Acta Informatica Pragensia ISSN 1805-4951

Template v1.0.

Online: aip.vse.cz

Business Rules Definition for Decision Support System Using Matrix Grammar

Eva Zámečníková[1], Jitka Kreslíková[1]

[1] Department of Information Systems, Faculty of Information Technology,
Brno University of Technology, Brno
Božetěchova 1/2, 612 66 Brno

{izamecni, kreslika}@fit.vutbr.cz

Abstract: This paper deals with formalization of business rules by formal grammars. In our work we focus on methods for high frequency data processing. We process data by using complex event platforms (CEP) which allow to process high volume of data in nearly real time. Decision making process is contained by one level of processing of CEP. Business rules are used for decision making process description.

For the business rules formalization we chose matrix grammar. The use of formal grammars is quite natural as the structure of rules and its rewriting is very similar both for the business rules and for formal grammar. In addtion the matrix grammar allows to simulate dependencies and correlations between the rules.

The result of this work is a model for data processing of knowledge-based decision support system described by the rules of formal grammar. This system will support the decision making in CEP. This solution may contribute to the speedup of decision making process in complex event processing and also to the formal verification of these systems.

Keywords: Business rule, decision support system, formalization, decision making, matrix grammar.

1 Introduction

Our research work is focused on a design of a method for formalization of business rules. A new method will be implemented as a part of knowledge-based decision support system and will be used during decision making process. This process is used by complex event platform (CEP) for better prediction of data. We would like to use the method in order to speed up the decision making process. The purpose of CEP is to analyze events and to find situations of interest. CEP detects and derives information, so we can become aware of a situation immediately and react in the best possible way. For the prediction of high frequency data, an existing solution of complex event processing platform will be used, where it is possible to implement our own decision support system module. CEP will ensure the prediction of data and we will add the component for decision making. Examples of high frequency data include: telecommunications, energy or market data used for algorithmic trading.

Complex event processing platforms are a set of tools for the support of the preprocessing, processing and prediction of complex events. These platforms are designed for processing of data from multiple different sources and primarily focus on processing of moving data streams in real time. These data are processed on several levels of abstraction according to the required level of interference. The output of the process is pattern recognition, mining of trends and patterns in data and thus predicting the flow of next input data.

CEP detailed description is not part of this paper, for more information about this topic authors recommend David Luckham's book The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems (Luckham, 2002).

1.1 State of the art

A number of approaches for formalization of business rules exist, but, as far as it is known to the authors of this paper, none of them use formalism of a matrix grammar.

To mention other approaches, in (Lovrenčić et al., 2006) authors present formalization of business rules based on ontology and UML modelling with the use of Object Constraint Language.

1.2 Outline

The first part contains information about decision support system and decision management system. In the next section will be stated how we can describe and extract business rules and what tools are used for their management and maintaining. Afterwards the method of business rule formalization by using decision table will be described. This method is base for a model of knowledge-based desicion support system. For the formalization is used matrix grammar. This section is followed by conclusion and the discussion about future work.

The result of this paper is design of model part of the knowledge-based decision support system model which will be described by the rules of formal grammar.

2 Decision Support System

Decision Support System (DSS) is a computer-based information system or subsystem that supports business or organizational decision-making activities. DSSs serve the management, operations, and planning levels of an organization and provides help in decision making

process. As a DSS is considered any computer application that enhances a person‘s or group’s ability to make decisions. Decision support systems can be either fully automated, human or a combination of both.

