Professor Dorin ZAHARIE, PhD
Professor Irina Bogdana PUGNA, PhD
Lecturer Cristina RADULESCU, PhD Candidate
The Bucharest Academy of Economic Studies
E-mail: zaharied@ase.ro

AN ONTOLOGY-BASED CONCEPTUAL DESIGN OF A DATA WAREHOUSE

Abstract. *Dimensional modeling, which is critical to development of decision support systems (DSS) centered on data warehouses, may be conducted in a data-driven, or requirements-driven manner. This paper proposes an approach directed at user requirements specification and creation of the dimensional model which captures them, based on the assumption that any decisional analysis relies on a conceptual framework that basically abstracts the particular manner in which the decision-maker perceives the reality of the organization and its environment. Using resource-event-agent (REA) enterprise domain ontology as a foundation, decision-maker's requirements are defined and represented as an application ontology scheme, structured on operational and policy levels. Following the design guidelines proposed and illustrated by the paper, an initial dimensional model is generated from the application ontology scheme and then it is later refined through validation with system stakeholders and confrontation with available data sources.*

Key words*: data warehouse, dimensional modeling, REA model, domain ontology, application ontology.*

JEL Classification: L86, M10, O33

1. Introduction

In the last ten years, data warehousing has become a subject of large interest, which can be explained through the increasing need for computer-assisted business decisions. In essence, a data warehouse is a repository used for collecting relevant information for the management of an organization [12]. Designing a data warehouse is a major undertaking, for at least two principal reasons: on one hand, it involves usage of a great range of components, with various purposes, functions and technologies and, on the other, it is directed at an area – business decision – much more complex and less likely to be formalized than others addressed by transactional information systems.

There are two common approaches to dimensional modeling of data warehouses: data-driven, in which case the starting point is represented by available data sources, and requirements-driven, when all starts by identifying information needs to be catered for by the data warehouse.

In order to define information needs is common practice to resort to goal and scenario modeling techniques. Another approach relies on the idea that the purpose of a data warehouse is to provide business performance measurements and therefore, data warehouse modeling should start from business process models [2].

The current paper advocates the necessity of finding a direct and accessible manner to help decision-makers define and clarify their requirements relative to a data warehouse system. The approach presented by this paper is based on the assumption that any decisional analysis relies on a conceptual framework, which involves a certain core internal logic, which actually is an abstraction of the particular manner in which the user perceives the reality of the organization and its environment. Analysis methods and techniques may change and evolve in time, without impacting on this basic structure. As a consequence, using this "native" conceptual framework creates the fundaments for building a data warehouse that is much more flexible and responsive to the information needs produced by the decision-making process. This does not imply that the data warehouse does not evolve, but that it changes within the same core structure, which remains stable.

Any assessment or decisional evaluation should be performed, by taking into account, in an explicit or implicit manner, an assembly of conceptual *things*, that can be represented through the means of an ontology. Given that such an ontology focuses on a certain area within the activity of an organization – the problem domain – its description represents an *application ontology.* Application ontology is a specialization of a broader conceptual framework - *domain ontology*. The presentation that follows adheres to this particular perspective.

By using enterprise domain ontology as a foundation, the requirements of decision-makers are defined and represented as an application ontology scheme. On the basis of generic guidelines within domain ontology, an initial dimensional model is generated from the application ontology scheme that later has to be refined through validation with system's stakeholders and confrontation with available data sources.

2. Dimensional modeling concepts

The key concepts of dimensional modeling are *facts* and *dimensions*.

Facts are described through a set of quantifiers or indicators, perceived as relevant by the decision-making people and designated as measures. Measures typically are represented by numeric values. Nevertheless, non-numeric values are accepted, as long as they are strictly ordered. At the same time, facts with no measures *(factless)* are also accepted, when the mere occurrence of such facts is considered relevant.

Dimensions define perspectives from which facts can be analyzed. The level on which instantiations of facts are directly attached to dimensions is the one of the greatest detail relative to information provided to decision-makers and is known as *base* or *terminal level*. Up from this level, new aggregation levels can be defined for each dimension, and in what follows they are called *dimensional levels*. These levels are organized in *hierarchies*. The hierarchies are those that guide operations in aggregation of measures (*rollup*) or in providing more detailed data (*drill-down*). It is worth mentioning that for the same terminal level several alternative hierarchies may be defined, each one with its own aggregation levels. Instantiations of levels within a dimension are known as *dimension members*.

As a consequence, the concept of dimension brings together data with common semantics for the reality that is subject to modeling. From a structural perspective, a dimension consists of a single level or it may consist of one or several hierarchies [13].

When defining classification hierarchies, *strictness* and *completeness* constraints must be met [11]. The former requires that, within the same hierarchy, any member on a certain level should correspond to a single member on the immediately-above level, whereas the letter states that the member on the highest level should include all members on levels below, and those members only.

Each level of a dimension must possess one or several attributes which offer a certain identity to its members and also describe them.

Both measures and descriptive attributes may be *atomic* or *derived* by using certain *derivation rules*.

On a logical level (or, more accurately, on a relational level), the structural configuration consisting of a fact associated with several dimensions is designated through the concept of *star*. Some other possible configurations are *snowflake* and *constellation*. Due to the lack of largely accepted equivalent terms on a conceptual level, the concept will be used in the present paper with the meaning of a *conceptual star*, in order to reference this specific structure as a information unit, while ignoring any detail related to representation on logical and physical levels.

In order to analyze business activity, even on a limited area, a single star generally is not enough, which poses the problem of relating various stars. Such relations are based on common dimensions, which leads to a configuration designated as constellation. Common dimensions to be found in such structures are known as *conformed dimensions* [10]. In practice, there are also situations when connections provided by dimensions (perceived as perspectives used for data analysis) are not enough. In order to accommodate such situations, another type of dimensions, called *derived dimensions*, must be used to show the occurrence of facts. Each member of such dimensions which resides on the leaf level identifies the business transaction which triggered the instantiation of a certain fact. As a result, it is possible, for instance, to produce analyses which correlate customers orders and shipping of orders. There is a strong resemblance to the concept of *degenerated dimension* [10], except that, in order to clearly state from a conceptual perspective the need to access the identity of a business transaction, this paper treats it as a distinct dimension.

