THE PUBLISHING HOUSE
OF THE ROMANIAN ACADEMY

*PROCEEDINGS OF THE ROMANIAN ACADEMY, Series A,*
*Volume 11, Number 3/2010, pp. 269–276*

# ORTHOGONALITY METRICS OF THE STRUCTURED ENTITIES

Ion IVAN, Daniel MILODIN, Marius POPA

Academy of Economic Studies, Bucharest, Romania
E-mail: ionivan@ase.ro

The paper defines the concepts of orthogonality and structured entities. The orthogonality metrics and their properties are presented. The orthogonality indicators are proposed and implemented.

*Key words*: Orthogonality; Structured entities metrics.

## 1. STRUCTURED ENTITIES

Structured entities are defined based on the components' inclusion property.

Structured entities create and implement storage and processing forms which are used for information management.

Structured text entities are built based on some ground concepts [4], such as:

- the symbol is a way of representation; by using symbols are put in correspondence objects, concepts, images with their representation;
- the alphabet is a finite multitude, consisting of letters used in writing a language; the basic feature of the letters to restore the words of a language is to respect a conventional order;
- the word is an association of sense and a complex sound; the word is the vocabulary's basic unity;
- the vocabulary consists of words multitude specific for a language; also, the vocabulary term defines the words specific to social categories, areas of activity and research;
- the separator is a symbol not belonging to the alphabet which as the role to delimit the words of the alphabet when used to transmit information; the texts are build by using separators; the separators : ; ? ! @ # $ % ^ & * ( ) ` ~ , . < > / \ - _ + = | " ' [ ] " " } { are allocated to the separators multitude;
- the appearance frequency is an indicator that provides information about the use level of terms from the basic vocabulary as part of the structured entities;
- the text consists of several separate words that send a certain information; the text is characterized by length, measured in number of words or the number of used symbols, the frequency of words or symbols' appearance, and the degree of affiliation to a vocabulary through is established whether the text uses terms from a field;
- the thesaurus is the multitude consisting of all words considered defining for an area;
- the referred system defines the assembly of structured elements, studied and implemented using the structured entities.

Structured entities perform functions involving the interaction with users [5], such as:

- data collection involves the existence of a calculation method that allows the users' access to the structures of entities;
- data storage ensures the retention of information by the structured entities for future reference;
- data updating requires the existence of a software module that has access to the existing information, validates the new information and updates the old information; the updating is done by total replacement or through the rewriting of some parts of the old information;
- data acquisition is the function which provides the data processing, according to the requirements and the software modules implemented.

Structured entities are built to perform operations with various types of data. Working with structured entities involves identifying the processed data, determining their specific, definition of criteria regarding their processing, storing, and updating. Structured entities are formed as a way of encapsulating and working with information. They ensure their synthesis and assembly on various types of data, in order to ensure the efficient use of data and information. Grouping the data is carried out based on performance and origin. Thus, structured entities data store and process low level data, with minimum size, such as keywords, but also voluminous data, which defines and describe complex concepts, characterized through the unity of the contained information.

Structured entities are characterized by the model of data disposal, the life cycle, the filters built to meet the structure and the implemented data types, the established performance criteria, the definition and implementation of operations that are executed with the help of structured entities, the redundancy control.

The entities' quality attributes are given by the below elements.

The entity's *opportunity* relates to its availability and usage at the right time, when the user needs. Though opportunity is identified the entities' characteristic of responding promptly to the customer's requirements by implementing a superior management of approaching the issue.

The entities' *comparability* expresses the quality characteristic that allows the entities determination with lower or higher values depending on the purpose of the comparison operation. The entities' comparison is done by a field or after several component fields, in this case marking the field that is considered primary and the secondary fields.

*Homogeneity* is the entities' characteristic of being composed of data of the same type, of structuring and storing the data according to the same model. This feature is defining for the structured entities.

*Completeness* is the entities' feature that shows to what extent the entities' fields contain values. An entity is complete when all its fields have been loaded. The completeness shows the entities' ability to provide information that normally should be stored. Also, the completeness ensures the existence of complete data regarding the treatment of a subject with the help of entities. The entities are built so that they allowed the processing of all data on the real system meant.

