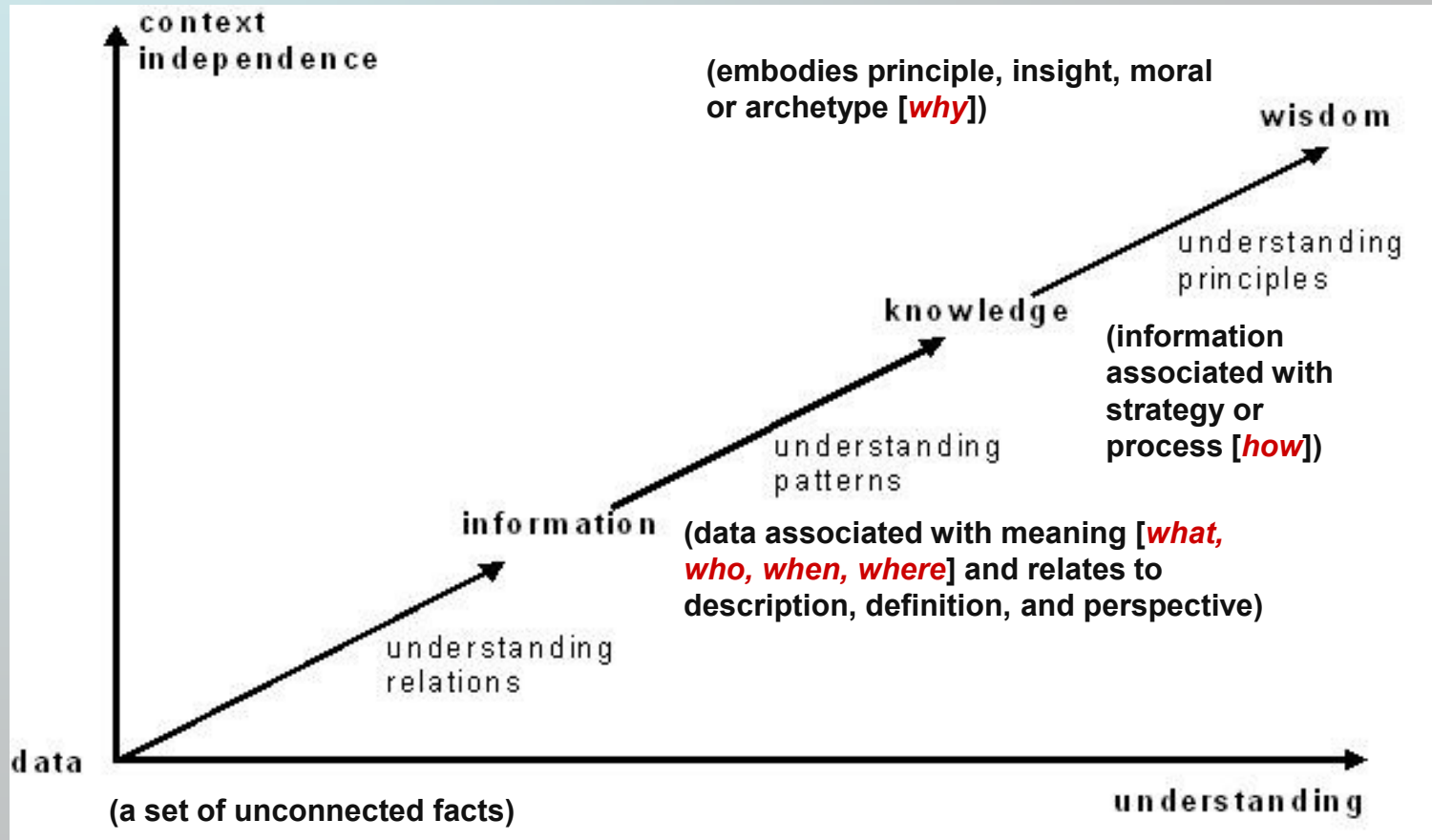


Figure 1 The relationships between data, information, knowledge, and wisdom considering the context independence and understanding (Source: Neil Fleming, Coping with revolution: Will the Internet change learning? Lincoln University, Canterbury, New Zealand ; With except of comments in round parenthesis, that I added as comments in the presentation)

Definitions, appliance domains, and practical usage

IT solutions to Knowledge Management centres to:

-business process management (BPM) , process-centric focusing on workflow around of a given process .

The business process can be defined as “the unique ways in which organizations coordinate and organize work activities, information, and knowledge to produce product or service” [1]

- business rules engines (BRE) applicable to business rules that determine BPM process.

The two solutions, BPM and BRE, are integrated into a more complex one solution: business rules management systems (BRMS).



Definitions, appliance domains, and practical usage



Business Rules:

- the day-to-day politics that determines the interaction way with customers, suppliers, employees, and partners;
- describe the operations, definitions, and constraints that applies to a company in realizing and attaining its roles and goals.