The fundamental structure which enables exploitation of data is the hypercube, often simply referred to as *cube*, in case of which dimensions correspond to axes and facts correspond to cells. There are two types of operations that can be performed on cubes: those acting upon the subject of the analysis (facts, cells, measures), and those acting upon the perspective on data. The former type include *Drill-across* and *Projection* while *ChangeBase*, *Roll-up* and *Dice* belong to the latter. *Slice* may be derived from this set of operations. As for *Drill-down*, this operation cannot be performed unless *Roll-up* was previously performed; as a consequence, *Drill-down* is the reverse of *Roll-up* [7]. In Abelo (2006) it is demonstrated that the cube algebra composed by these operations is closed, complete and minimal. The same paper also presents inter-cubes operations: *union and intersection*, provided that cubes are defined on the same domain (n-dimensional space) and *union over domains*, in case of multiple domains.

What this paper is actually focused on is *Drill-across* operations, which allow changing constituent cells while preserving the n-dimensional space unaltered, which involves connecting instantiations of a fact to those of another. Taking into account the fact that a star is defined on a single fact, this makes possible the correlation of contents of several different stars.

3. REA as a business domain ontology

The ontology, a concept from philosophy where it forms, together with the epistemology, branches of metaphysics, has lately become a subject of most intensive interest, especially because of the research on "contents" management. In this respect, the definition provided by Gruber [6] "*an ontology is an explicit specification of a conceptualization*", which is one of the most often quoted, was completed by two remarks: specification must be formal, meaning that it must be handled by a computer, and conceptualization must be shared, which means it must agreed-upon by a community.

An information system always integrates a certain comprehension of the data to be processed and it is essential that this conceptualization is identical to that of the system's users. Such harmonization can be sensibly facilitated by predefining domain-specific ontologies, which may be shared by system modelers and users, during the system development process. A *domain-specific ontology* describes the concepts used in a particular field, their classification, relationships and axioms [3]. Within the realm of a domain-specific ontology, an *application ontology* will be defined for any system that must be developed by attaching some additional elements that are system-specific.

REA - Resource, Event, Agent – was initially presented as semantic data model for accounting [14]. Through later research, it was extended and analyzed as a domain-specific ontology [4]. According to this model, all exchanges and economic conversions which constitute the activity of a business follow a common pattern: "*There is a transaction (an economic event) where an internal agent (an economic unit or agent) gives something of value (an economic resource) to an outside person (an economic agent); this decrement event is always paired with a mirror-image increment event where the internal agent receives in kind another type of economic resource which has more value to the enterprise in its pursuit of its entrepreneurial goals.*" [4 p.3]. In this model, the event-resource relationship is called *stockflow*, the relationship between two agents and an event is called *control*, whereas the relationship between an event and its peer, *duality*. In addition, the model includes a reflexive relationship on the economic unit, called *responsibility*, that can be used to represent the hierarchy level, or the authorization and control level associated with event occurrences. Though it originates in accounting, this model manages to decouple the exact expression of economic events and their technical treatment in accounting.

Later research have confirmed REA as a business domain ontology, which has four major applicability areas: model-driven design, supply chain collaboration, knowledge representation, education [3]. The OWL code of the REA ontology is publicly available and it makes use of the previously introduced terms, except for a single difference: the dichotomy of internal-external economic agent is replaced by that of the agent providing or receiving, relative to an event.

3.1. Operational level

Operational level corresponds to the actual occurrence of economic events. Figure 1 offers a UML representation of the primitives in the REA model: *economic resource*, *economic event* and *economic agent*. The *stockflow* relationship has attributes, which indicate the type of the flow – *inflow* or *outflow* – and its quantification. Hence, it was represented as a association-class. According to the multiplicities in the diagram, an instantiation of *economic event* affects a single instantiation of *economic resource*; this remark is extremely significant from the perspective of dimensional modeling.

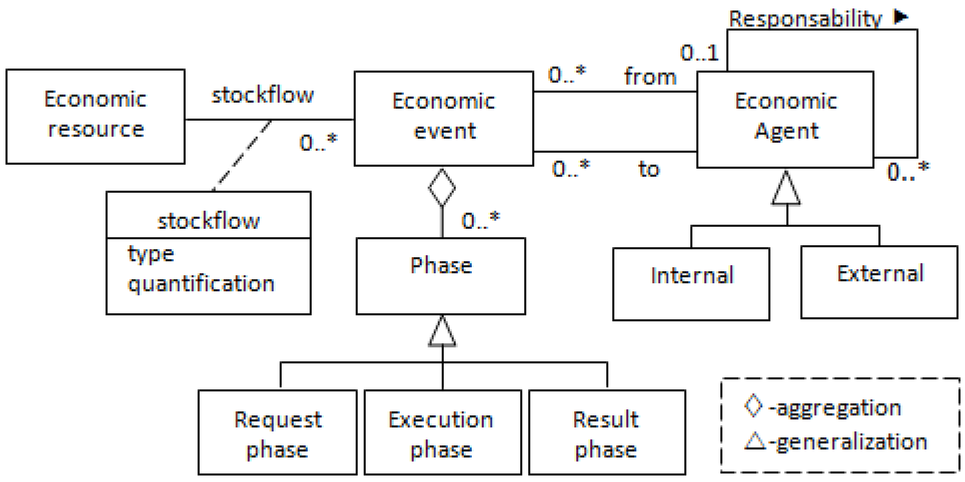


Figure 1. A UML class diagram for operational level

One instantiation of *economic event* involves two instantiations of *economic agent* represented through *from* (*provide*) and *to* (*receive*) roles. Unlike the original model, in this case, the *responsibility* relationship might exist for any agent, regardless of the role it plays. Both roles may be acted by internal as well as external agents (*economic units*, in original terminology).

An e*conomic event* may be perceived as a whole, or it may be decomposed in multiple phases, which is represented through the aggregation symbol and its multiplicity. These phases fall into three categories: *request*, *execution*, *result.* This classification requires some additional comments.

The typology of the phases suggests event sequentially, in the sense that an economic event is a *request* or *promise* but still unfulfilled, is *executing*, and finally, completed and *accepted*. In this manner, the analysis may include business process execution, starting from the decision which triggered the process (*request*) until its completion (*result*). This functionality is operational and, possibly, still interesting, even on the maximum level of granularity. For example, selling a product to a certain customer is an economic event. The customer's order which triggered the process of selling is the first phase of the event (request); successive partial deliveries of products may be mapped to the next phase (execution), whereas the completion of delivery (result) is the phase when the actual occurrence of the event may be conceptually confirmed. Note that for a certain economic event, these kinds of phases occur in similar combinations of economic resources and economic agents. As mentioned before, this particular decomposition of events is not mandatory: it is up to the user to decide whether such decomposition is useful, or the image of the event, as a *thing* that occurred, is actually good enough.