*Gradual character* is the entities' characteristic that ensures a progressive approach to the concepts used. The gradual character ensures the concepts' defining through using concepts already presented. The gradual character is a defining feature for the structured entities, the approach modality in their case being from simple to complex.

*Abstraction* is the feature that allows entities to take over the essential characteristics of the concepts used and their transfer from simple concepts to complex concepts, concepts inheriting their characteristics.

In Fig. 1 it is presented the administration process of the structured entities.

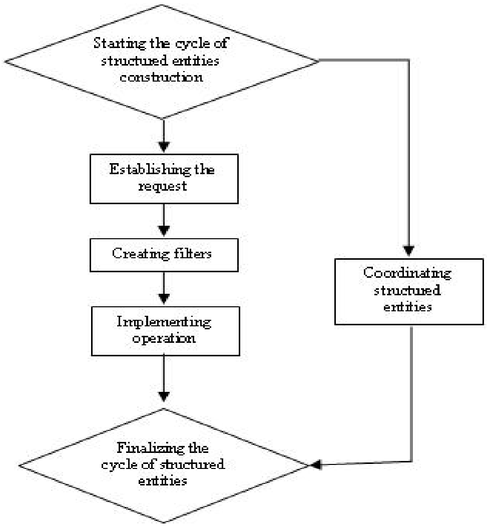
Starting the cycle of structured entities construction
Establishing the request
Creating filters
Implementing operation
Coordinating structured entities
Finalizing the cycle of structured entities

Fig. 1 – Structured entities management process.

Starting the cycle of structured entities construction is the moment that triggers the whole cycle of building structured entities. It sets the goal to implement, and in the next stage are set out requirements needed in order to achieve the objective. Filters and operations that are built represent the materialization of the conditions imposed by the requirements to implement. Completion of the construction of structured entities is marked by entry into the system that implements the concepts.

Structured text entities are a concept that combines the text in an environment built on a model. Widespread use of the concept helps to ensure a high level of orthogonality by identifying similar entities. Structured entities find applicability in a wide range of areas, ensuring the original character of the components.

## 2. METRICS OF ORTHOGONALITY

Software metric is a mathematical model that contains one or more equations or inequalities and has one or more target functions and the role is to describe the state of the associated system [6]. Metrics are implemented models to test the quality of entities, taking into account factors influencing the measured feature.

Need to use metrics is given by the following considerations: allow targets for improving implemented entities, ensure an effective way to achieve these objectives, identify the causes that adversely affect the quality characteristics of structured entities, identify requirements to be followed for the development of models of structured entities superior quality.

Metrics test differences between the entities resulting from implementation of a model structure and the expected results and determine the causes that led to these differences. Metric is a definition, an algorithm or mathematical function used for quantitative assessment of the product tested.

A metric is built to meet the following objectives: quantifying features, determining the influence of indirect factors, aggregating values, ranking entities, benchmarking. Below, it is presented the featured of metrics.

*Measurement function*. It is emphasized the level of quality characteristics measurement, by expressing the units of measurement metrics. On the basis of indicators and system of quality characteristics defined by the IEEE are the primary data obtained through simple software features recording levels.

*Compare function*. Intended use of quality metrics is to examine the terms of the quality characteristics of an entity and to make comparisons with it, placing it in a defined category of software, or with other entities, placing them on a certain stage in the hierarchy.

*Function analysis* gives meaningful numerical results obtained by applying mathematical models associated to metrics. Based on the numbers obtained, entities are granted quality characteristics.

*Function synthesis*. When investigating groups of entities homogeneous groups of entities are created and on their basses are summarized essential characteristics for the entire class of entities.

*Function estimation*. Software metric used to measure the trend of increase/decrease the quality level of the entities.

*Check function*. The results obtained by applying software metrics are used to confirm and strengthen or refute the findings obtained by other methods. Using a software metrics validation involves its validation and the independence of results.

In [8], a template for defining and documenting an indicator is provided. In accordance with that template, the following fields are included [8]:

- Precise objective of the indicator: description of the purpose of the indicator;
- Inputs: list of the data elements that are used in indicator applying;
- Algorithms: the algorithm or formula required to combine data elements;
- Assumptions: business environment, business processes and so forth, as conditions for collecting and using the indicator;
- Data collection information: description of how, when, how often and by whom data are to be collected; also, if a standard form is used to collect data this is referenced;
- Data reporting information: specification of who is responsible to report data, who is going to do reporting and to whom it is going, how often the data will be reported;
- Analysis and interpretation of results: identifying the meaning of the different values for indicator.