Definitions, appliance domains, and practical usage

Business rules represented by:

- **facts**: data about the business environment (such as orders, invoices, information about consumers, information about employees etc);
- **simple instructions**: of the de forma $\langle \text{if } (\textit{condition}) \text{ then } \textit{actions} \rangle$ in which *condition* evaluated by comparison with a set of facts and *actions* executed if the condition is satisfied, as in the following samples:
 - ✓ If the client is favorite (as Gold or Platinum to Vodafone, for example) then rise the given discount with 3%;
 - ✓ If the employee works during the current month overtime then pay him with 1 and half (1,5 times) the normal hourly salary for hours worked overtime.
- **actions**: simple returns or calls to direct actions (for example: If QuantityPurchased ≥ 50 Then allay a discount of 30%). An action can be the source of generating a new fact (this is called *inference*).





Definitions, appliance domains, and practical usage

The appliance domains of business rules includes:

- Marketing strategies;
- Price politics;
- Management practices used in the client relationship;
- Human resources related activities;
- Control forces, either to insure the conformity with the laws/ regulations, either to make decisions in accordance with laws/ regulations;
- Products and services offers;
- Insurance and healthcare;
- Finance and risk analysis;
- Operational management;
- Call centers.



Definitions, appliance domains, and practical usage

- The business rules approach is applicable to both humans and IT systems.
- Business Rules Engine allows define the rules in a simple language so that the users are not constrained to understand or know a programming language.
- The rules architecture continue the evolution of programming (figure 2):
 - Initially programming languages such as C/C++ or Java;
 - Internet adds HTML for logical display;
 - Business Rules adds rules for policies.

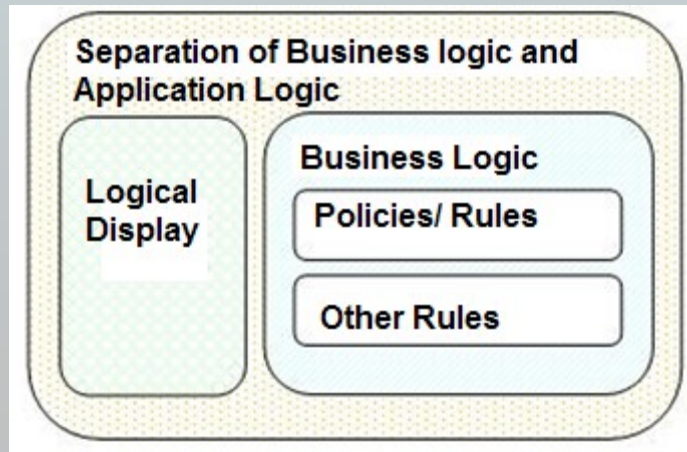


Figure 2 Rules Architecture
(Source: Oracle Business Rules)

Definitions, appliance domains, and practical usage

- The Rules Engines use the rules to analyze the facts (figure 3).
- After the facts analysis the Rules Engine can return results (represented by simple values or by complex data structures) or can invoke actions.

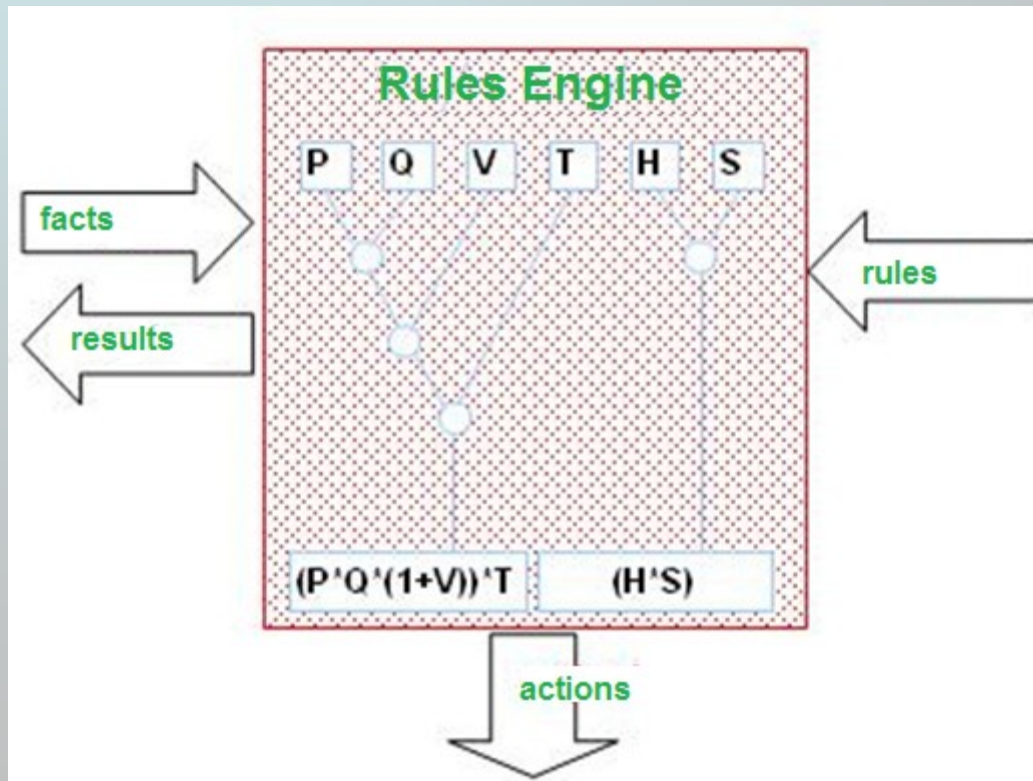


Figure 3 Rules Engine
(Source: Rule Enabling Applications with Oracle Business Rules, Oracle Whitepaper, September 2005)

