Methods for Heterogeneity Detection During Multi-Dimensional Data Mart Integration

Geno Stefanov*

Summary:

The paper focuses on the detection of heterogeneities between multi-dimensional data marts. In many cases, data which resides in multiple and independently developed data marts is needed for decision-making. The multi-dimensional model introduces, in addition to the ER data model, dimension and fact entity. As a result of the multi-dimensional model elements, two groups of heterogeneities have been identified – dimension and fact. The former depends on differences between the dimensions' hierarchies, their members, the names of the members, their levels and dimensions. The latter kind of heterogeneities occurs when facts in different data marts are in different names, values (inconsistent measures), formats or even on a different scale. Therefore, the paper examines and classifies the heterogeneities which can occur during the integration of independently developed data marts and four methods for heterogeneity detection are proposed and discussed. The methods are as follows: method for metadata extraction, method for detecting schema-instance heterogeneities, method for detecting heterogeneities among dimensions and method for detecting heterogeneities among facts. The paper ends with conclusions about the advantages of the proposed methods for heterogeneity detection during the integration of data marts.

Key words: multi-dimensional model, data mart, method.

JEL Classification: C8, L86.

1. Introduction

One problem that needs to be resolved in many practical cases is the integration of data marts that have been developed and operated independently. In many cases decision making requires data that resides in multiple and independently developed data sources. For example, if we want to compare several KPIs (Key performance indicator) from different data marts or to define new **KPI**, which is defined by the KPIs residing in several different data marts, we face two possible choices – building a new data mart or the integration of the existing ones. Building a new data mart is a costly and time consuming task, so it would be better to look for mechanisms for integration of the existing data marts. Another case that requires marts integration is in the event of mergers and acquisitions of different companies. In this case, for example, one company acquires another company and the data marts in the acquired company should be integrated into the data warehouse of the acquiring company. Here we have the same possible choices as in the previous case.

The multidimensional model (MDM) on which the concept of data mart is based is built out of three basic constructive elements: the facts which are analyzed, the dimensions (coordinates of the fact) and the measures

* Assistant Professor, PhD, Department of Information technologies and communications, University of National and World Economy; e-mail: genostefanov@unwe.bg

which allow for the quantitative evaluation of the facts (Vassiliadis, P., Sellis, T. K., 1999). It is quite often the case that large organizations need integrating independently developed data marts. These data marts should be based on common dimensions and facts, but in many cases different company develop their own data marts and their integration becomes a difficult task. The difficulties stem from the heterogeneities of the MDM elements and from a semantic point of view they can be classified as dimension and fact heterogeneities. Dimension heterogeneities occur when the dimension schema structures, dimension members or the naming of semantically-related dimensions have semantic discrepancies. Fact heterogeneities occur when the measures in different data marts are given different names, values (inconsistent measures), formats or even different scale. There has been research on the problem of resolving the heterogeneities occurring in data mart integration (Tseng, F. S. C., Chen, C-W., 2005), (Torlone, R., 2008) and (Berger, S. and Schref, M., 2008). The main part of data mart integration is resolving the possible heterogeneities. To the best of our knowledge there are no proposals for methods or techniques which will allow the data mart integration process to automatically identify the possible heterogeneities between the integrated MDM models.

2. Data mart construct and heterogeneities

Intuitively, a DM defines one or more measure variables within fact tables, categorized by some dimensions that are organized in hierarchies of levels. Facts and dimensions consist of both their schema and the corresponding instances. For the dimension instances, the commonly accepted term dimension members (or members for short) (Vassiliadis, P., Sellis, T. K., 1999) is used throughout the paper.

In order to achieve the goal of the current paper, the constructing elements of a data mart need to be analyzed and defined. A DM = {F1,..., Fn, D1, ..., Dm}, consists of non-empty set of dimensions and fact tables (Berger, S., Schref, M., 2008), (Golfarelli, M., Maio, D., Rizzi, S., 1998).

Now let {τ1, ..., τm} be a finite set of data types (e.g. integers) with their domain defined by function dom(τ).

A dimension $D \in \{D1, ..., Dm\}$ of a DM is composed of:

- The dimension schema SD = {LD, S (LD), HD} containing finite, non-empty set of Levels LD = {l1,…., lm, lAll}, with level schema S(LD) = {Sl1,….,Slm} and the roll-up hierarchy HD⊆ LD x LD;
- The level Schema Sl ∈ S(LD) of some level l_i is an attribute schema (K,N1, ……, Nk) with name l, key K(the dimensional attribute) and optional non-dimensional attributes N1,… Nk. Every Nk attribute ∈ Sl has a domain dom (Nk) = dom (τk);
- The dimension instance d(SD) over schema SD with name d containing a set of members Vd with each v ∈ Vd being a tuple over a level schema Sl, and a family of "roll-up" relationships between the member subsets, defined by the dimension hierarchy.

A fact table F ∈ {F1, ..., Fm} of DM is composed of:

- The fact scheme Sf = {Af, Mf} containing a finite, non-empty set of dimensional attributes Af = {A1,… An} and a set of measure attributes Mf ={ M1,… Mn}. Each Ai ∈ Af is linked with a level l ∈ LD of the dimensions D, each Mj ∈ Mf with a τj. The domain of the attributes in Af is defined as dom(Ai) = members(l) and dom(Mj) = dom(τj);
- The fact instance f(Sf), a set of tuples over {[dom(A1) × ... × dom(An)], [dom(M1) × ... × dom(Mm)]}. A tuple f ∈ f(Sf) is called a "cell" or "fact". Moreover, we call the values [f(A1), ..., f(An)] the "coordinates" of a cell, modelling the multi-dimensional context for the measures [f(M1), ..., f(Mm)].