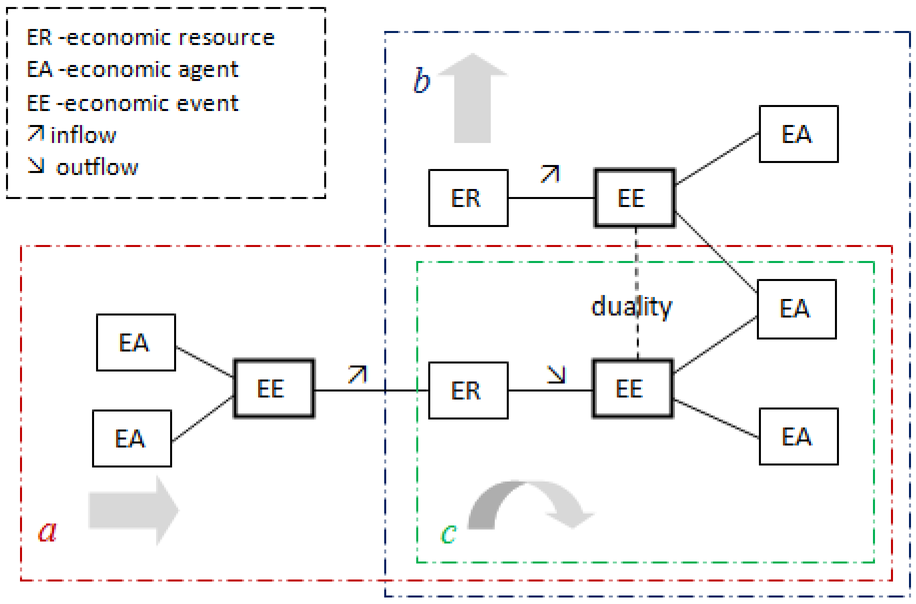


Figure 2. Three ways to determine the business process to be modeled

There are three axioms defined for the REA ontology: *stockflow axiom*, *duality axiom* and *participation axiom*. *Stockflow axiom* postulates that for each resource, there is at least one *inflow event* and one *outflow event*. This approach leads to a certain type of representation and analysis, which is focused on resources (case *a,* in Figure 2). *Duality axiom* formulates duality of events, in a manner which is less straight-forward than in the original model: "*all events effecting an outflow must be eventually paired in duality relationship with events effecting an inflow and vice versa*" [3 p. 243]. This leads to another approach to representation and analysis, which is based on enterprise's *value chain* (case *b,* in Figure 2).

Having added case *c* to the above mentioned, the specifics of which is that an economic event is treated in stand-alone manner, according to Figure 1, we end up with a typology for identification and representation of business processes targeted by the data warehouse.

Relative to membership of participants, there are two kinds of economic events that can be inferred: *exchange* and *transformation*. In case of *exchange economic events,* one of the economic agents is from outside the enterprise. A *transformation economic event* occurs between internal economic agents. Purchasing of materials and selling of products are examples of exchange economic events, whereas consumption of materials in order to make final products is an example of a transformation economic event. *Participation axiom* accommodate these differences and postulates that for exchange events only it is necessary to instantiate both the inside and outside subsets.

3.2. Policy level

In order to assess how business processes are performed, it is necessary to extend the model so that pre-calculated or anticipated quantifications of process activities are also captured. Building this level requires two major abstraction mechanisms - *typification* şi *grouping* – applied on the operational level. As a result, cognitive level structures called *type images* are being defined. Type images create a policy infrastructure, which can be used to specify standards, policies and budgets [5].

Typification captures hierarchical relationships between types and subtypes produced by generalization. Typification enables classification of resources, events or agents, which represent the object of constraints and guidelines decided by enterprise management in the course of day-to-day operation of the business. *Grouping* defines collections of elements that carry a particular meaning for the enterprise management. For example, in case of a passengers transport company, vehicles, which all have features like engine series, registration number, acquisition date are classified according to their model or type. Each model is defined through its name, maximum number of seats, and standard fuel consumption. The values of these properties for a certain model represent a definition which is applicable to any of the vehicles with that typed, owned by the company. The company groups the vehicles in fleets, depending on the territorial unit which manages each vehicle. A fleet is characterized by name (identical to that of the territorial unit), number of vehicles and the maximum daily transport capacity. In grouping, derived attributes are particularly important; in our example, all attributes of a fleet are derived. It is obvious that typification and grouping relationships are distinct and the same element – the vehicle, in our example – is involved in each of them.