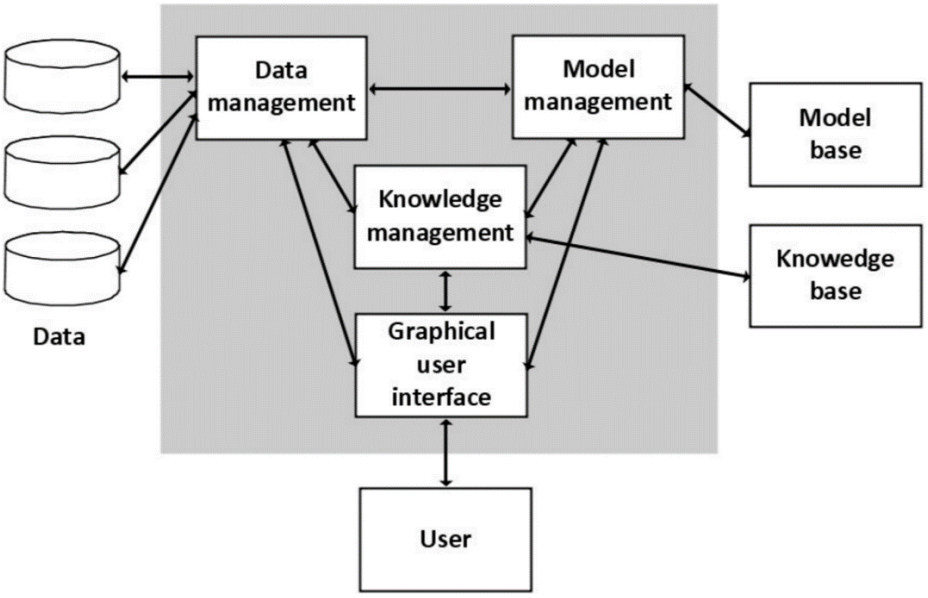

Fig. 1. Schema of a Decision Support System – based on (Jao (Ed.)., 2011)

In Fig. 1 can be found a schema of a decision support system. Main components of DSS are data, models for data processing and knowledge. Data can be external or internal including all information about processes that needs to be covered by DSS. Models upon the data are usually used for the accounting and financial analysis, simulation models and for the evaluation of the plans. Information supporting decisions can be input into the system also by user through an appropriate user interface.

DSSs are frequently mistakenly confused with decision management systems (DMS). The difference is that DMS makes decisions without further human interaction, but these systems make just decision, they are not responsible for the workflow. According to the James Taylor (Taylor, 2012) both DSS and DMS apply expertise and judgment, but DSS relies on the user to have experience and apply their own judgment. This means that decision support works better for strategic and management/control decisions where the user is likely to have some significant experience and that decision management is a better approach for operational decisions.

Decision Management Systems are built by focusing on the repeatable, operational decisions that impact individual transactions or customers. Once these decisions are discovered and modeled, decision services are built that embody the organization’s preferred decision-making approach in operational software components. The performance of these components, and the impact of this performance on overall organizational performance, is tracked, analyzed and fed back in order to improve the effectiveness of decision-making.

DSS model component is created by the defining of business rules which can be found within the business process. Information in this section was based on (Power, 2000), (Decision Management Solutions, 2016), (Debevoise, 2005) and (Taylor, 2012).

2.1 Classification of DSS

There are several classifications and taxonomies of DSS applications. Classification divide DSS into five main categories. Today‘s the most common and widespread DSS classification, popular with many authors, such as Power (Power, 2000) or Turban (Turban et al., 2008) is as follows:

- *Data-driven DSS*, which are primarily based on the data and their transformation into information. These systems usually analyze large amount of data, they support decision-making by allowing users to extract useful information. Data are collected in data warehouses for this purpose. Online analytical processing and data mining can then be used to process the data.
- *Model-driven DSS*, which puts the main emphasis on the use of simulation and optimization models. Early DSS system were mainly model based standalone systems.
- *Knowledge-driven DSS*, characterized by the use of knowledge technologies to meet the specific needs of decision-making process. Usually consits of knowledge about a particular domain.
- *Document-driven DSS*, that helps users acquire and process unstructured documents and web pages and thus provide complete document retrieval and analysis.
- *Communications-driven and group DSS*, which includes all systems built using communication technologies to support collaboration of user groups.
- *Hybrid DSS*, all above listed categories can be combined to create compound or hybrid systems.

Information based on information from (Jao (Ed.), 2011) and (Power, 2000).

Our designed system is classified as knowledge-based driven DSS, specificaly the model part.

3 Business Rules

Bussines rules should support the decisions of the business, not just describe the technical/fixed conditions in the system. The recognition of business rules is not done fully automatically. There are decisions that cannot be done without human interactions. But on the other hand there are a lot of situations, business opportunities or threats that can be detected by the use of historical data and known trends in data.

In (Halle, 2006), a way on how to recognize business rules within an organization is described. This recognition is split into several parts. Business analytics extract the rules statement in sentences of natural language during the first part. These sentences are then associated to a specific part of business process. During the next phase analysts transform these rules into more structured and detailed statement, eg. *condition-action* statements. Single rule statement can yield more condition-action rules. The last phase is to design and transform rules into highly structured executable rules. Any statement that enforces the relation between data is considered a business rule.