Identifying the indicators is made on a methodology as it is presented in [8]. The steps of the methodology are [8]:

Step.1 Identifying the goals;
Step.2 Identifying what it wants to know;
Step.3 Identifying the sub-goals;
Step.4 Identifying the entities and attributes;
Step.5 Formalizing the measurement goals;
Step.6 Identifying the measurement questions and indicators;
Step.7 Identifying the data elements;
Step.8 Defining and documenting measures and indicators;
Step.9 Identifying the actions needed to implement measures;
Step.10 Preparing a plan.

Depending on measurement scale, the indicators are classified in the following classes:

- Qualitative: measurement scale has discreet units as very good, good, satisfactory, poor, very poor;
- Quantitative: assessment is a very precise one and it is a numerical one.

Depending on aggregation level, the indicators included in an assessment system are classified in the following classes [9]:

- Primary indicators (metrics): they are computed in a single process, within department/module/ component and they aim the primary characteristics of the analyzed system;
- Aggregated indicators: they are the results of many assessment processes, multiple applications or aggregation operations of the primary metrics.

To create operational metrics, the following steps are necessary to be taken:

- defining the exogenous variable and the exact, unambiguous measurement manner;
- building software that automatically measure the levels of exogenous variables from the program;
- verifying in all cases that this software is highlighted and indeed the obtained levels are accurate, free from imperfections of definition/identification contained in the modules;
- establishing precise rules for building the test examples or structure;
- identifying the real casuistry through which are recorded levels of behavior;
- creating the indispensability that the software producers to use the same product for evaluating the metrics for a factor, ensuring in this way the compatibility and therefore the comparability of the results.

Defining the quality metrics is an important step in analyzing the entities' quality, as building the metrics of superior quality provides identification of the qualitative entities.

## 3. ORTHOGONALITY ASSESSMENT

Orthogonality studies the similarity level between two or more entities. Through this quality characteristics is determined the degree in which the entities differ one from another [1].

The orthogonality concept is derived from mathematics, where it has in view the following aspects [3]:

- two planes are orthogonal if the angle formed at their intersection has the cosinus equal to zero; a finite set of planes is orthogonal if the planes are perpendicular two by two;
- two lines are orthogonal if they form congruent adjacent angles; a finite set of lines is orthogonal if the lines are perpendicular two by two;
- two vectors are orthogonal if their scalar product is null; a finite set of vectors is orthogonal if the scalar product of any of two different vectors is null.

Orthogonality is studied on the basis of the orthogonality criteria. With the help of these criteria are stated the characteristics which have the same value for the studied entities and are determined the similarity levels.

An area in which orthogonality is very important is programming. The programming languages are so designed that their orthogonality is maximum, meaning that the implemented notions, the terms and the key words used are unique in order to avoid confusion among the users and especially in order to keep the compilers from not knowing the meaning of a code line, due to several possible options.

The comparison of two entities is reduced to reporting an entity to the other one, respectively identifying the similar parts and the differing ones. Thus are compared the correspondent characteristics of the two entities.

With the purpose of studying orthogonality, is defined a transformed orthogonality indicator within the [0,1] range, which takes the following values [2]:

- 1, if the elements are orthogonal, meaning they have nothing in common;
- 0, the elements are identical, meaning they do not have different values for any taken characteristics;

If the value of the indicator is close to one means that the data sets contained by the two entities tend to orthogonality, and if the value of the indicator is close to 0 it means the data sets have very many identical elements. The entities' orthogonality is evaluated depending on their content, in this way:

- orthogonality at text level, which consists in defining and quantifying the primary indicators; there are identified, selected and classified the quality characteristics of the texts for which is made the analysis, these determining also the granularity of the analysis, are aggregated the values of the indicators in order to obtain the overview of the studied phenomenon;
- the orthogonality of the constructions with numerical and alphanumerical representation, such as tables, figures, diagrams; are identified and analyzed the structural elements, are developed the indicators for the compared analysis of the text's structures and elements and is applied the aggregation operation for the values obtained for the two component types.