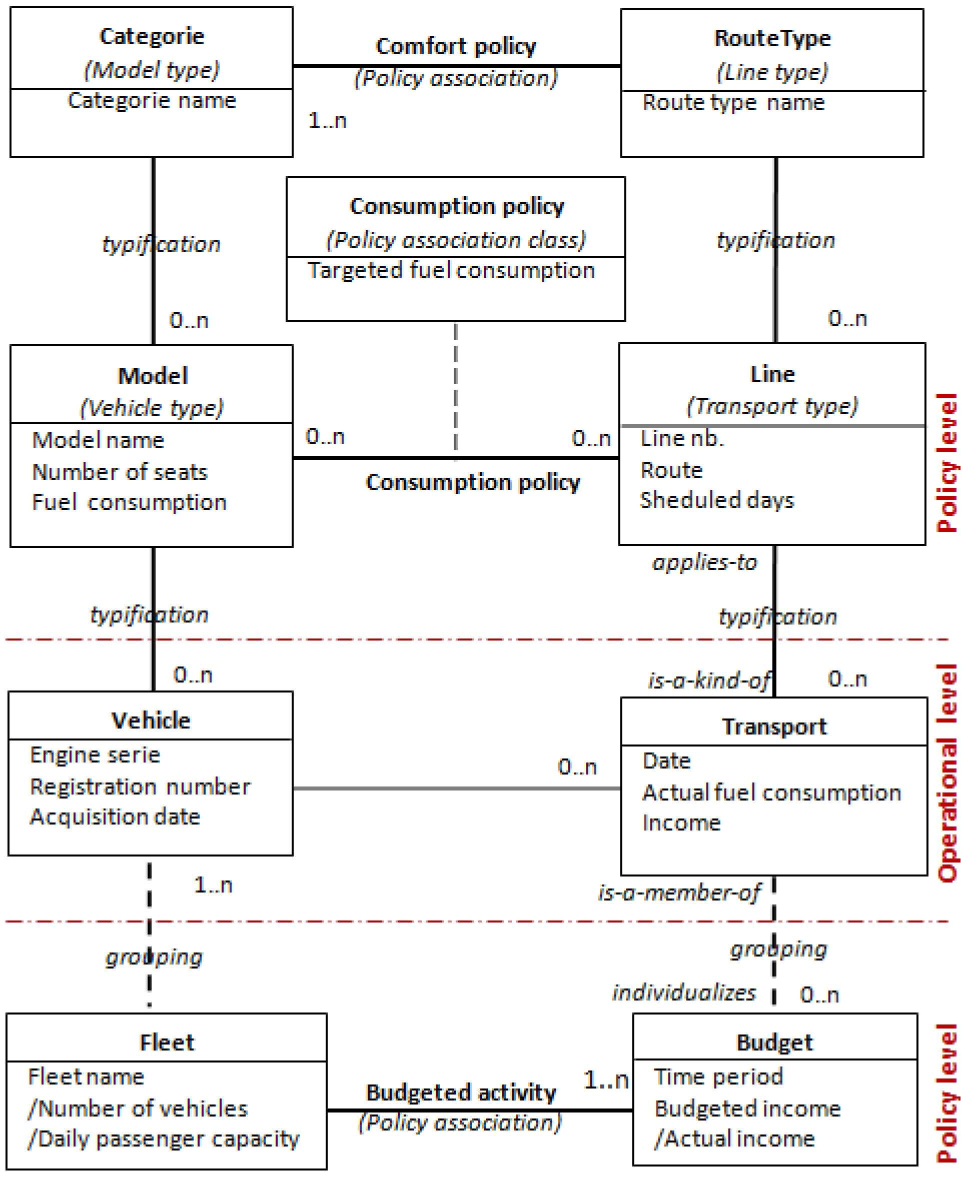


Figure 3. **Example of policy association and policy association class** (adapted from [5])

There are three kinds of policy definition: *knowledge-intensive description*, *validation rules* and *target descriptions*. Their implementation on the operational level in realized through *inference*, in case of knowledge-intensive description, through *validation,* in case of validation rules and through *discrepancy analysis,* in case of target descriptions. Given the scope of present paper, only target description is referred to. It frequently presents itself as a *standard* or *budget* (or budget element).

Relative to the types and groups on the policy level and their relationships, there are three specific ways of creating a *policy definition*, as they rely on values of attributes, or relationships, or their combination (through associations with attributes - UML association-class). In order to illustrate this, the same example presented above, will be used again in what follows.

The transports are operated regularly, on predefined routes, according to a certain schedule which specifies days of the week as well as departure and arrival hours. The routes are classified as local, inter-county and international. Also, the vehicle models are classified according to the comfort level, into three categories: premium, standard, economy.