Business rules approach manages the flow of business process by using constraints or decision blocks. Business rules classify, compute, compare and control data to direct the flow.

Business rules patterns can simulate following types of events behavior:

- logical operations - AND, OR, NEG
- threshold patterns - triggers when a threshold value has been processed
- subset selection patterns - selection of significant rules in a set
- modal patterns - check if assertion is true

- time or spatial restrictions, eg. according to the spatial restriction, a credit card fraud can be detected

Note also that each business rule may be expressed in one or more formal rule statements, although each formal rule statement must be an expression of just one (atomic) business rule. A formal rule statement is an expression of a business rule in a specific formal grammar.

3.1 Business Rules Notation

Managing and modeling decisions is crucial for business. The DMN (Decision Model and Notation) standard emphasizes the importance of business decisions, and also offers a standard notation and expression for decision requirements and decision logic (Object Management Group, 2016).

3.2 Use of Business Rules

Typical use of these rules is in processes that contains easily automated actions that do not depend on other human interaction or opinion. Rules can be used for example in the algorithmic trading, for instance for the placement of the limit order.

Here are two examples of business rules in practice:

Execution of stock trading - limit order

- If the price is less than $20 (limit minimum price) BUY stock of COMPANY.
- If the price is at $35 (limit maximum price) or more SELL stock of COMPANY.

These two rules are displayed in Fig. 2.

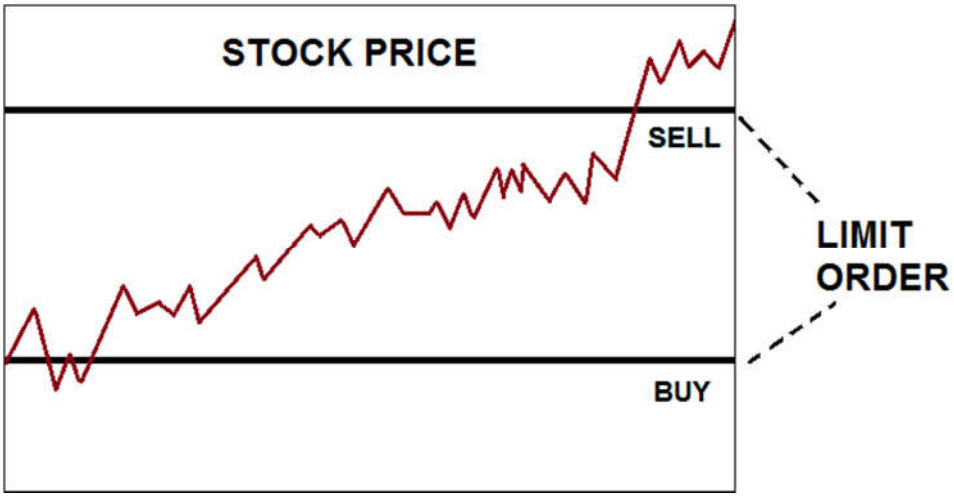

Fig. 2. Example of Limit Order

3.3 Decision tables

Decision tables are tool for:

- Capturing certain kinds of system requirements and knowledge.
- Documenting internal system design.

They are used to record complex business rules that a system must implement. In addition, they can serve as a guide to creating test cases. Decision tables represent complex business rules based on a set of conditions. The outcome might be multiple actions, not just one action.

The notation of decision tables says that the first column contains the labels of all evaluated facts and actions. Every other column contains one rule. Each cell in the table then indicates what value has the fact for given row or what action should be taken.

Decision tables are clear to understand both for the people who implement them into decision support system and for the analysts who are creators of the rules. The description of rules is declarative, it is a set of implication which are executed over the set of facts. This fixed order of conditions allows a complete overview of decision rules for a specific decision. It also allows grouping of related rules into tables, thereby providing an overview to a large number of decision rules. Decision table is one of the possible model for the decription of business rules for creating of DSS.

Trade decision table for the Buy order				
	Rule 1	**Rule 2**	**Rule 3**	**Rule 4**
Conditions				
Valid Symbol	No	Yes	Yes	Yes
Valid Quantity	DC	No	Yes	Yes
Sufficient Funds	DC	DC	No	Yes
Actions				
Buy?	No	No	No	Yes

Tab. 1. The example of decision table of the Placement of the Buy order – based on (Copeland, 2003)

The example in the Tab. 1 shows the condition and the rules for the Buy order. As we can see the only case when the buy order is placed is when there is *valid symbol* **AND** *valid quantity* **AND** *sufficient funds* available. Conditions that do not affect the outcome are marked "DC" for "Don't Care." So the Rule 1 indicates that if the Symbol is not valid, ignore the other conditions and do not execute the Buy order.