Two vectors, X = $\{X_1, X_2, \ldots, X_N\}$ and Y = $\{Y_1, Y_2, \ldots, Y_N\}$ are orthogonal if their scalar product is null, respectively equals 0. Starting from the vectors' orthogonality the following algorithm is proposed to establish the orthogonality of two texts.

Texts are considered, $T_1$ and $T_2$, formed of the words $T_1 = \{c_{11}, c_{12}, \ldots, c_{1n}\}$ and $T_2 = \{c_{21}, c_{22}, \ldots, c_{2m}\}$. The words which make up the two texts are characterized by the position held.

After ordination, the component words of the two texts are compared, the result being 1, if the words of the two texts are similar, and 0, otherwise. Below, an algorithm applied on the two defined texts is presented.

The texts $T_1$ and $T_2$ are considered:

$$T_1 = \{t_{11}, t_{12}, t_{13}, \ldots, t_{1N}\}$$
$$T_2 = \{ t_{21}, t_{22}, \ldots, t_{2M} \}$$

For the analysis of the orthogonality of the two texts are made the following steps:

$P_1$: from the texts are taken out the connecting words;

$P_2$: also, are taken out the repeated words, a single term being left;

$P_3$: the words are stored in a word vector:

$$V_1 = (v_{11}, v_{12}, v_{13}, \ldots, v_{1N})$$
$$V_2 = (v_{21}, v_{22}, \ldots, v_{2M}).$$

$P_4$: the word vector is put in order;

$P_5$: it is analyzed the word vector, in the following way:

– if the word on position i of the $V_1$ vector can also be found in $V_2$, is allocated value 1 for the resulted vector;

– else, is allocated value 0.

The two vectors resulted are:

$$VR_1 = (vr_{11},vr_{12},\ldots,vr_{1N})$$
$$VR_2 = (vr_{21},vr_{22},\ldots,vr_{2M});$$

$P_6$: it is determined the scalar product of the resulted vectors and its value is interpreted.

In order to establish the orthogonality of $T_1$ and $T_2$ texts, the formula below is used:

$$H_{TK/TL} = 1 - \frac{\sum_{i=1}^{n} \mathrm{VRK}_i}{n}, \quad (1)$$

where $\mathrm{VRK}_i$ is the vector resulted, corresponding to text TK, and $N$ represents the dimension of text TK.

Orthogonality uses a series of indicators, such as:

– length, which is applied on the stored text with the help of the entity, offering information on the number of words used, on the number of letters used or on the number of key words;

– appearance frequency of the words is an indicator applied on the studied texts in order to determine the orthogonality level;

– the distance between key words, identify the position of each key word, and also the number of words between two consecutive key words.

The texts T$_1$ and T$_2$, are considered representing two procedures that compute the sum, respectively the maximum of three numbers:

```
procedure suma(int a, int b, int c){
int nr;
nr:=0; nr:=a+b+c;
printf("%d", nr);
}
```

```
procedure maxim(int a, int b, int c){
int nr;
if (a>b) nr:= a; else nr:= b;
if (nr<c) nr:=c;
printf("%d", nr); }
```

On the two procedures it is applied the $H_L$ orthogonality indicator, which compares two or more texts, by reporting to their length and it is defined by the following formula:

$$H_L = \frac{LG(T_i)}{LG(T_j)}, \tag{2}$$

where LG($T_i$) < LG($T_j$), and is constituted as a weight of the smaller text within the longer text.

Tables 1 and 2 are built containing the terms from which are formed the procedures, and the repeat frequencies:

*Table 1*

Repeat frequencies of terms in procedure sum()

| Term | Frequency |
|---|---|
| int | 1 |
| nr | 4 |
| : = | 2 |
| + | 2 |
| printf | 1 |
| %d | 1 |
| int | 1 |
| nr | 4 |

*Table 2*

Repeat frequencies of terms in procedure maxim()

| Term | Frequency |
|---|---|
| int | 1 |
| nr | 6 |
| if | 2 |
| > | 1 |
| : = | 3 |
| else | 1 |
| < | 1 |
| printf | 1 |

For this example considered $H_L = 6/9 = 0.67$.