Definitions, appliance domains, and practical usage

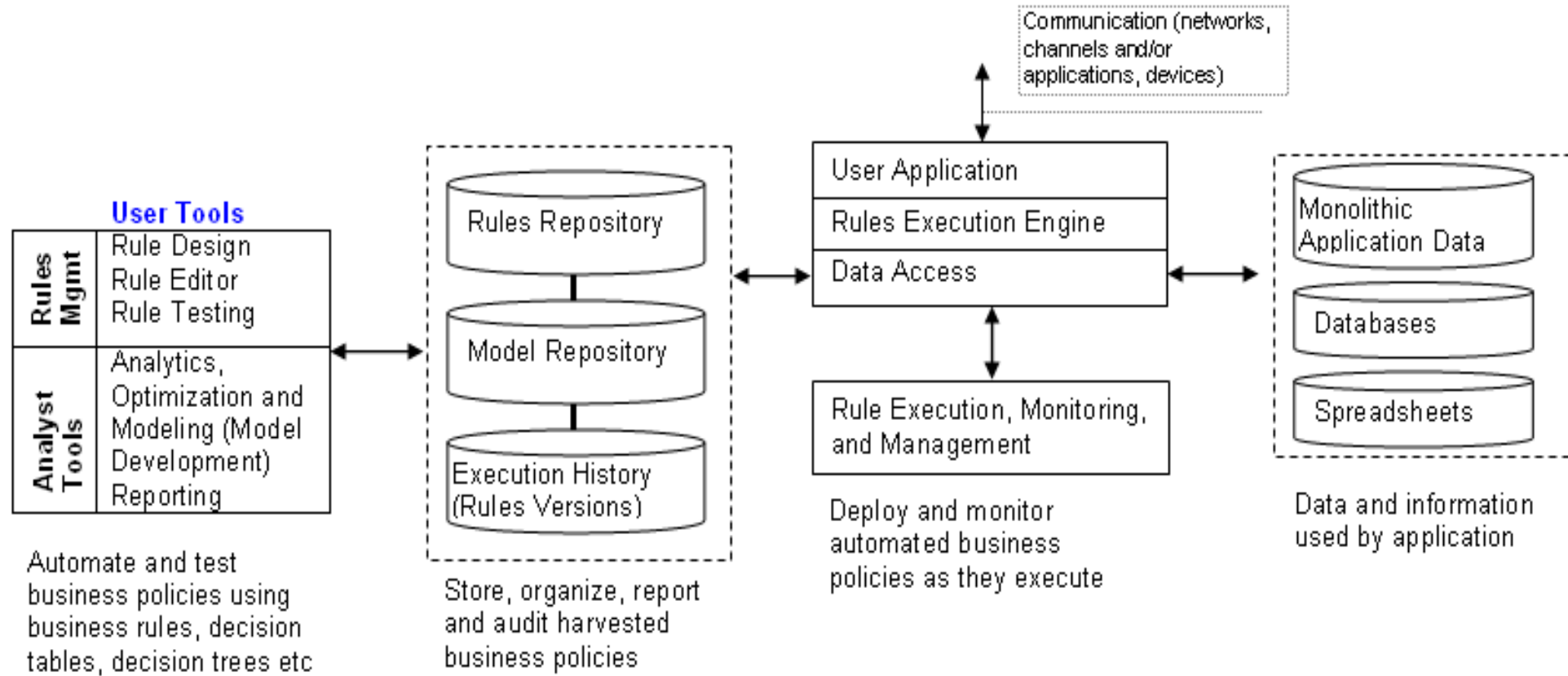


Figure 4 BRMS components (Source: [3])



Definitions, appliance domains, and practical usage

Within BRMS:

Business policy experts can manage and evolve business policies using the methods and vocabulary with which they are most familiar;

Technology experts can manage and evolve technology using methods and vocabulary most suitable to their tasks.

The information management solution for small and midsize companies is to fully integrate, share and deliver information so that they are able to improve accuracy and speed of decision making by providing consistent, rapid and secure access to all relevant information. This type of information management implies the usage of business rules management systems. As a small to midsize business, budgets don't allow for a rip-and-replace approach and the software licensing can be prohibitive and restrictive, leaving few options for complete information management [2].