Table 1. Heterogeneities between multidimensional data marts

Heterogeneity	Level of heterogeneity	**Description**
Dimension naming heterogeneities	Dimension Schema	Different name labels for ontologically same schema elements
Non-corresponding Dimension schemas	Dimension Schema	Two dimension schemas do not have any common equivalent aggregation level
Inner-level corresponding	Dimension Schema	Two dimension schemas have at least one, but not all equivalent level(s) in common and the base level is different
Base-level corresponding	Dimension Schema	Two dimension schemas have at least one, but not all equivalent level(s) in common and the base level is equivalent
Flat corresponding dimension schemas	Dimension Schema	Two dimension schemas have equivalent sets of aggregation levels with identical base levels, but the hierarchies do not match
Inner level domain conflicts	Dimension Schema	Two attributes in level schemas of autonomous dimensions assign different sets of allowed values (i.e., different domains).
Base level domain	Dimension Schema	Two attributes in level schemas of autonomous dimensions assign different sets of allowed values (i.e., different domains).
Non-dimensional attributes domain	Dimension Schema	Two attributes in level schemas of autonomous dimensions assign different sets of allowed values (i.e., different domains).
Heterogeneous roll-up functions	Dimension Instance	The roll-up functions of two equivalent members define ambiguous parent members
Non-dimensional value heterogeneity	Dimension Instance	Some N-attribute contains ambiguous values among two equivalent members that belong to levels with equivalent schemas
Fact naming heterogeneities	Fact Schema	Different name labels for ontologically same attributes
Non-corresponding fact schemas	Fact Schema	Two fact schemas do not have any equivalent dimensional attribute in common
Partially corresponding fact schemas	Fact Schema	Two fact schemas have at least one, but not all equivalent dimensional attribute(s) in common
Domain heterogeneities	Fact Schema	Two attributes in autonomous fact schemas assign different sets of allowed values
Overlapping Fact cells	Fact instance (measure)	Two fact tables define identical dimensional attributes and overlapping measure attributes
Different measures	Fact instance (measure)	Two fact tables have all equivalent dimension attributes, but do not have any measures in common

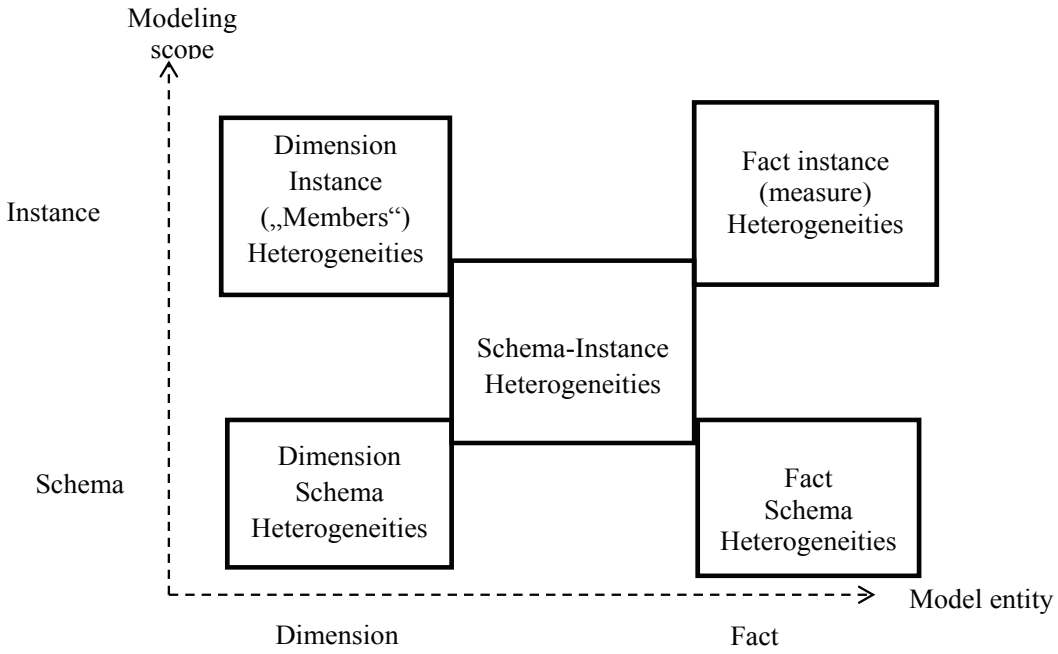


Fig. 1. Levels of heterogeneities

Based on the formal definition of data mart elements, an analysis of the possible heterogeneities is required. Heterogeneities among independent and autonomous data marts may occur either at the schema or at the instance level and involve both dimensions and facts. Generally, heterogeneities among Data Marts covering the same application domain (e.g., sales figures of grocery stores) result from the use of either (1) different modelling patterns and methodologies, or (2) ambiguous domain vocabularies, or (3) a combination of both these factors. In recent years the heterogeneities that are specific to data warehouses and the multidimensional conceptual model are analyzed in (Torlone, R., 2008), (Berger, S., Schref, M., 2008) and (Golfarelli, M., Maio, D., Rizzi, S., 1998). A classification of the possible heterogeneities between multidimensional data marts is proposed along two dimensions: modelling scope (schema level, instance level) and model entity (dimension, fact). Thus five basic levels of heterogeneities are obtained (fig. 1).