The degree in which the two texts use the same base of words in the vocabulary is given by computing the following indicator [10]:

$$H_{FC} = \frac{\min\{Lgv(V_{co}),\ Lgv(V_{re})\}}{\max\{Lgv(V_{co}),\ Lgv(V_{re})\}}, \tag{3}$$

where $V_{co}$ is vocabulary of common words in the two texts, and $V_{re}$ represents reunion vocabulary of the texts under the comparative study.

The weight of the identical elements in the two texts is given by the ratio, [10]:

$$H_{PI} = \frac{ncrf}{Lgv(V_{re})}, \tag{4}$$

where *ncrf* is the number of words in the reunion vocabulary which have the same appearance frequency, and $Lgv(V_{re})$ represents the length of the reunion vocabulary.

For any two random source texts ET$_{Si}$ and ET$_{Sj}$, the form of the orthogonality indicator for *m* metrics is:

$$H_{ij} = 1 - \frac{\min(M_i^m,\ M_j^m)}{\max(M_i^m,\ M_j^m)}, \tag{5}$$

where $M_i^m$ represents the value of $m$ metrics for the source program $ET_{Si}$, and $M_j^m$ represents the value of $m$ metrics for the source program $ET_{Sj}$.

The definition of metrics on texts offers the numerical expression of their quality and allows making quantitative. Indicator with synthetic character, $I_{AC}$, is built. It regards reaching the quality objectives of a text entity:

$$I_{AC} = \sum_{i=1}^{nca} pd_i \frac{\min(Ch_i^{pl},\ Ch_i^{ef})}{\max(Ch_i^{pl},\ Ch_i^{ef})}, \tag{6}$$

where $pd_i$ is weight $I$ associated to the quality characteristic $Ch_i$, and $nca$ is the number associated to the quality characteristic;

Orthogonality certifies the originality of the structured entities, materialized under diverse forms: scientific articles, research project with financing, handbooks, scientific theses, legislative texts. Building up the methods of identifying orthogonality assures the concrete validation means for validating the elements under analysis.

## 4. EXPERIMENTAL RESULTS

For the analysis and evaluation of the orthogonality characteristic of structured text entities, was developed the ORTOES application. It is built with the purpose of analyzing the differentiation degree between two or more structured entities. ORTOES application is composed by four modules (Fig. 2), taking over data, analysis of data, display of results and administration.

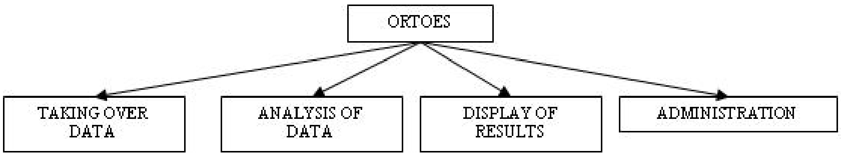

Fig. 2 – The structure of the software product ORTOES.

These modules were built in order to assure the interaction with the users, the processing of data input by them, the display of results, both at individual and aggregate level, and also the administration of the application functionalities.

ORTOES application is tested having in view data sets from the users' collectivity.

The users' collectivity is formed by 482 members, out of which for the structured entity composed of texts $T_1$, $T_2$, $T_3$, interdependent, were checked the orthogonalities, according to Table 3:

*Table 3*

The weight of users in the calculus of orthogonality

| Entity | Weight |
|---|---|
| Structured entity project $T_1$ | 78% |
| Structured entity project $T_2$ | 55% |
| Structured entity project $T_3$ | 63% |

The orthogonality levels for the three entities, presented by ranges, are presented in Table 4:

*Table 4*

Weight of orthogonality level

| Entity | Orthogonality in range [0; 0.75) | Orthogonality in range [0.75; 0.85) | Orthogonality in range [0.85; 1.00] |
|---|---|---|---|
| $T_1$ | 10% | 0.002% | 89.998% |
| $T_2$ | 0% | 25.6% | 74,4% |
| $T_3$ | 0% | 35.4% | 64.6% |

In the case that there were made re-inputs of texts in order to increase orthogonality, were recorded the evolutions presented in Table 5:

*Table 5*

The orthogonality evolution of the structured entity $T_1$

| Number of re-input | Number of users who re-input entity $T_1$ | Orthogonality in range [0; 0,75) | Orthogonality in range [0,75; 1,00] |
|---|---|---|---|
| first re-input | 54 | 4 | 50 |
| second re-input | 4 | 0 | 4 |

The result is a synthetic approach obtained by modifying 54 entities, in 92,59% of the cases for the first modification was recorded an increase of orthogonality over the imposed limit of 0,75, and for the second modification, was recorded an increase of orthogonality in 100% of the cases by respecting the limit that the orthogonality of entity $T_1$ is over the threshold of 0,75, which shows the users' tendency to realize entities with a high orthogonality level.