Definitions, appliance domains, and practical usage

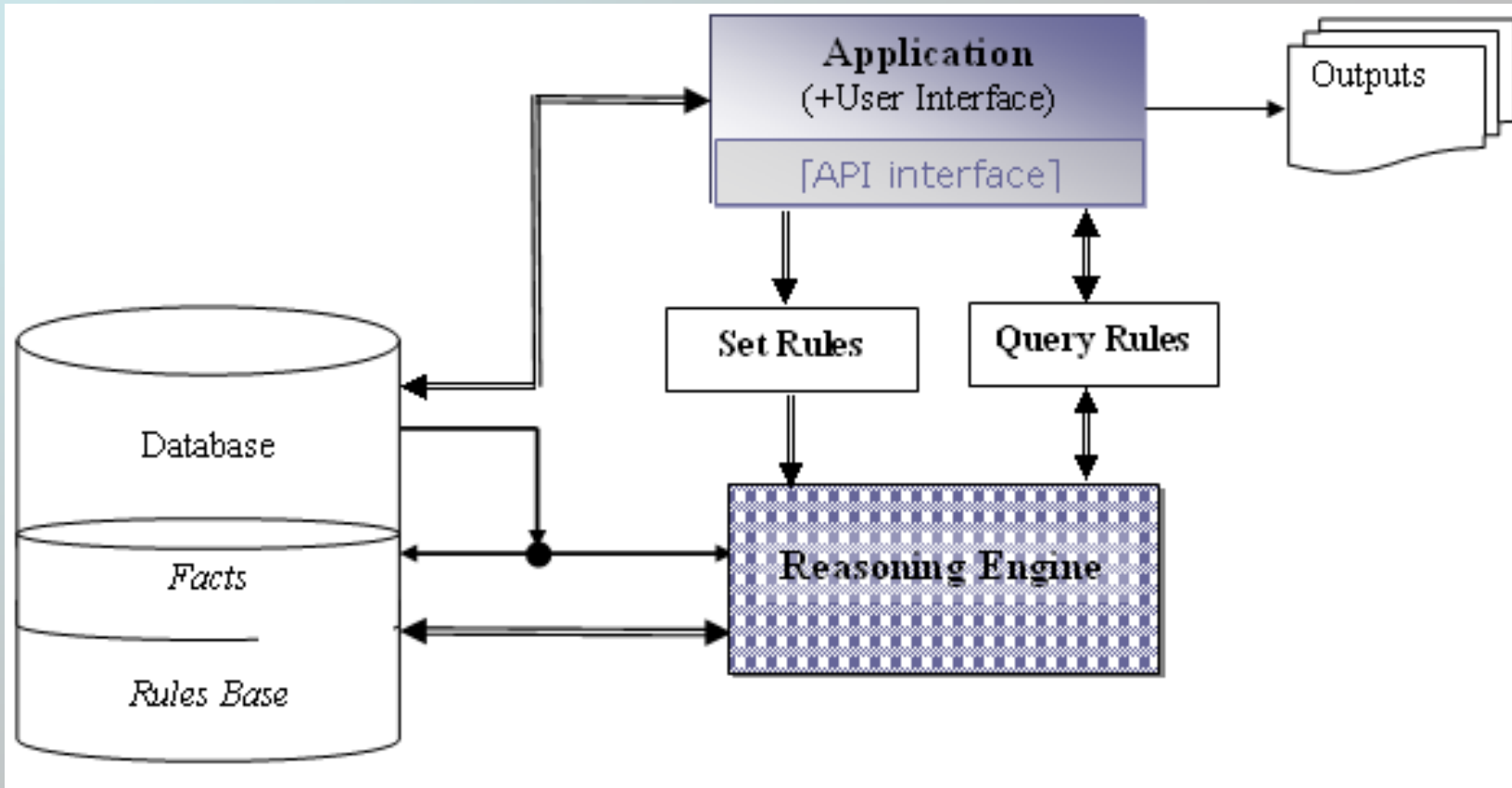


Figure 5 Integrating databases together with rules bases via reasoning engines

Definitions, appliance domains, and practical usage

➤ Offered Advantages:

Increased Agility: the business analysts can quickly develop and deploy business rules that reflects the new business policies.

High Transparency: the business analysts can review the rules and can easy deduce the policies they implement.

Low Costs: the business policies are implemented using authorized facilities that are extremely efficient and that are offered via an efficient interface for describing policies.

Reducing the dependence on IT: business analysts can directly implement the new policies or those affected by the change of working rules that don't necessitate or necessitate a little assistance from the part of IT.



2. Formalizing Rules

Designing and managing rules with ARulesXL

- The Rules Engine ArulesXL (figure 6) offers interfaces to:
- Define and maintain the rules collections (rule base);
 - Query the rule base and obtaining results by supplying the required input data.