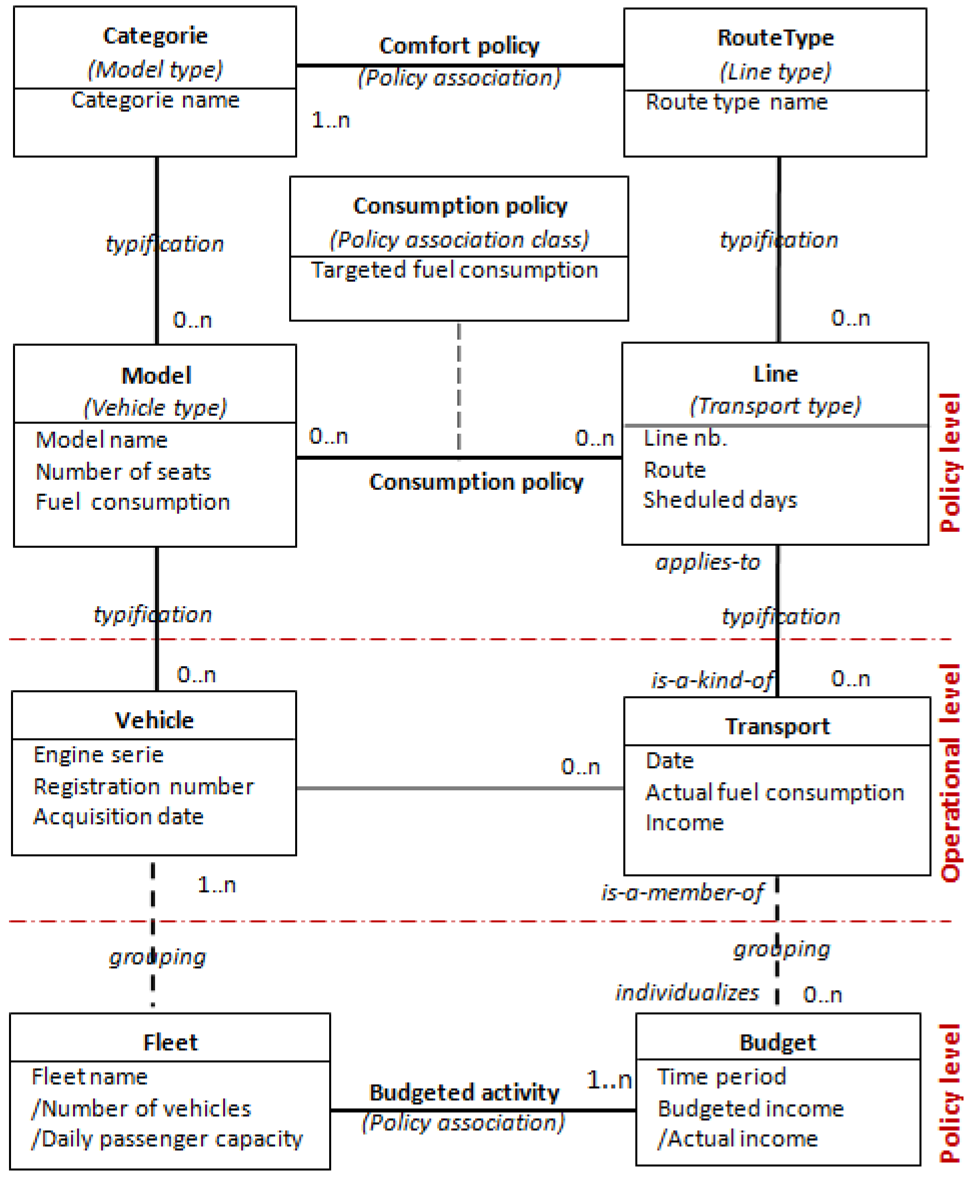


Figure 3 captures a simplified REA policy model, represented in UML. *Vehicle* and *Transport* are abstracted through both typification and grouping. In case of typifications, a particular layering may be noticed: the first layer hosts *Model* (vehicle type) and *Line* (transport type) whereas the second, *Category* (model type) and *RouteType* (line type). Typification is transitive and, according to the actual modeling needs, there may be used as many layers as necessary. Is-a-kind-of and applies-to associations for *RouteType* were explicitly marked in the diagram; they actually operates for all typifications, but they were left out of the diagram in order not to impair its readability.

In case of vehicles, grouping shows their condition of being part of a certain fleet, as stated before. Transports are also subject to grouping, depending on fleet and budgeting period. Each of such groups define, for the associated fleet and period, the *budgeted income* and *actual income*, with the latter being a derived attribute. One last remark: although typifications and grouping could be represented by making use of the UML symbols for inheritance and aggregation, as they carry a rich meaning, a simpler representation was preferred in order to avoid confusion. At the same time, placing typifications and grouping in distinct areas is not a rule: the reason it was used here is to facilitate understanding of the model.

In this particular example, policy definition is presented in two forms: association and association class. Differentiation of vehicle categories by *route type* is the expression of a standard and is represented through an association (*Comfort policy*). *Consumption policy*, which relates *Model* and *Line*, is represented as an *association-class*, whose attribute indicates *targeted fuel consumption* for a certain model and a certain line. *Actual fuel* consumption may be subject to a discrepancy analysis relative to the targeted fuel consumption for a particular vehicle type used on a specific date. *Budgeted activity* is also a policy definition represented as an association between Fleet and Budget groups.

4. Ontology-based dimensional design guidelines

By confronting this ontological framework with the concepts of dimensional modeling, the present paper offers a set of guidelines for dimensional modeling, which, applied on an application ontology scheme, should enable rapid articulation of a data warehouse structuring solution. Some of these guidelines might also be considered in case of a semi-automated approach, especially when computational ontologies are being used.

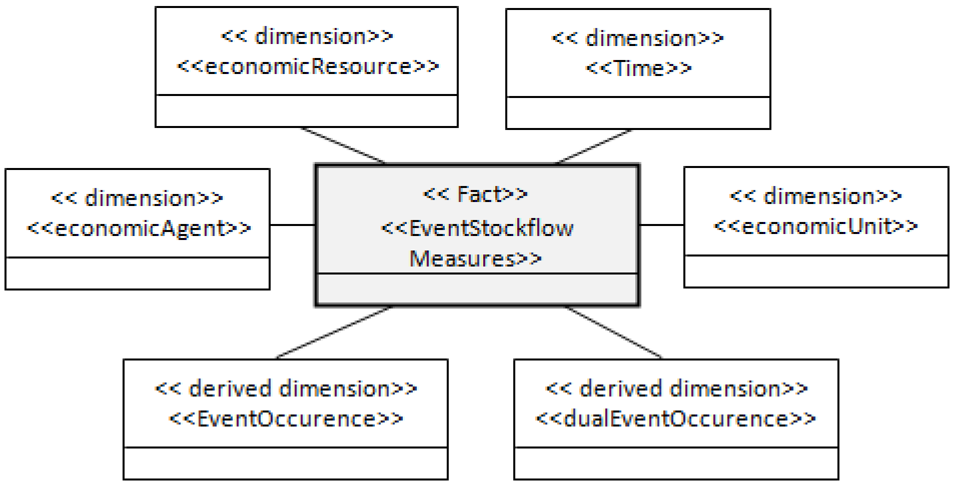
4.1. Facts and dimensions

The guidelines for identification of facts and dimensions are based on the diagram in Figure 4, where correlations are specified through pairs of UML stereotypes. There are two situations that may be identified, depending on whether phasing of economic events is taking into account or not.

In the latter case, the following guidelines may be employed:

a. each economic event result into a (conceptual) star;
b. economic resource, economic agent, economic unit and, implicitly, time, are the dimensions of that star;

c. the attributes in the stockflow association represent the measures of the fact;
d. economic event, as an occurent thing, forms a dimension which serves as a link to the dual event;
e. responsibility association introduces a dimensional level for economic unit.



<< dimension>> <<economicResource>>
<< dimension>> <<Time>>
<< dimension>> <<economicAgent>>
<< Fact>> <<EventStockflow Measures>>
<< dimension>> <<economicUnit>>
<< derived dimension>> <<EventOccurence>>
<< derived dimension>> <<dualEventOccurence>>

Figure 4. Dimensional design guidelines

When phasing of the *economic event* is required, two alternative path may be followed:

i. in case of one-to-one relationship between phases, phases will be distinctly represented as states, within facts;

ii. in the opposite situation, each phase results into a star configuration as stated above, whereas phases are correlated through derivated dimensions of facts.