A summary of the possible heterogeneities on the different levels are presented in the table 1.

3. Methods for heterogeneities detection

DWs and data marts are basically the implementation of multidimensional data models along with some tools for manipulating the multidimensional data. Data marts implement/represent the multidimensional data instance either as relational systems (ROLAP), proprietary multidimensional systems (MOLAP), or a hybrid of both (HOLAP).

In ROLAP (relational online analytical processing) systems, the multidimensional data model is an E/R model and it is represented as a relational database (RDB) instance.

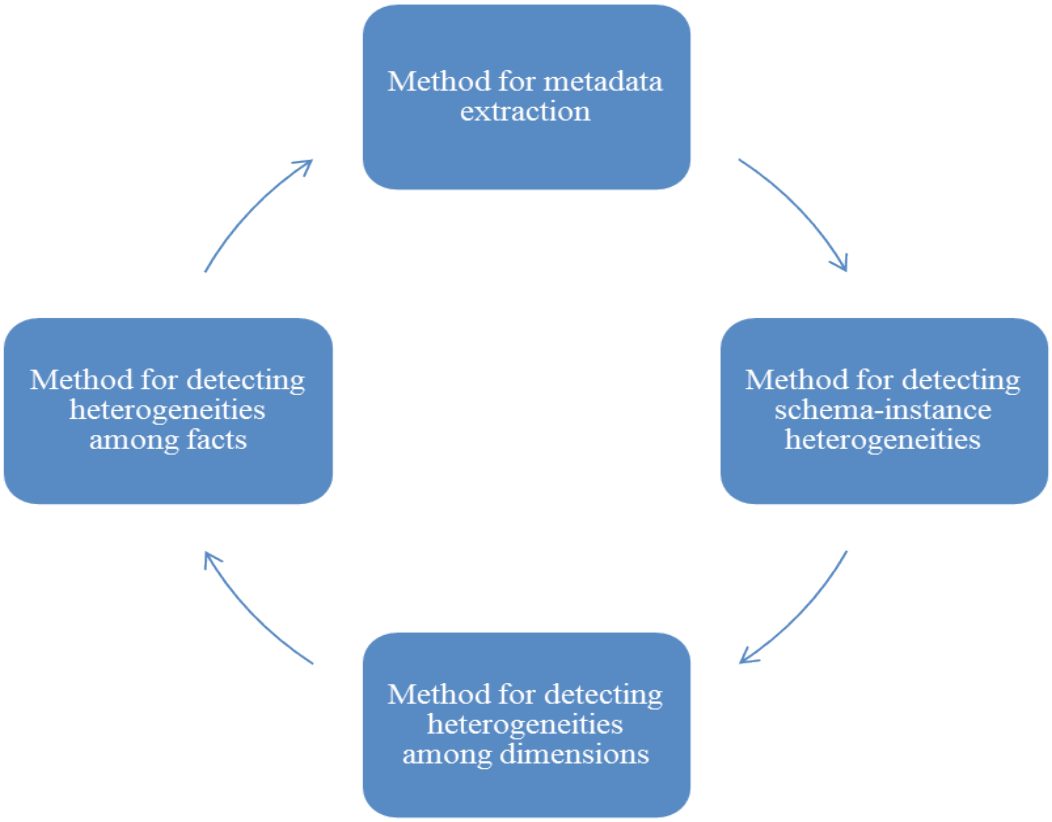


Fig. 2. Methods for heterogeneities detection

Although ROLAP uses a relational database to represent the MD instance, the database must be carefully designed for analytical processing.

Star and Snowflake are the two best-known relational schemas that are specifically designed for ROLAP MD databases. ROLAP maps operations on multidimensional data model to standard relational operations. An advantage of these systems is that they can be easily integrated into other relational information systems (Kimball, R., Ross, M., 2002). Based on (Kimball, R., Ross, M., 2002), ROLAPs are the most common implementation of the multidimensional instance in DWs. The rich infrastructure of RDBs enables using standard, common and established techniques (Wijsen, J., 2003). Throughout this paper the realization of a DM as a relational database, based on a star schema model, is being considered.

Considering the process of data mart integration and the fact that ROLAP is the most common implementation of multidimensional data mart, four methods for heterogeneity detection are proposed and developed (fig. 2).

3.1. Method for metadata extraction

In order to achieve the idea of detecting heterogeneities a set of data describing the MDM data model is needed. This first method has as its goal to extract metadata about the data marts which are integrated. The method provides the opportunities for extracting the following metadata (fig. 3):

- Metadata coming from the data mart structure including automatically

identifying dimensions and fact tables, names of dimensions and facts, names of measures, dimensional and non-dimensional attributes, dimensional and non-dimensional domains, automatic identification of facts measures;

- User's metadata including defining dimensional and non-dimensional attributes within the dimension schema, defining dimension hierarchy.