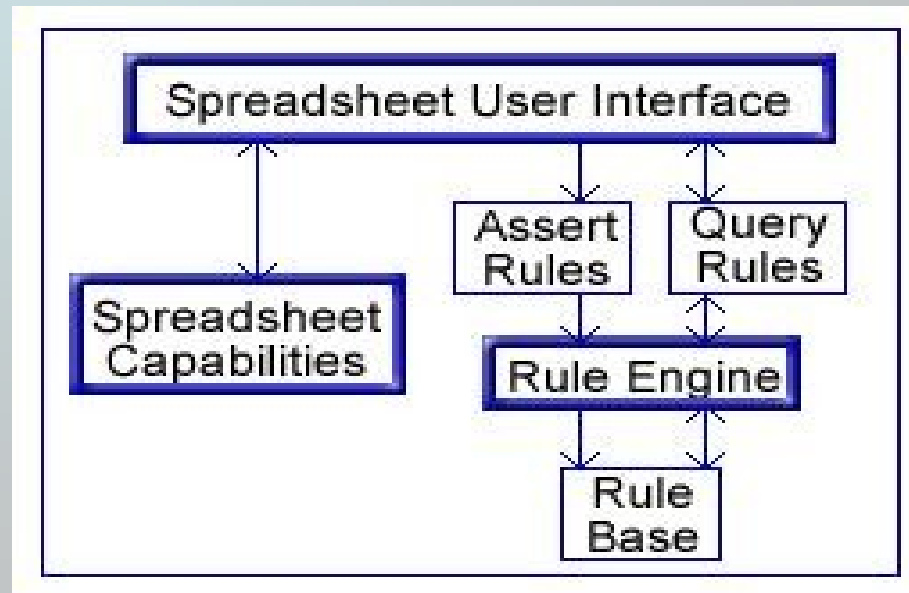


Figure 6 The Rules Model of ArulesXL

(Sursa: <http://www.arulesxl.com>)

Formalizing, Designing, and Maintaining Rules

For the Payroll Problem the rules here can be expressed directly in a natural language as follows:

- the salary is obtained by multiplying the worked hours with the tariff when is not weekend and no overtime worked;
- the salary is obtained by multiplying the allowed time with the tariff and by raising that with one and half time the salary computed for the difference between the worked time and the allowed time when is overtime or weekend;
- the weekend is when the day is “Saturday” or “Sunday” or is “Friday” after the beginning of weekend.

The facts and rules are interpreted together by the reasoning engine (figure 7 and 8) which determines for every cell in column Salary in the table the right salary computation rule depending on the facts recorded (day, allowed time, worked time) into the table and other general parameters (such as salRaiseOverTime).



Formalizing, Designing, and Maintaining Rules

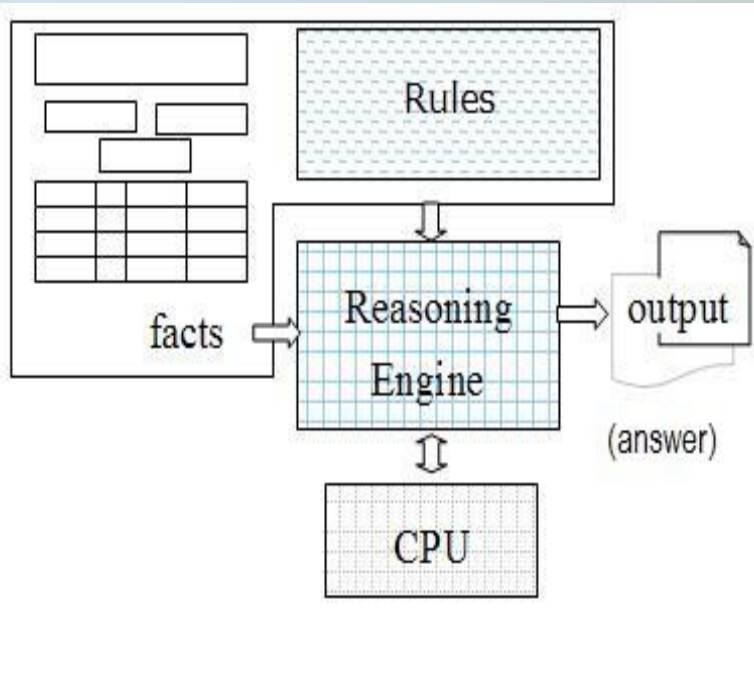


Figure 7 The representation of processing for Logical knowledge [ARulesXL]

Each spreadsheet implementing a Business Rule solution uses three basic building blocks, as illustrated in figure 8:

- Outputs: the result of querying the rules;
- Inputs: values that comes from the user, data tables and other sources;
- Rules: deduces values for outputs based on inputs.

that acts under a <Inputs-Processing-Outputs> pattern.