## 5. CONCLUSIONS

ORTOES application assures the processing of source texts of applications with the purpose of establishing their orthogonality level. Building up applications with a high orthogonality level and imposing restrictions regarding the ranges in which they should be included, and also respecting precise specifications, imposed when developing informatics applications, contribute to increasing the applications' homogeneity level, with the view of their better usage by citizens.

The quality of the entities from the orthogonality point of view is established also by using the software product with which is highlighted the originality degree of a paper, and also its inclusion in a certain paper category: synthesis, original paper, compilation, unoriginal paper.

The existence in electronic format of specialized magazines and books leads to the creation of virtual entities and by elaborating software which operates on such entities are created the premises of orthogonality analysis on every content generated.

Building up ORTOES application assures the automatic character of the process of establishing the orthogonality for structured entities, stored in electronic format. Building up orthogonality metrics assures the implementation of qualitative indicators, fact which assures a high level of the testing process.

## REFERENCES

1. ION IVAN, DANIEL MILODIN, MARIUS POPA, *Operation on text entities*, Informatica Economica, **11**, *1*, pp. 14–20, 2007.
2. ION IVAN, DANIEL MILODIN, *Data Orthogonality Metrics*, Workshop Information Systems and Operations Management, Romanian American University, Bucharest, 2006, pp. 326–341.
3. DANIEL MILODIN, LEONARD SĂCUIU, *Probleme ale ortogonalității aplicațiilor informatice neomogene*, Simpozionul Internațional al Tinerilor Cercetători (Ediția a VI-a), ASEM Chişinău, 18–19 April 2008, vol. I, ASEM Printing House, Chişinău, pp. 341–343, 2008.
4. DANIEL MILODIN, SORIN DUMITRU, *The security of the application for evaluating the text entities' orthogonality*, Conferința Internațională de Informatică, ediția a 9-a, ASE Printing House, Bucureşti, pp. 897–902, 2009.
5. MARIUS POPA, *Evaluarea calității entităților text. Teorie şi Practică*, ASE Printing House, Bucureşti, 2005.
6. CĂTĂLIN BOJA, ION IVAN, *Metode statistice în analiza software*, ASE Printing House, Bucureşti, 2004.
7. MICHAEL BERRY, GORDON LINOFF, *Data Minig Tehniques*, John Wiley, New York, 2004.
8. WOLFHART GOETHERT, WILL HAYES, *Experiences in Implementing Measurement Programs*, Software Engineering Measurement and Analysis Initiative, Carnegie Mellon University, Technical Note, November 2001.
9. MARIUS POPA, *Characteristics for Development of an Assessment System for Security Audit Processes*, Economy Informatics, **9**, *1*, pp. 55–62, 2009.
10. ION IVAN, MARIUS POPA, PAUL POCATILU, *The Fingerprint – an Unique Way to Identify Programs*, Proceedings of the International Symposium "Knowledge Technologies in Business and Management", Iaşi, 6 June 2003, pp. 40–45, 2003.
11. TUDOR BARBU, *An Automatic Unsupervised Pattern Recognition Approach*, Proceedings of the Romanian Academy, **7**, *1*, 2006.
12. ION IVAN, CĂTĂLIN BOJA, MIHAI DOINEA, Determining the Optimal Length of Useful Information, Economic Computation and Economic Cybernetics Studies and Research, **42**, *1–2*, pp. 23–34, 2008.
13. GHEORGHE PĂUN, *Grammar systems. A grammatical approach to distribution and cooperation*, Invited lecture ICALP 1995, LNCS 944 (Z. Fulop, F. Gecseg, eds.), Springer-Verlag, pp. 429–443, 1995.

*Received January 4, 2010*