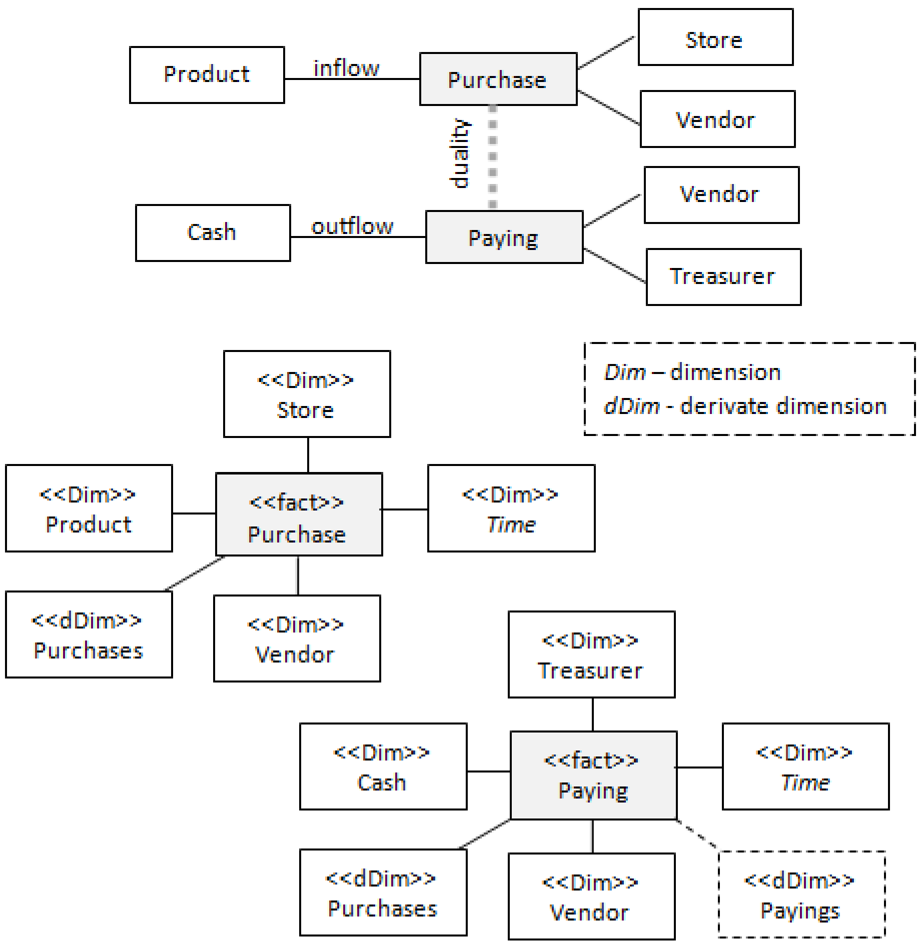


Figure 5. Purchase REA model and corresponding dimensional schemata

For the sake of clarity, Figure 5 resumes the example on purchasing, presented in [14]. The two *economic events* engaged in a *duality* relationship are *Purchase* and *Payment* (*Cash disbursement)*. According to the guidelines presented above, the associated dimensional model consists of two stars (Figure 5). In the first one, a *Purchase fact* includes measures which quantify the product inflow, usually, quantity and amount. There are five dimensions: for the economic resource (*Product*), for the two agents (*Vendor, Store*), for the economic event (*Purchase*) and, finally, for *Time*.

The second star includes paid amounts in *Payment fact*. The number of dimensions now raises to six, as to those symmetrical to dimensions in the first star, a new one was added, *Purchases* dimension, in order to relate each payment to each purchase. The configuration of the model allows representation of situations when each payment is associated with a single purchase; in different words, Payment and Purchases are related through a one-to-many type of correspondence. If the same payment may occur for multiple purchases, the solution of degenerate facts [11] should be applied, namely insertion of a bridge between fact and dimension, meant to transform the many-to-many correspondence into two many-to-one correspondences.

At this point, some additional comments are required. Firstly, apart from what was discussed here, the REA model also includes a responsibility relationship between the internal economic agent and the unit which is its hierarchical superior. Normally, its representation leads to a dimension related to the subordinate one, thus ending up, even from a conceptual perspective, in a snowflake configuration. An alternative approach would be to add a completely new dimension, to the existing ones. The conclusion reached after reviewing multiple case-studies by various authors from a REA-approach perspective is that the second solution is the most frequent, in spite of direct determination between dimensions in question, emphasized by [1].

Secondly, the presence of derived dimensions in stars should be approached from a mere conceptual perspective. There may be considered several solutions for their representation on a logical or physical levels, but they are irrelevant in the context of this paper. The need to correlate facts in multiple stars whenever the decision-making process requires was pointed out in literature [8], [16] and there are several authors who expanded on this subject [1],[2],[15].

4.2. Hierarchies

For any dimension, two types of hierarchies may be defined: *common hierarchy* and *policy hierarchy*.

Common hierarchy comprises general classifications, common for the REA primitive for which the dimension in question was defined. For instance, for agents that are external to an organization, classifications based on *location* or *connected-to* relationships are usually performed, internal agents are frequently classified through *play-the-role-of, responsible-for, has-rights-to* relationships, economic resources are classified through *is-a, part-of* relationships and so on. As the nature of the *thing* from which the dimension originates also dictates the type of relations which govern the common hierarchy or hierarchies that may be created, *dimensional design patterns* may be defined [9].

Policy hierarchies are determined by structures captured by *type images*. Their impact on decision-making is unquestionable, given that they enable controls and discrepancy analysis driven by policy elements defined by the enterprise. Classification of individual customers (external agents) into age groups or income levels directly corresponds to a certain policy vision and no longer has the generality of the first type of hierarchies.

4.3. Policy stars

The three kinds of *policy definition - attribute, association, association class –* have various representations within hierarchies. The simplest of all is achieved by inserting the attribute or respective attributes on the corresponding dimensional level. The other two introduce associations between levels in different dimensions.

The presence of associations between dimensions of the same star or those of different stars was discussed, for example, in [1]. However, the situation is different here, as these associations are many-to-many associations and, in addition, they also have attributes. Therefore, the solution proposed by the current paper is directed at representing policy associations through distinct stars. As dimensions, they have types or groups, in relation to which policy elements are defined by the organization management. In order to build the cubes required for performing controls and discrepancy analysis, the measures in the corresponding operational star or stars are aggregated up to the dimensional level of the policy type or policy group and then are merged with the policy star, thus resulting a fact structure which holds together standard values or targeted values as well as the actual values. In view of such an approach, the following guidelines directed at the policy level may be stated**:**

a. each policy association or policy association class generates a policy star;
b. targeted values (standards, budgets) represent the measures of the policy star's fact;
c. policy types or policy groups involved in policy associations become dimensions;
d. if policy definition is dependent on time, a time dimension must be added.

The result of applying these guidelines on the example in Figure 3 is represented in Figure 6. Transport star groups data associated to each of the transports that have been operated. For the three policy associations, there have been created three policy stars. Both comfort star and consumption star have two dimensions, corresponding to the policy types for which they have been defined. *Targeted fuel consumption* is the measure of the consumption star's fact. Comfort star's fact does not comprise any measures (factless). As Budget is defined on time's period, the corresponding star also includes time dimension. *Budgeted income* and *Actual income* (derived) are the measures of ActivityBudget star's fact. Actual income is calculated by summing up the values of *Income* in case of transports operated on each *Line*, and confined to the period stretching from *Beginning date* to *Ending date*.