The mode of operation and executed tasks of the method for metadata extraction can be described in the following steps (fig. 4):

- Determining dimension schema
  - Determining dimensional attributes and their properties and characteristics;
  - Determining non-dimensional attributes and their properties and characteristics;

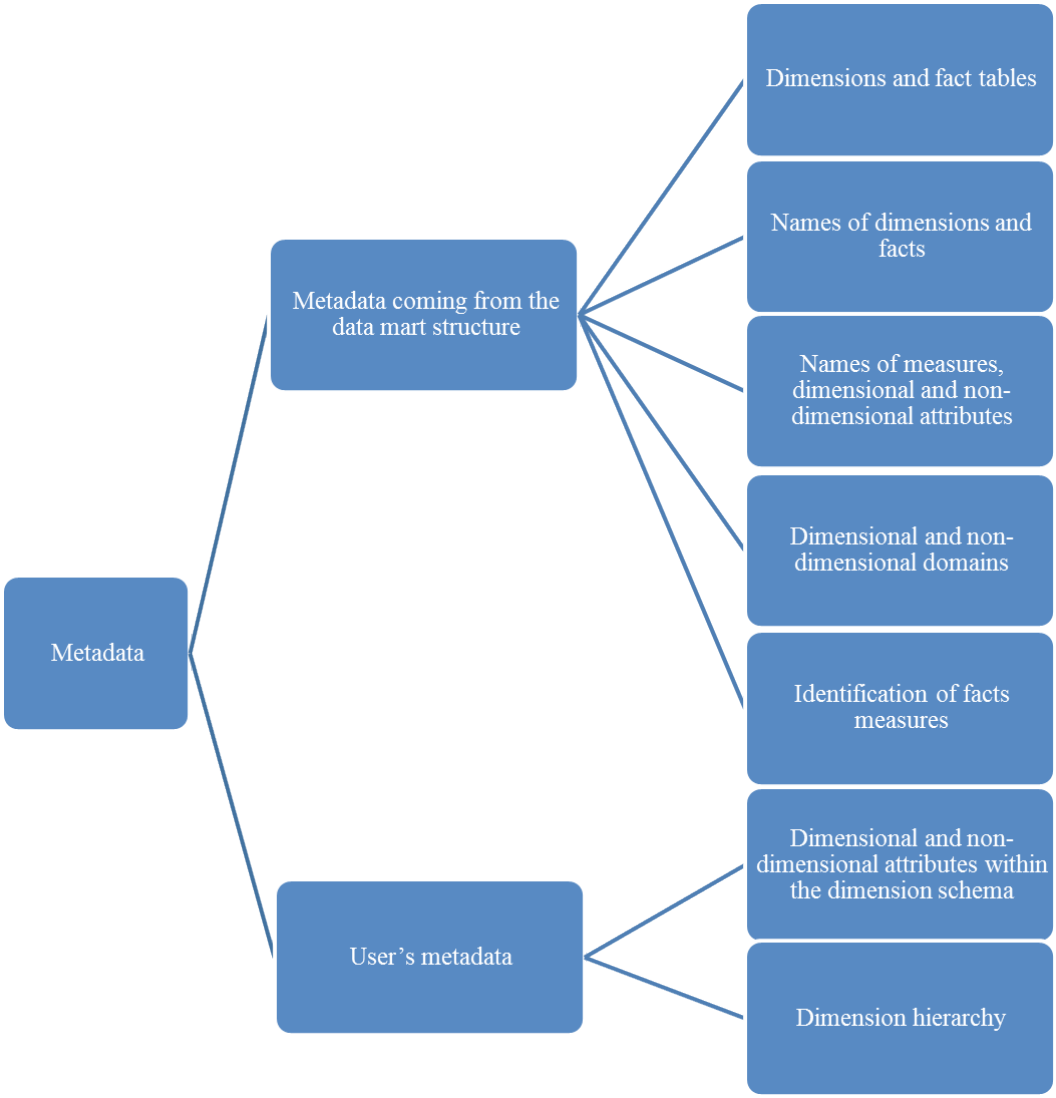


Fig. 3. Metadata elements

- Determining facts schema
  - Determining dimensional attributes within the fact schema;
  - Determining measures within the fact schema;