Decision rules dictionaries are other way how to describe rules in DSS. Generally, the dictionary of rules is a set of facts, global values and functions. Another way of describing of business rules is the enumeration of *IF condition THEN action*. The condition part of the rule is in the form *fact operator value* and that is that the fact is tested if it is equal, less or greater than the given value. All described ways are based on the form which is clear for both technical and nontechnical business people.

3.4 Implementation of Business Rules

The configuration of the decision support by business rules can be realized by some existing tool or framework. To mention the most used frameworks we name JBoss Rules/Drools, Jess, or ILOG JRules. The idea of decision tables is directly implemented in the Open Rules framework (Open Rules, 2016). Open Rule supports the OMG standard for business rules notation – DMN. This framework uses excel sheet for definition and maintenance of the set of business rules. The set of rules can be updated also at the run time. OpenRules offers an enterprise level of business rules repository implementation. Its advantage in comparison with other tools listed above is that this framework is available as open source.

4 Formalization of Business Rules

Number of approaches for formalization of business rules exist, but, as far as it is known to the authors of this paper, none of them use formalism of a matrix grammar. In the designed approach we want to take advantage of main characteristics of matrix grammars which is the generative power and ease of use. User may update the set of rules as required without the need of third party to control the decision making process. CEP decision making is stateful, so we use the information from the previous states.

4.1 Matrix Grammar

Formal grammars can be used for description of behavioral patterns and set of business rules extracted by CEP and for prediction of data in CEP platforms. Briefly, a formal grammar is a set of rules for rewriting strings, along with a "start symbol" from which rewriting starts. Matrix grammar belongs to the group of regulated rewriting grammars. For further reading about this topic authors recommend (Rozenberg et. al, 1997).

4.1.1 Definition of matrix grammar

Matrix grammar is a pair $H = (G, M)$, where $G = (N, T, P, S)$ is context-free grammar and M is finite language over P, $(M \subset P)$ - sentence of this language is called **matrix**.
Formally, a matrix grammar is a pair $H = (G, M)$, where

- $G = (N, T, P, S)$ is a context-free grammar, where:
  - N is an alphabet of nonterminal symbols
  - T is an alphabet of terminal symbols
  - P is a finite set of rules, $P \subseteq N \times (N \cup T)$
  - S is starting symbol, $S \in N$
- M is a finite language over P, $(M \subset P)$ - a sentence of this language is called a matrix.

Further, for $u, v \in (N \cup T)$, $m = p_1 \dots p_n \in M$ we define $u \Rightarrow v\,[m]$ in H, if there are strings $x_0 \dots, x_n$ such that $u = x_0, v = x_n$, and for all $0 \leq i < n, x_i \Rightarrow x_{i+1}\,[p_{i+1}]$ in G. The language generated by H, denoted by $L(H)$, is defined as $L(H) = \{w: w \in T, S \Rightarrow w\}$.

Even thought that matrices contain only *context-free* rules, they may generate the *context-sensitive* language.

4.2 Formalization of Business Rules by Using Matrix Grammar

Input: Business rules in various forms. Business rules can be in the form of decision tables, enumeration of condition-action rules or sentences of natural languages. Form of business rules is discussed above. In this example we will refer to the general form of decision table above – Tab. 1.

Output: DSS described by the business rules in the form of matrix grammar $H = (G, M), G$ is quadruple (N, T, P, S)

Method: $G = (N, T, P, S)$, where:

$N := \{Action_1, Action_2, \dots, Action_n\}$

$T := \{condition_1, condition_2, \dots, condition_m, action_1, action_2, \dots, action_n\}$

$P := N \times (N \cup T)$

for each $Rule_p$, $\{Rule_1, Rule_2, \dots, Rule_p\}$ from the decision table consider all suffice conditions, the set $\{Condition_1, Condition_2, \dots, Condition_m\}$ and do:

1. add rule $p, p \in P: S \rightarrow < Rule_1, Condition_1 >< Rule_1, Condition_2 > < Rule_1, Condition_m >$
2. add rule $p, p \in P: S \rightarrow < Rule_2, Condition_1 >< Rule_2, Condition_2 > < Rule_2, Condition_m >$
3. ...
4. add rule $p, p \in P: S \rightarrow < Rule_p, Condition_1 >< Rule_p, Condition_2 > < Rule_p, Condition_m >$,
5. add $< Rule_p, Condition_m >$ to N; m, p are positive integers.