According to the author's knowledge, the problem of representation of policy elements in dimensional modeling was not tackled by literature. Their representation through different stars associated to type images completes the model, while remaining coherent with dimensional modeling - related concepts - facts and dimensions – and OLAP operations.

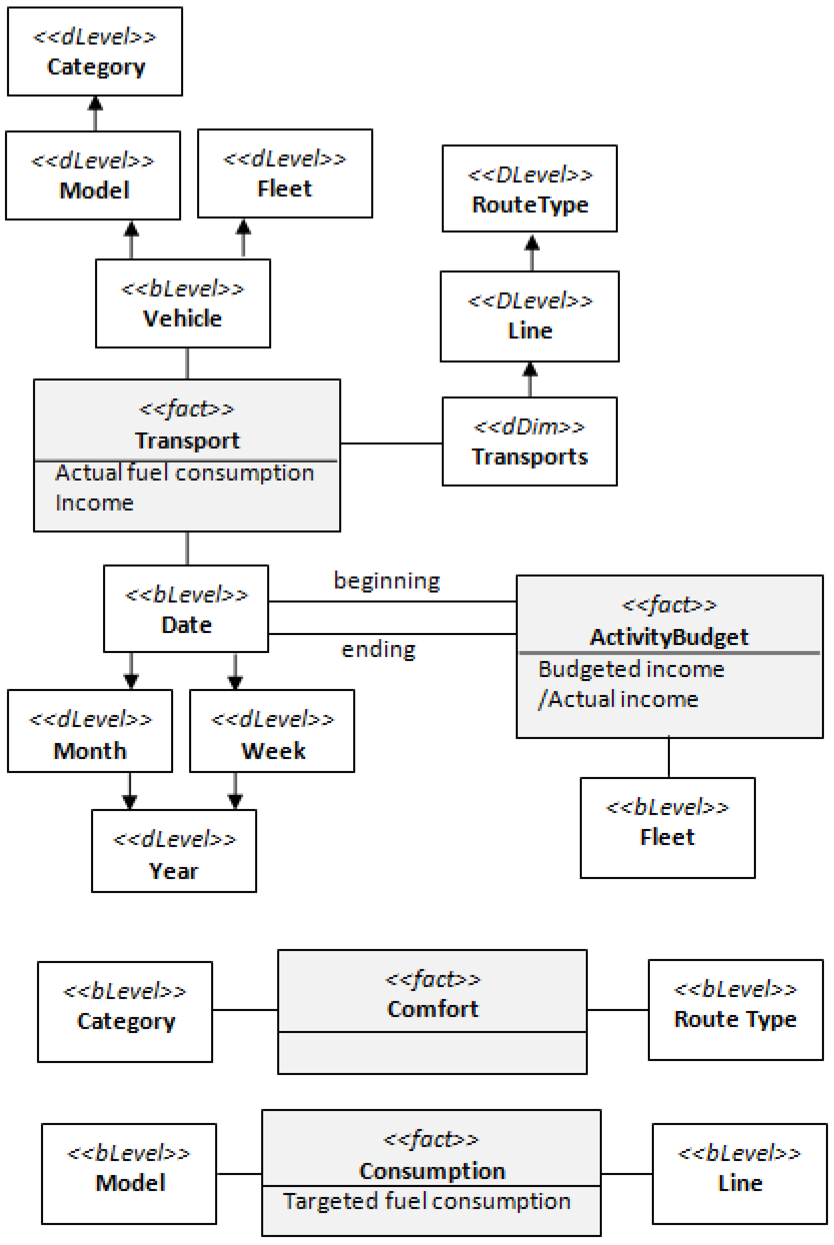


Figure 6. Dimensional schemata for policy level example

It is imperative to mention that capturing the evolution in time requires new additions to the model. However, they are outside the scope of the current paper, but they have been extensively discussed in [13], within the general framework of dimensional modeling.

5. Conceptual design process of a data warehouse

A conceptual model offers the representation of a system, from the perspective of its users. As it's the case with any business information system, we have to deal with two key issues: data storage and data querying. Data warehousing-related approaches emphasize consultation of data: the schema, in itself, it is considered as directed at data querying. However, there is a second aspect to consider, namely data feeding and storage. The fact that the data source is represented by databases used by other systems, which are already operational, does not change things dramatically. The process of periodically feeding the warehouse with data must also be addressed by conceptual modeling.

As a result, this paper advocates a process of conceptual dimensional modeling consisting of the following steps:

a. Definition of application ontology scheme
b. Creation of the initial dimensional model scheme
c. Preliminary validation of the scheme
d. Confrontation with data sources and preliminary definition of ETL processes
e. Adjustment and final validation of the dimensional model scheme.

Definition of application ontology relies on REA domain ontology and begins with detailed specification of economic resources, economic events and economic agents employed by enterprise business processes. From this point on, there are particularities which set apart industries, and even enterprises within the same industry. For example, resources, events and agents will be completely different in case of banking and manufacturing, or in that of healthcare and insurance. These particularities stem not only from the specifics of business activities, but also from dissimilar managerial visions. The actual interpretation of each of the REA primitives may be different from one manager to another, but the concepts of resource, event or agent have the quality of being equally familiar to all of them.

What follows next is to decide how economic events are being articulated, which directly impacts on how the analysis is structured, as it may be: focused on resources (case *a* in Figure **2**), focused on duality (case *b* in Figure **2**), focused on isolated treatment of events (case *c* in Figure **2**), or mixed. Kimball (2008) mentions three types of fundamental fact grains: *transaction*, *periodic snapshot* and *accumulating snapshot*. The option for one or another is, as it can easily be inferred, in a relation of close interdependency with the manner in which economic events are articulated.

Operational policy elements are further defined, by identifying types and groupings in present managerial practice or intended for the future, as well as policy definitions.

Definition of application ontology imperatively requires stakeholders participation.

The initial dimensional model scheme is created by employing the above-stated guidelines, starting on the operational level. As already mentioned, each economic event and, if necessary, each economic event phase, generally leads to he definition of a distinct star. For the identified dimensions, except the derived ones, common hierarchies are defined.

For stars defined in such manner, the implications of each type image on the policy level are examined. As previously explained, this may lead to new hierarchies, new descriptors on pre-existing levels or dimensions, or may require definition of policy stars. An aspect that was not detailed by the present paper is establishing measures as constituents of facts, as well as descriptors for dimensions and dimension levels. Many of these elements actually result “naturally”, as the two levels of the application ontology are being built, or they stem from common hierarchies.