Formalizing, Designing, and Maintaining Rules

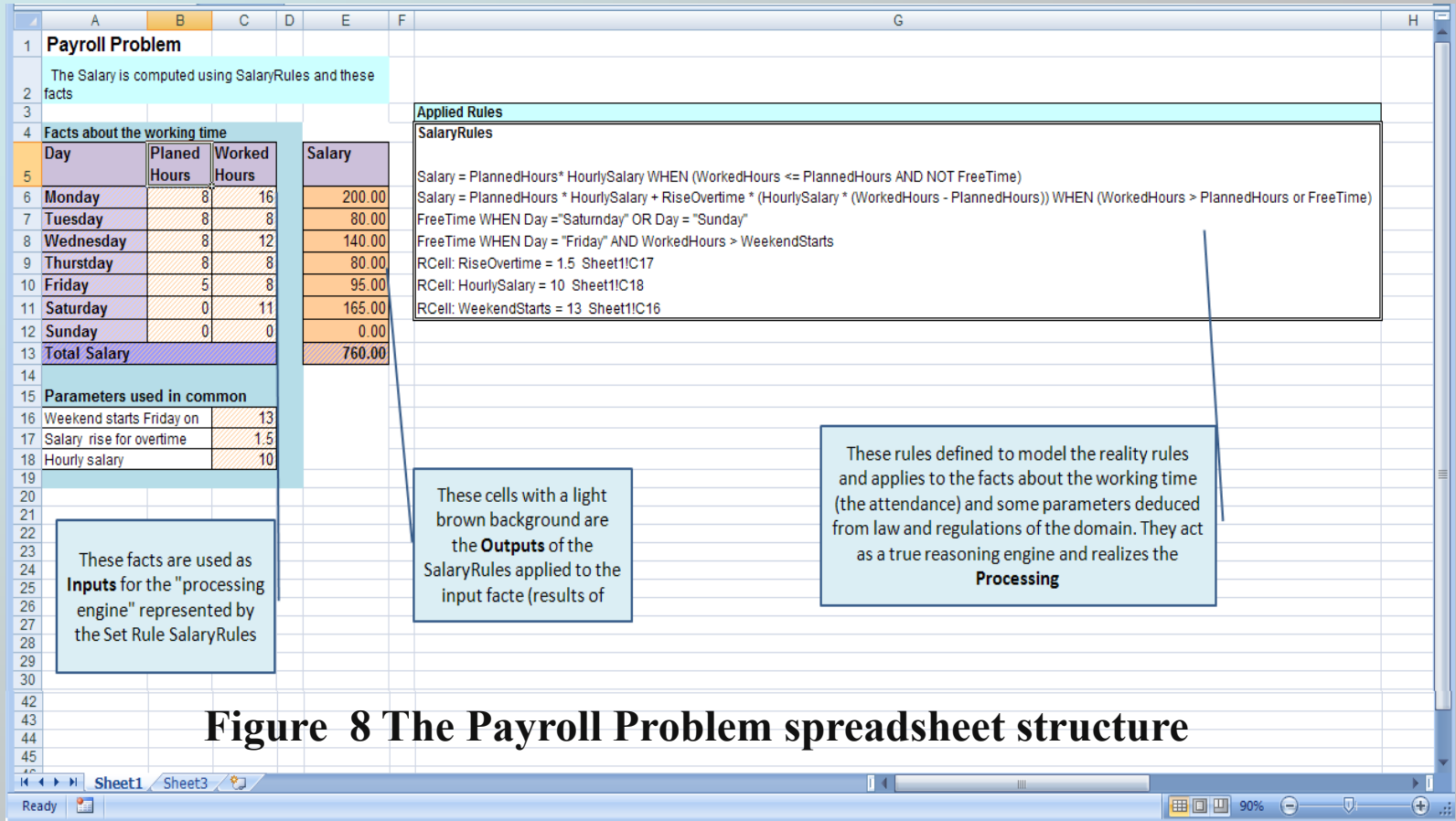


Figure 8 The Payroll Problem spreadsheet structure

Each spreadsheet implementing a Business Rule solution uses three basic building blocks, as illustrated in figure 8:

- **Outputs:** the result of querying the rules;
 - **Inputs:** values that comes from the user, data tables and other sources;
 - **Rules:** deduces values for outputs based on inputs.
- that acts under a **<Inputs-Processing-Outputs>** pattern.





Formalizing, Designing, and Maintaining Rules

Inputs are specified with `RArray()` and/ or `RCell()`

`=RArray("name", range, ifRowHeader, ifColumnHeader, storeAsUnidimensionalVector)`

where:

- "*name*" is the name we give to the input array;
 - *range*, is the range of cells for the input. The input array must be a contiguous range of cells;
 - *ifRowHeader*, is TRUE if we want include the headers for rows. The header can be referenced in the array index as `input["row header"]` for retrieval (as an index for the array). If the header not required use FALSE;
 - *ifColumnHeader*, is TRUE if we want include the headers for columns. The header can be referenced in the array index as `input["column header"]` for retrieval (as an index for the array). If the header not required use FALSE;
 - *storeAsUnidimensionalVector*, is TRUE if we want represent the multi-dimensional array (maybe) as a unidimensional vector. Otherwise use FALSE.
- 



Formalizing, Designing, and Maintaining Rules

=RCell("variableName",cellAddress)

where:

-**"variableName"**, is the name you want give to the value in the cell given by *cellAddress*. - *cellAddress* is a reference (address) of a spreadsheet cell.

Examples RCell() definitions (figure 8):

SalaryRules

...

=RCell("RiseOvertime", C17)

=RCell("HourlySalary", C18)

=RCell("WeekendStarts", C16)



Rules

The rule set is a named collection (the name is specified in the first cell/ line of the range for definition). The rules can be defined in any order because they are declarative. For easy management purpose is preferable that they defined grouped. The base expression for rules is

Fact = Value WHEN conditions

where:

- Fact**, is the name of a fact for which we specify a new rule;
- Value**, is an expression that can be a number, a text or a date to be assigned to Fact;
- Condition**, is a Boolean expression that tell when the fact take the value specified by the rule.