For each $< Rule_p, Condition_1 >$ add rules:

6. $< Rule_p, Condition_1 > \rightarrow Action_n condition_1$
7. $Action_n \rightarrow action_1 action_2 \dots action_n$, where $action_n$ are all actions taken after fulfilling of all sufficient conditions for the $Rule_p$.

For each $< Rule_p, Condition_m >$ for $m \in N, m > 1$ add rules:

- $< Rule_p, Condition_m > \rightarrow condition_m$

$S := S$;

$M := \{m_1, m_2, \dots, m_p\}$, where $m_p = [< Rule_p, Condition_1 > \rightarrow Action_n condition_1,$
$< Rule_p, Condition_m > \rightarrow condition_m$ for all m > 1]

Component M is usually created by business analyst by determining parallel actions. Only the actions that leads to the execution of actions are added to matrices. In this case the matrices are determined by grouping of all conditions into matrix and all actions in one matrix. All rules in each matrix have to be taken in one computational step.

4.3 Case study

This section is devoted to an example of the use of the designed model on the execution of the buy limit order. Buy order is executed only if all conditions are met. Given case is already described by the decision table Tab.1. described in section 3.3 above.

Case description:

Placement of the buy (limit) order. Generally buy limit order is an order to purchase a security at or below a specified price. A buy limit order allows traders and investors to buy a security with the restriction on the maximum price to be paid or to sell a security with the restriction of the minimum price to be received. By using a buy limit order, the investor is guaranteed to pay that price or lower. The order will only be executed at a specified limit price or better but there is no guarantee that the order will get filled in the first place.

The construction of the rules set described by matrix grammar $H = (G, M), G = (N, T, P, S)$ for the table Tab.1. above will be as follows:

$N := (DONT\_BUY, BUY, S)$

$T := (valid\_symbol, valid\_quantity, sufficient\_funds, buy)$

$P:=(S \rightarrow < R1, C1 >< R1, C2 >< R1, C3 >, \quad S \rightarrow < R2, C1 >< R2, C2 >< R2, C3 >,$

$S \rightarrow < R3, C1 >< R3, C2 >< R3, C3 >, \quad S \rightarrow < R4, C1 >< R4, C2 >< R4, C3 >,$

$< R1, C1 > \rightarrow DONT\_BUYvalid\_symbol, < R2, C1 > \rightarrow DONT\_BUYvalid\_symbol,$

$< R3, C1 > \rightarrow DONT\_BUYvalid\_symbol, < R4, C1 > \rightarrow BUYvalid\_symbol,$

$BUY \rightarrow buy, DONT\_BUY \rightarrow \varepsilon,$

$< R1, C2 > \rightarrow valid\_quantity, < R1, C3 > \rightarrow sufficient\_funds,$

$< R2, C2 > \rightarrow valid\_quantity, < R2, C3 > \rightarrow sufficient\_funds,$

$< R3, C2 > \rightarrow valid\_quantity, < R3, C3 > \rightarrow sufficient\_funds,$

$< R4, C2 > \rightarrow valid\_quantity, < R4, C3 > \rightarrow sufficient\_funds)$

Where $< Rp, Cm >$ denotes nonterminals $< Rule_p, Condition_m >$ (according to the method in formalization process described above), S denotes the starting nonterminal.

After the addition of nonterminals to the set of nonterminals N, N will look like:

N:= $(DONT\_BUY, BUY, S, < R1, C1 >, < R1, C2 >, < R1, C3 >, < R2, C1 >, < R2, C2 >, < R2, C3 >, < R3 >, < R3, C2 >, < R3, C3 >, < R4, C1 >, < R4, C2 >, < R4, C3 >)$

M:=[$< R4, C1 > \rightarrow BUYvalid\_symbol, BUY \rightarrow buy, < R4, C2 > \rightarrow valid\_quantity,$

$< R4, C3 > \rightarrow sufficient\_funds$]

Matrix M was determined on the basis of execution of buy order, which is executed only for the $Rule_4$only and if only all the conditions are met.