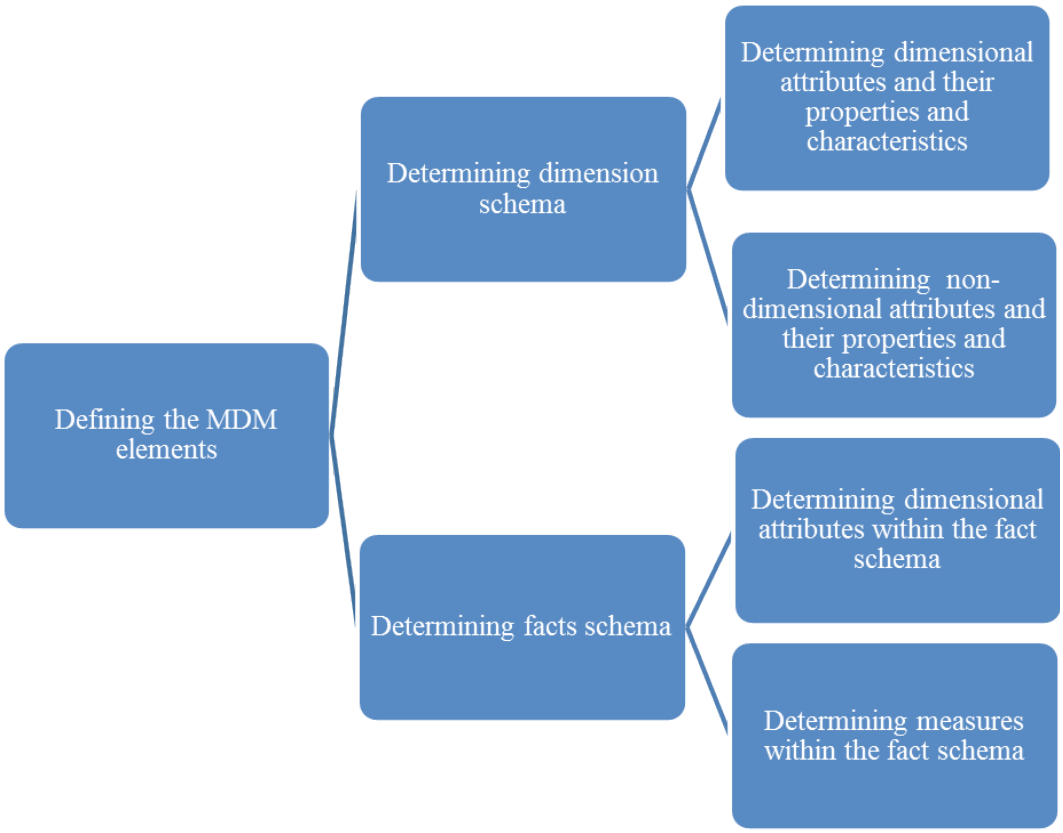


Fig. 4. Tasks within the method for metadata extraction

The algorithmic description of the method includes all the necessary steps for retrieval and storage of the data mart's elements and their characteristics in the form of metadata (fig. 5).

Method for metadata extraction

Input: relational schema R={$E_{1,...}E_n$}

Output: multi-dimensional data model MDM = {$F_1 = (A_{f1}\ M_{f1}, dom(A_1), dom(F_1))$,

f $(S_f1), \ldots, F_n = (A_{fn}M_{fn}, dom(A_n), dom\ (F_n)$ f $(S_f))\ D_1 =$

(d(SD1), dom(l. K1), dom(l. N1), LD1, S(LD1), HD1), ..., D_n = (d (SD1), dom(l. K1), dom(l. N1), LDn, S (LDn), HDn

```
1:  for all E ∈ R do
2:      if (E_i. getPKCount () = 1) then
3:          Add (E_i) as D to MDM
4:      else if (T_i. getFKCount () > 0) then
5:          Add (E_i) as F to MDM
```

```
6:              end if
7:              end if
8:    end for
9:    for all D ∈ MDM do
10:             for all D.Attr ∈ D_i
11:                     if (askUser (D_i. Attr_j) = dimAtrr) then
12:                             Add (Attr_j) as l to LD_i and HD_i
13:                             Add (l. K_i.dataType ()) as dom (l. K_i) to LD_i
14:                     else
15:                             Add (Attr_j) as l. N_i to Sl
16:                             Add (l. N_i.dataType ()) as dom (l. N_i) to Sl
17:                     end if
18:             end for
19:   end for
20:   for all F ∈ MDM do
21:             for all F.Attr ∈ F
22:                     if (F_i. Attr_j = FK) then
23:                             Add (Attr_j) as A_i to F_i
24:                             Add (members (getLevel (Attr_j))) as dom (A_i) to F_i
25:                     end if
26:                     else
27:                             Add (Attr_j) as M_i to F_i
28:                             Add (M_i.dataType ()) as dom (M_i) to F_i
29:                     end if
30:             end for
31:   end for
32:   return MDM
```

Fig. 5 Algorithmic description of the method for metadata extraction

The result of the implementation of the method for metadata extraction represent the multi-dimensional data model elements extracted from relational schema. The implementation of this method is very important for the correct functioning and implementation of the following methods, considering that the analysis of the extracted metadata will determine the possible heterogeneities between the integrated data marts.

3.2. Method for detecting schema-instance heterogeneities

The second method for detecting heterogeneities is the method for detecting schema-instance heterogeneities. This method provides possibilities for detecting heterogeneities on schema-instance level using the extracted metadata from the method for metadata extraction.

The purpose of this method is to provide indication of differences between the ways of modeling of the integrated data marts. Sometimes, part of the fact context is modelled using elements at the schema once and elements at the instance level across distinct autonomous data marts once. The schema-instance heterogeneities are very difficult to detect automatically; thus a user interaction is needed to verify the existence of this heterogeneities. The mode of operation and executed tasks of the method for detecting schema-instance heterogeneities can be described in the following steps (fig. 6):

- Determining the count of the dimensions within the both of the integrated data marts;
- Ascertaining if the number of dimensions is equal.

The algorithmic description of the method includes all the necessary steps for solving the defined tasks of the method (fig. 7).