Formalizing, Designing, and Maintaining Rules

Examples Rules (figure 8):

Salary = PlannedHours * HourlySalary WHEN (WorkedHours <= PlannedHours AND NOT FreeTime)

Salary = PlannedHours * HourlySalary + RiseOvertime * (HourlySalary * (WorkedHours - PlannedHours)) WHEN (WorkedHours > PlannedHours or FreeTime)

FreeTime WHEN Day = "Saturday" OR Day = "Sunday"

FreeTime WHEN Day = "Friday" AND WorkedHours > WeekendStarts



Output

The cells for answers to the status contains queries defined as
`=RQuery(ruleSetName, "find aFactName")`

where:

- *ruleSetName*, is the name of the Rule Set in which you search for output;
- "find *aFactName*" is a simple expression as defined here, that find in the rule set the fact according with the inputs and the evaluation of conditions attached to the rules. Generally is a search sentence of the form:

FIND Fact WHEN InputFact1=Value1 AND InputFact2=Value2 ...



Formalizing, Designing, and Maintaining Rules



Output contains queries of the form (with A6,B^,C6 adapted to the new cell coordinates):

=RQuery(SalaryRules, "FIND Salary WHEN Day = _1 and PlannedHours = _2 and WorkedHours = _3", A6, B6, C6) with:

A6 the cell containing the Day name

B6 the cell containing the value for Planned Hours

C6 the cell containing the value for Worked Hours

The cell Total Salary in the output contains a normal Excel Sum() for the salary obtained each day



Formalizing, Designing, and Maintaining Rules



Defined standards at W3C – World Wide World Consortium (by working groups or adopted):

→ RIF (Rules Interchange Format) for Web

→ RuleML – Rule Markup Language

→ BRML – Business Rules Markup Language; Business Rules Management Systems

→ RDF – Resource Description Framework

→ Standards for Rules for Semantic Web

→ ...



Annex 1. Exemplu de utilizare a unui motor de reguli descrise în foi de calcul electronice (source [4])

Exemplu Salarii

Date pentru timpul de lucru
Salariul este calculat utilizand ReguliSalariu si aceste date

Zi	OreNormate	OreLucrate	Salariu
Luni	8	16	360.00
Marti	8	8	144.00
Miercuri	8	12	252.00
Joi	8	8	144.00
Vineri	5	8	171.00
Sambata	0	11	297.00
Duminica	0	0	0.00
Total Salariu			1,368.00

Parametrii utilizati in comun:

Timpul Liber incepe Mnerea la ora:	13
Salariu ore suplimentare	1.5
Salariu Orar	18

ReguliSalariu

- 1 `.Salariu = .OreNormate * .SalariuOrar
WHEN (.OreLucrate <= .OreNormate AND NOT
.TimpLiber)`
- 2 `.Salariu = .OreNormate * .SalariuOrar +
.MarireOreSup * (.SalariuOrar * (.OreLucrate
- .OreNormate))
when (.OreLucrate > .OreNormate or .TimpLiber)`
- 3 `.TimpLiber WHEN .Zi =Sambata OR .Zi =
Duminica`
- 4 `.TimpLiber WHEN .Zi = Vineri AND .OreLucrate >
.InceputTimpLiberVineri`
- 5 `RCell: .MarireOreSup = 1.5 Sheet1!C18`
- 6 `RCell: .SalariuOrar = 18 Sheet1!C19`
- 7 `RCell: .InceputTimpLiberVineri = 13 Sheet1!C17`

Regulile aplicate

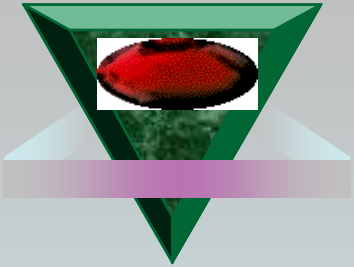
```
ReguliSalariu
.Salariu = .OreNormate * .SalariuOrar WHEN (.OreLucrate <= .OreNormate AND NOT .TimpLiber)
.Salariu = .OreNormate * .SalariuOrar + MarireOreSup * (.SalariuOrar * (.OreLucrate - .OreNormate)) when (.OreLucrate > .OreNormate or .TimpLiber)
.TimpLiber WHEN .Zi =Sambata OR Zi = Duminica
.TimpLiber WHEN .Zi = Vineri AND .OreLucrate > .InceputTimpLiberVineri
RCell: MarireOreSup = 1.5 Sheet1!C18
RCell: .SalariuOrar = 18 Sheet1!C19
RCell: .InceputTimpLiberVineri = 13 Sheet1!C17
```


Annex 1 (cont). Exemplu de utilizare a unui motor de reguli descrise în foi de calcul electronice (Source [4])

Interpretarea regulilor:

1. Dacă nu este timp liber (weekend) și orele lucrate sunt mai puține sau egale cu cele normate atunci salariul se obține prin calculul $\text{Ore Lucrate} \times \text{Salariu Orar}$
2. Dacă este timp liber sau a lucrat în afara programului atunci pentru orele normate se acordă salariu normal iar pentru cele în afara programului se acordă de 1,5 ori suma normală;
- 3 și 4. Timpul liber este toată Sâmbăta și Duminica iar Vineri începe de la ora la care începe weekend-ul;
- 5, 6 și 7 definesc celulele care conțin parametrii comuni cu numele folosit la specificarea regulilor.