Conclusion

The goal of this paper was to design a model part of knowledge-based decision support system. This category of DSS helps users in problem solving, it usually transforms knowledge into rules. Rules specifying knowledge may be represented by several ways - in textual or graphical way or in a table. We chose to describe the rules by the rules of formal grammar, the rules are defined by using matrix grammar. For the clarity the business rules can be written in the form of the decision table.

The formalization of business rules is intended to be used for the formal verification of complex event platforms. According to the prediction for CEP trends (Luckham, 2012, p. 93) this area is still not fully explored. The trend in research of complex event processing will be aimed to next generation language for specifying complex event patterns and pattern triggered reactive rules.

The advantage of presented concept of business rules description is in the use of matrix grammar. This grammar belongs to the group of grammars with regulated rewriting. This regulation allows us to define the dependencies between the rules and thus to regulate the use of business rules. Matrices group the rules which have to be processed in one computational step.

The next step is to implement decision support system as a module to complex event platform. This module will use both historical and actual data and thus will support the

decision making. DSS may have capabilities to discover, describe and predict knowledge hidden in data relations and patterns. Presented solution will benefit to the decision making process in complex event processing and also to the formal verification of CEP systems.

The further work should be focused on testing of the described approach. There should be tested mainly the performance demands and the requirements for real time processing.

The future research and development will be focused on creating algorithmic model designs to effectively process high volume market data. We will use advantage of CEP platform for its processing in real time and we will integrate business rules to control the flow of events prediction.

5 Acknowledgment

This work was partially supported by the BUT FIT grant FIT-S-14-2299, "Research and application of advanced methods in ICT".

References

Copeland, L. (2003) *A Practitioners Guide to Software Test Design*: London, UK: Artech House, 320 p., ISBN 978-1-58053-791-9.

Debevoise, T. (2005) *Business Process Management with a Business Rules Approach: Implementing The Service Oriented Architecture*: Canada, Business Knowledge Architects, ISBN 978-1419673689.

Decision Management Solutions (2016, February 29), *Decision Management Solutions Blog*. Retrieved from http://www.decisionmanagementsolutions.com/.

Jao, C. (Ed.) (2011) *Design and Development of a Compound DSS for Laboratory Research,* in: *Efficient Decision Support Systems - Practice and Challenges From Current to Future*, vol. 1, InTech, Rijeka, 556 p., ISBN 978-953-307-326-2.

Von Halle, B. (2006) *The Business Rule Revolution: Running Business the Right Way*: Cupertino, CA: Happy About, xxviii, 291 p. ISBN 978-1- 60005-013-8.

Lovrenčić, S., Rabuzin, K., Picek, R. (2006) *Formal Modelling of Business Rules: What Kind of Tool to Use?*: Journal of information and organizational sciences, Volume 30, Number 2.

Luckham, D. (2012) *Event Processing for Business: Organizing the Real-time Enterprise*: Hoboken, NJ: Wiley, xiii, 273 p. ISBN 978-0-470- 53485-4.

Luckham, D. (2002) *The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems*: Boston, USA: AddisonWesley Professional, xiii, 400 p. ISBN 978-0201727890.

Object Management Group (2016, May 4), *Decision Model And Notation (DMN)*. Retrieved from: http://www.omg.org/spec/DMN/.

Open Rules (2016, May 13), *Open Rules*, Retrieved from: http://openrules.com/.

Power, D. J. (2000) *Decision Support Systems Hyperbook*, Cedar Falls, IA: DSSResources.COM. Retrieved from: http://dssresources.com/dssbook/.

Rozenberg, G. Salomaa, A. and (editors) (1997), *Handbook of Formal Languages Vol. 2 Linear Modeling: Background and Application*: Berlin: Springer, ISBN 3- 540-60648-3, 528 p.

Taylor, J. (2012) *Decision Management Systems: A Practical Guide to Using Business Rules and Predictive Analytics*: Upper Saddle River, NJ: IBM Press, xiii, 284 p. Information management. ISBN 978-0-13- 288438-9.

Turban, E.; Aronson, J. E.; Liang, T. & Sharda, R. (2008). *Decision Support and Business Intelligence Systems (eight edition)*, Pearson Education, ISBN 0-13-158017-5, Upper Saddle River, New Jersey.