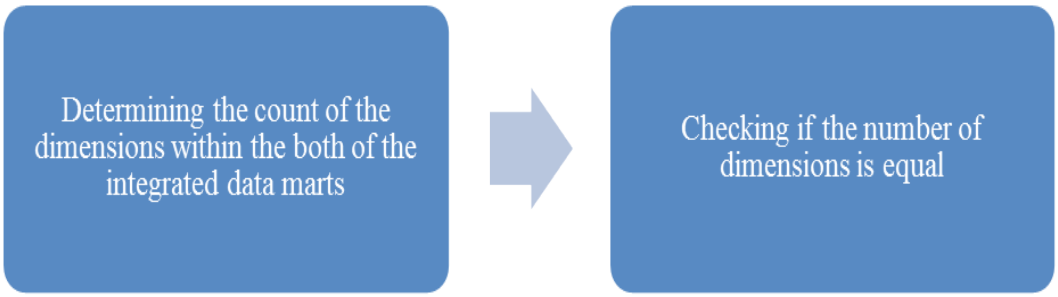


Fig. 6. Tasks within the method for detecting schema-instance heterogeneities

Method for detecting schema-instance heterogeneities

Input: two multi-dimensional data models – MDM = $\{F_1 \dots F_n , D_1 \dots D_n\}$ и MDM' = $\{F_1' \dots F_n' , D_1' \dots D_n'\}$

Output: ***true*** or ***false***, respectively for the presence or absence of this kind of heterogeneities

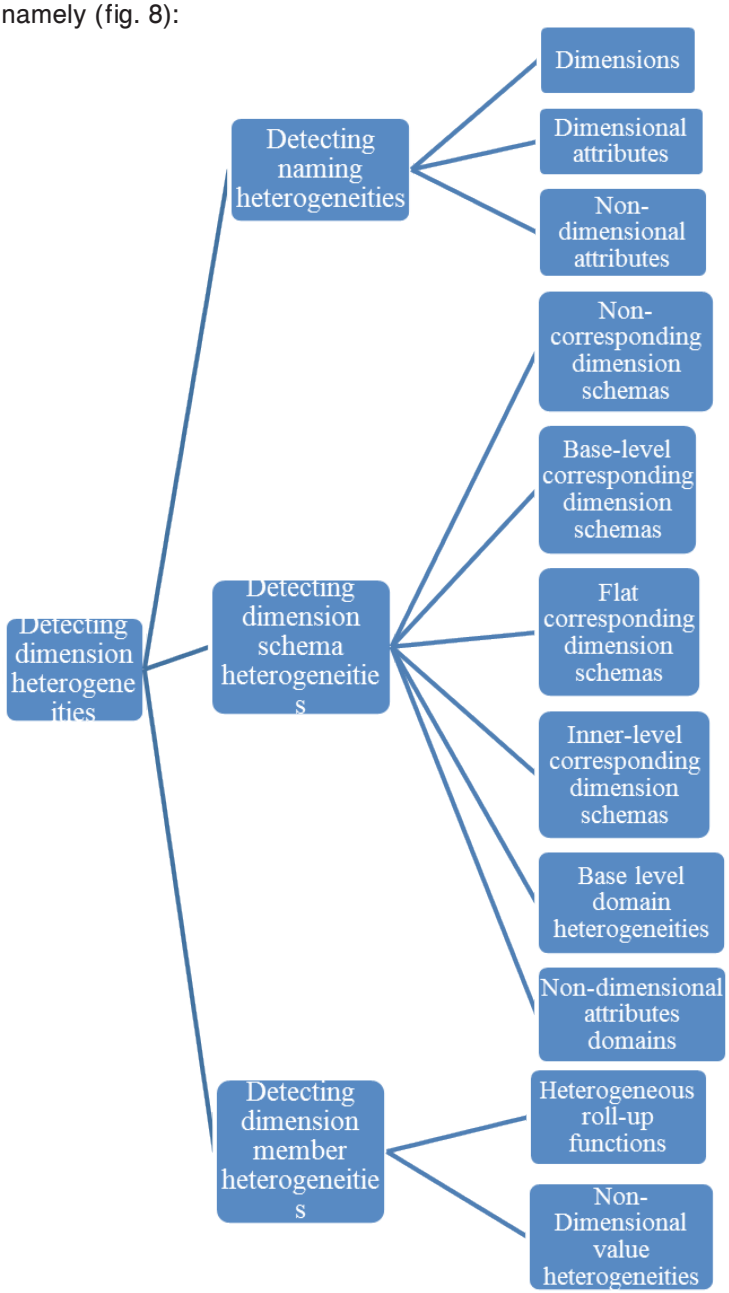
```
1:  for all D ∈ MDM do
2:          countMDM ++
3:  end for
4:  for all D ∈ MDM do
5:          countMDM'++
6:  end for
7:  if (countMDM ≠ countMDM') then
8:          return true
9:  else
10:         return false
11: end if
```

Fig. 7. Algorithmic description of the method for detecting schema-instance heterogeneities

If the number of dimensions is different in the integrated data marts, this denotes a different number of dimensional attributes in the fact tables schemas.

3.3. Method for detecting heterogeneities among dimensions

The third method for detecting heterogeneities is the method for detecting heterogeneities among dimensions. This method provides opportunities for detecting heterogeneities on dimensions level using the extracted metadata from the method for metadata extraction, namely (fig. 8):



Fig, 8. Method for detecting heterogeneities among dimensions

- Detecting naming heterogeneities among (1) dimensions, (2) dimensional attributes, (3) non-dimensional attributes in level schema;
- Detecting dimension schema heterogeneities – (1) non-corresponding dimension schemas, (2) inner-level corresponding dimension schemas, (3) base-level corresponding dimension schemas, (4) flat corresponding dimension schemas, (5) base level domain heterogeneities, (6) non-dimensional attributes domains;
- Detecting dimension member heterogeneities – (1) heterogeneous roll-up functions and (2) non-dimensional value heterogeneities.