The initial dimensional model scheme may be created without direct participation of the system’s stakeholders, as it actually represents the dimensional expression of what the ontological application scheme conveys.

The purpose of *preliminary validation* is to inform stakeholders on what the future system could offer in terms of capturing and analyzing business processes, according to the commonly agreed ontology. This is when facts, dimensions and levels are filled with measures and descriptors, respectively. When agreed, certain dimensions and levels, regarded as irrelevant may be removed, while new dimensions and levels may be added in order to satisfy particular requirements. Involving future users, the opportunity of maintaining policy stars is assed by balancing decisional benefits against exploitation costs. It is not uncommon that validation requires reviews of the ontological application scheme, for changes and corrections.

Confrontation with data sources is directed at identifying a) the data sources which will feed the data warehouse and b) transformations that should be performed on data in those sources. Such data sources are either databases of transactional systems which are already operational, or various collections of data, from inside or outside the enterprise, which are stored in a variety of formats and technologies. As the purposes for which these data sources have been created have nothing to do with the data warehouse, two key issues have to be addressed: checking data availability and integration possibilities. The former simply establishes whether the necessary data can be supplied for available stars. Depending on the actual situation, some stars may be eliminated, or, when possible, they are adjusted to match available data sources. The latter problem involves the stars for which necessary data is available, but data structure and/or semantics do not match those of the data warehouse. The confrontation is meant to establish the mappings between the schemas of the data sources and dimensional schema, necessary transformations of original data, and naturally, whether they are possible or not. This confrontation is a conceptual one, but it should be performed with maximum care in order no to compromise future development effort and system’s transition to exploitation phase. Such inter-schema mappings and the corresponding transformations are the foundation of ETL processes of the future data warehouse.

Depending on the results of such confrontations, the already-discussed dimensional model scheme may remain unchanged, or it may be altered as required. Hence, a new *validation with the stakeholders* will take place, in order for the result of changes to be explained, which is extremely important for the transparency of the modeling process, as well as for gaining stakeholders support. This is when the final dimensional scheme is eventually completed serving as a foundation for definition of elements on the customization and application layers - data cubes, predefined reports and analysis - and when final adjustments are being operated, if necessary.

6. Conclusions

This paper has presented an ontology-based dimensional modeling approach for data warehouse design. REA domain ontology has been used for this purpose, serving as a foundation for the specification of a set of guidelines for dimensional modeling. By representing business processes targeted by the data warehouse through an application ontology scheme based on REA domain ontology, the corresponding dimensional scheme may be produced even as the result of semi-automated generation. This scheme is subject to two stages of validation and adjustment: in relation to data sources, as well as to specific user needs. This particular approach favors communication with stakeholders in a manner which is familiar to them and, at the same time, shifts the focus of modeling on representation of business processes and elements of business policy, and therefore creates openness towards various analysis. Confrontation with data sources in early phases of modeling is also a positive aspect, as it allows reduction of risks during data warehouse development process.

An important direction for future research is represented by the use of semantic web technologies such as Web Ontology Language (OWL) and Semantic Web Rule Language (SWRL), with the purpose of creating specialized REA-based OWL ontology to model application domain at operational and policy levels.

REFERENCES

[1] **Abelo A., Samos J., Saltor F. (2006),** ***YAM2: A Multidimensional Conceptual Model Extending UML***. *Information Systems 31* , 541–567;
[2] **Dori D., Feldman R., Sturm A. (2008),** ***From Conceptual Models to Schemata: An Object-process-based Data Warehouse Construction Method***. *Information systems 33* , 567–593;
[3] **Gailly F., Laurier W., Poels G. (2008),** ***Positioning and Formalizing the REA Enterprise Ontology***. *Journal of Information Systems; Vol 22, No 2* , 219-248;
[4] **Geerts G. L., McCarthy W. E. (2002),** ***An Ontological Analysis of the Economic Primitives of the Extended-REA Enterprise Information Architecture***. *International Journal of Accounting Information Systems 3* , 1–16;
[5] **Geerts G.L., McCarthy W.E. (2006),** ***Policy-level Specifications in REA Enterprise Information Systems***. *Journal of Information Systems Vol. 20, No. 2* , 37-63;
[6] **Gruber, T. (1993),** ***A Translation Approach to Portable Ontology Specifications***. *Knowledge Acquisition* , 199-220;

[7] **Hacid M., Sattler U. (1997),** ***An Object-centered Multi-dimensional Data Model with Hierarchically Structured Dimensions.*** *Proceedings of the IEEE Knowledge and Data Engineering Exchange Workshop (KDEX 1997)* (pp. 65–72). IEEE Computer Society;
[8] **Huang C-C,Tseng T-L., Li,M-Z., Gung R.R. (2006),** ***Models of Multi-dimensional Analysis for Qualitative Data and Its Application***. *European Journal of Operational Research 174* , 983–1008;
[9] **Jones M.E., Song I.Y. (2008),** ***Dimensional Modeling: Identification, Classification and Evaluation of Patterns***. *Decision Support Systems 45* , 59–76.
[10] Kimball, R. et al. (2008). *The Data Warehouse Lifecycle Toolkit, second edition.* Wiley Publishing, Inc.
[11] **Lujàn-Mora S., Trujillo J., Song I.Y. (2006*), A UML Profile for Multidimensional Modeling in Data Warehouses***. *Data & Knowledge Engineering 59* , 725–769;
[12] **March S.T. , Hevner A.R. (2007),** ***Integrated Decision Support Systems: A Data Warehousing Perspective.*** *Decision Support Systems 43* , 1031– 1043.
[13**] Malinowski E., Zimànyi E. (2008),** ***A Conceptual Model for Temporal Data Warehouses and Its Transformation to the ER and the Object-relational Models***. *Data & Knowledge Engineering 64* , 101–133;
[14] **McCarthy, W. (1982),** ***The REA Accounting Model: A Generalized Framework for Accounting Systems in a Shared Data Environment***. *The Accounting Review Vol. LVII, No. 3* , 554-578;
[15] **Schneider, M. (2008),** ***A General Model for the Design of Data Warehouses.*** *International Journal of Production Economics 112* , 309–325;
[16] **Wormus, T. (2007),** ***Complex Event Processing***. *Business Intelligence Journal Vol. 13, No. 4* , 53-58.