Coloana Salariu conține aceeași frază de interogare a regulilor:
`=ARulesXL.xla!RQuery(ReguliSalariu, "FIND .Salariu WHEN .Zi = _1 AND .OreNormate = _2 and .OreLucrate = _3", A6, B6, C6)`
(cu adaptarea corespunzătoare a adreselor de celule Ax, Bx și Cx). `_1`, `_2` și `_3` sunt parametrii și sunt reprezentați de celulele din tabel Ax, Bx și Cx.



References:

- [1]. Vasile Avram, Gheorghe Dodescu – Informatics: Computer Hardware and Programming in Visual Basic, Editura Economica, pages 11-46
- [2]. Vasile Avram, The Informatics and Enterprise in Information Society. Virtual organization., 2008, <http://www.avrams.ro/master/>
- [3]. AVRAM Vasile, Diana Avram, Managing knowledge within Small and Midsized Companies, rev. Informatică Economică, Nr. 44, ISSN: 1453-1305
- [4]. AVRAM Vasile, Diana Avram, Utilizarea sistemelor bazate pe proceduri pentru descrierea și automatizarea regulilor afacerii, in volume Calitate, Management, Integrare Europeana, Ed. Cartea Universitară, 2007, ISSN: 1582-2559
- [5] Vasile AVRAM (2007) - *Replacing Factual and Procedural Knowledge by Logical Knowledge in Application Software*, Review of Management and Economical Engineering Vol. 6, No. 5, ISSN: 1583-624X, 2007
- [6]. The contribution of Knowledge Management Systems to Interorganizational Learning, Marla Beth Greenman, SIGMIS-CPR'06, ACM 1-59593-349-2/06/0004, 2006
- [7]. An Exploratory Study on the Roles of Network Structure and Knowledge Processing Orientation in Work Unit Knowledge Management - Seokwoo Song , Sridhar Nerur, James T.C. Teng, The DATABASE for Advances in Information Systems, Volume 38, Number 2, May 2007
- [8]. The Knowledge Pyramid: A Framework for Understanding the Challenges and Opportunities of Managing Decisions as an Enterprise Asset - CORTICON Technologies, Inc., www.corticon.com
- [9]. Albin, Stephen T. The Art of Software Architecture: Design Methods and Techniques. John Wiley & Sons. © 2003. Books24x7.
<http://common.books24x7.com/book/id_6020/book.asp>
- [10]. Len Bass; Paul Clements; Rick Kazman, Software Architecture in Practice, Second Edition, Addison-Wesley Professional, 2003
<<http://acmsel.safaribooksonline.com/0321154959>>
- [11]. Michael Harvey, Essential Business Process Modeling, O'Reilly, 2005



Bibliografie:

- 
- [12]. Claudia M. Baca, Project Management for Mere Mortals®: The Tools, Techniques, Teaming, and Politics of Project Management, Addison-Wesley Professional, 2007, <http://acmsel.safaribooksonline.com/9780321423450>
- [13]. Architecture Knowledge Management: Challenges, Approaches, and Tools - Muhammad Ali Babar, Ian Gorton, 29th International Conference on Software Engineering (ICSE'07 Companion), 0-7695-2892-9/07 IEEE Computer Society, 2007
- [14]. Tim Berners-Lee, Dan Connolly, Ralph R. Swick, Web Architecture: Describing and Exchanging Data, W3C Note 7 June 1999, www.w3c.org
- [15]. Daconta, Michael C., Leo J. Obrst, and Kevin T. Smith. The Semantic Web: A Guide to the Future of XML, Web Services, and Knowledge Management. John Wiley & Sons. 2003. Books24x7. http://common.books24x7.com/book/id_6073/book.asp
- [16]. ArulesXL – <http://www.arulexl.com>
- [17]. <http://www.forrester.com>
- [18]. <http://www.w3c.org> – World Wide Consortium, Web Standards
- [19]. Building Flexible Enterprise Processes Using Oracle Business Rules and BPEL process Manager, Oracle Corporation, Ian 2005, <http://www.oracle.com>
- [20]. Defining Business Rules – What Are They Really ?, Business Rules Group, Final Report, revision 1.3, July 2000, <http://www.businessrulesgroup.org>
- [21]. <http://www.objectconnections.com> – Common Knowledge 3.0
- [22]. Business Rules for Electronic Commerce, Project at IBM T.J. Watson Research, <http://www.ibm.com>
- [23]. Oracle Business Rules, Bruce Lowenthal, <http://www.oracle.com>
- [24]. Business Rules Markup Language (BRML), <http://xml.coverpages.org>
- [25]. Business rules management systems, www.infoworld.com
- [26]. www.javarules.org
- [27]. www.essentialstrategies.com/publications/businessrules/
- 