The algorithmic description of the method includes all the necessary steps for detection of heterogeneities which exist on dimension level (fig. 9).

Method for detecting heterogeneities among dimensions

Input: dimensions D and D'
Output: List of all existing heterogeneities among the dimensions D and D' *L*

```
for all l ∈ LD do
  for all l' ∈ LD' do
        ds=1-LevenshteinDistance (name (l_i),name (l'_j))/count_charact (name (l_i),name (l'_j))
            if (dataType(l_i) = dataType(l_i) & ds > 0.6) then
            l_i ≈ l'_j
            count++
            else
            noncorres++
            end if
        end for
end for
if(count=0 & (1 − LevenshteinDistance (name(D), name(D'))/ count_charact (name (D), name (D')) < 0,5) then
Add NonCorspondDimSchemasConf (D,D') to L
go to end
else
        if (name (D) ≠ name (D')) then
        Add DimNameConf (name (D), name (D')) to L
        end if
        for all l ∈ LD do
            for all l' ∈ LD' do
                if (l_i ≈ l'_j & (name (l_i) ≠ name (l'_j))) then
Add LevelNameConf (name (l_i), name (l'_j)) to L
                end if
if (l_i ≈ l'_j & dom (l_i. k) ≠ dom (l'_j. k')) then
Add DimDomainConf (dom (l_i. k), dom (l'_j. k')) to L
                end if
                for all l_i.N ∈ Sl do
                    for all l'_j.N' ∈ S'_l do
                    if (l_i ≈ l'_j & dom (l_i. N_k) ≠ dom (l'_j. N'_m)) then
                Add NonDimDomainConf (dom (l_i. N_k), dom (l'_j. N'_m)) to L
                        end if
                    if (name (l_i. N_k)≠name (l'_j. N'_m)) then
```

```
33:                                   Add NonDimmNameConf (name (l_i. N_k),
                                                           name (l'_j. N'_m)) to L
34:                                   end if
35:                                   end for
36:                               end for
37:                       end for
38:           end for
39:           if (l_0^D ≠ l_0^D' & count < noncorres) then
40:           Add InnerSchemaCorresConf (SD, SD') to L
41:           else if (l_0^D = l_0^D' & count < noncorres)
42:           Add BaselevelCorresConf (SD, SD') to L
43:           if (HD ≠ HD') then
44:           Add DiffHierrarchyConf (HD, HD') to L
45:           end if
46:           for all v ∈ Vd do
47:                       for all v' ∈ Vd' do
48:                               if (v (l.K) = v' (l'.K) & r_D^{l→k} ≠ r_D'^{l'→k} ) then
49:                               Add HeterRollUpFuncConf (v, v') to L
50:                               end if
51:                               for all v (l.N) ∈ Vd  do
52:                                       for all v'(l'.N') ∈ Vd' do
53:                                           if (v (l.N_i) = v' (l'. N_i) & v (l. N_i) ≠ v' (l'. N_i)) then
54:                                       Add NonDimValueConf (v (l_i. N_i), v (l'_j. N'_j)) to L
55:                                               end if
56:                                       end for
57:                               end for
58:                       end for
59:           end for
60: end if
61: return L
```

Fig. 9. Algorithmic description of the method for detecting heterogeneities among dimensions

The result of the implementation of the software method for detecting heterogeneities among dimensions represents a list with all identified heterogeneities among two dimensions.

3.4. Method for detecting heterogeneities among facts

The last method for detecting heterogeneities is the method for detecting heterogeneities among facts. This method provides opportunities for detecting heterogeneities among facts using the extracted metadata from the method for metadata extraction, namely (fig. 10):

- Detecting naming heterogeneities among (1) fact tables, (2) dimensional attributes, (3) measures;
- Detecting fact tables schema heterogeneities – (1) non-corresponding fact schemas, (2) partially corresponding fact schemas, (3) domain heterogeneities;
- Detecting fact tables instances heterogeneities – (1) overlapping fact cells, different measures.

The algorithmic description of the method includes all the necessary steps for detection of heterogeneities which exist on the fact table's level (fig. 11).

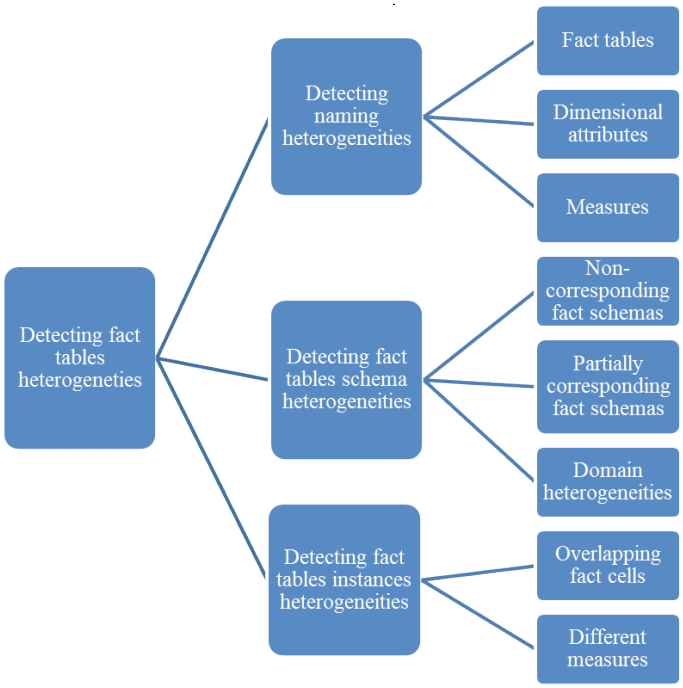


Fig. 10. Method for detecting heterogeneities among fact tables

Method for detecting heterogeneities among fact tables

Input: fact tables F = { $S_f = \{A_f, M_f\}, f(S_f)$} and F' = { $S_f' = \{A_f', M_f'\}, f'(S_f')$

Output: List of all existing heterogeneities among the fact tables D and D' L

```
1:  for all A[l] ∈ A_f do
2:   for all A[l]' ∈ A_f' do
3:          if (A[l]_i ≡ A[l]'_j) then
4:                 count++
5:          else
6:                 noncorres++
7:          end if
8:   end for
9:  end for
10: if (count=0) then
11: Add NonCorspondFactSchemasConf (F,F') to L
12: go to end
13: else
14:          if (name (F) ≠ name (F')) then
15:          Add FactNameConf (name (F), name (F')) to L
```

```
16:         end if
17:         for all A_i ∈ A_f do
18:             for all A_i' ∈ A_f' do
19:                 if (A_i ≈ A'_j &  (name (A_i) ≠ name (A'_j)))  then
20:                 Add DimAttrNameConf (name (A_i), name (A'_j)) to L
21:                 end if
22:                 if (A_i ≈ A'_j & dom (A_i) ≠ dom (A'_j)) then
23:                 Add DimAttrDomainConf (dom (A_i), dom (A'_j)) to L
24:                 end if
25:             end for
26:         end for
27:         for all M_i ∈ M_f do
28:             for all M_i' ∈ M_f' do
29:                 if (dom (M_i) ≠ dom (M'_j)) then
30:                 Add MeasureDomainConf (dom (M_i),dom (M'_j)) to L
31:                 end if
32:             end for
33:         end for
34:         if ( noncorres>0) then
35:             Add PartialCorresponFactSchemas (S_f, S_f') to L
36:         end if
37:         for all c ∈ V_d do
38:             for all v' ∈ V_d' do
39:                 if (v (l.K) = v' (l'.K) & r_D^{l→k} ≠ r_{D'}^{l'→k} ) then
40:                 Add HeterRollUpFuncConf (v, v') to L
41:                 end if
42:                 for all v (l.N) ∈ V_d do
43:                     for all v' (l'.N') ∈ V_d' do
44:                         if (v  (l.N_i)  =  v' (l'. N_i) & v  (l. N_i)  ≠  v' (l'. N_i))
    then
45:                     Add NonDimValueConf (v (l_i. N_i), v (l'_j. N'_j)) to L
46:                         end if
47:                     end for
48:                 end for
49:             end for
50:         end for
51: end if
52: return L
```

Fig. 11 Algorithmic description of the method for detecting heterogeneities among fact tables

The result of the implementation of the software method for detecting heterogeneities among fact tables represents a list with all identified heterogeneities among two fact tables.

Conclusions

In conclusion we can say that the use of the proposed methods for detecting data mart heterogeneities may shorten the process of data mart integration by reducing the risks of unidentified heterogeneities or user's mistakes. The redundancy of unidentified heterogeneities, in conjunction with common mistakes made by users, are the leading causes of data mart integration project failure (inefficient uses of time and data management in the context of running and maintaining data marts for business purposes). To achieve optimal functionality and reduce the aforementioned risks, and above all to streamline and shorten the process of data mart integration, the implementation of the proposed methods for detecting data mart heterogeneities is recommended.

References

Berger, S., Schref, M., 2008. From federated databases to a federated data warehouse system. HICSS, 0:394, http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.130.6785, [accessed October 2014]

Golfarelli, M., Maio, D., Rizzi, S., 1998. The dimensional fact model: A conceptual model for data warehouses. Int. J. Cooperative Inf. Syst., 7(2-3):215-247, http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.45.1702, [accessed November 2014]

Kimball, R., Ross, M., 2002. The Data Warehouse Toolkit: the Complete Guide to Dimensional Modeling. Wiley, 2nd edition.

Torlone, R., 2008. Two approaches to the integration of heterogeneous data warehouses, Distrib. *Parallel Databases* 23(1):69-97, http://www.dia.uniroma3.it/~torlone/pubs/dapd08.pdf, [accessed November 2014]

Tseng, F. S. C., Chen, C-W., 2005. Integrating heterogeneous data warehouses using XML technologies. *Journal of Information Science*, 31(3):209-229, http://jis.sagepub.com/content/31/3/209.short, [accessed November 2014]

Vassiliadis, P., Sellis, T. K., 1999. A survey of logical models for OLAP databases, http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.79.4986, [accessed December 2014]

Wijsen, J., 2003. Condensed Representation of Database Repairs for Consistent Query Answering. In Proceedings of the International Conference on Database Theory (ICDT03), volume 2572 of Lecture Notes in Computer Science, pages 375–390. Springer.